



Sri Shridevi Charitable Trust (R.)
SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
Sira Road, Tumkur - 572 106, Karnataka, India.

Phone: 0816 - 2212629 | Principal: 0816 - 2212627, 9686114899 | Telefax: 0816 - 2212628

Email: info@shrideviengineering.org, principal@shrideviengineering.org | Website: www.shrideviengineering.org

(Approved by AICTE, New Delhi, Recognised by Govt. of Karnataka and Affiliated to Visvesvaraya Technological University, Belagavi)

ESTD: 2002



Criteria 1.1

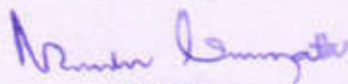
Curriculum Planning and Implementation

Course Files (CSE)

PRINCIPAL
SIET, TUMAKURU.

COURSE FILE - CHECK LIST

1. Calendar of events
2. Class time table
3. Individual time table
4. Syllabus copy
5. Lesson plan
6. Student Name List
7. Notes according to syllabus
8. Internal Test Question Paper & Scheme of valuation
9. University Question Papers



PRINCIPAL
SIET, TUMAKURU



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY, TUMKUR-572106

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CALENDAR OF EVENTS

VI & VIII Semester B E APRIL - July 2022 Session

APRIL			MAY			JUNE			JULY		
Date	Day	Activities	Date	Day	Activities	Date	Day	Activities	Date	Day	Activities
* 4/4/22	Mon	# Commencement @	1/5/22	Sun		1/6/22	Wed		1/7/22	Fri	IA Test-III(8th SEM)
5/4/22	Tue	\$Deptl Staff Meeting @	* 2/5/22	Mon	HODs Meeting	2/6/22	Thu	MockTest :C/C++/Quiz	2/7/22	Sat	
6/4/22	Wed	Softskills Training @	3/5/22	Tue	Basava Jayanathi	3/6/22	Fri		3/7/22	Sun	
7/4/22	Thu	Softskills Training @	4/5/22	Wed	Deptl Staff Meeting	4/6/22	Sat		* 3/7/22	Mon	# Lab IA Test
8/4/22	Fri	Softskills Training @	5/5/22	Thu	Expert Lecture	5/6/22	Sun		5/7/22	Tue	\$ Lab IA Test
9/4/22	Sat	Softskills Training @	6/5/22	Fri		* 6/6/22	Mon	HODs Meeting	6/7/22	Wed	Lab IA Test
10/4/22	Sun		7/5/22	Sat		7/6/22	Tue	Deptl Staff Meeting	7/7/22	Thu	Lab IA Test
* 11/4/22	Mon	HODs Meeting	8/5/22	Sun		8/6/22	Wed		8/7/22	Fri	Lab IA Test
12/4/22	Tue	Deptl Staff Meeting	* 9/5/22	Mon		9/6/22	Thu	Webinar[Experts]	9/7/22	Sat	Lab IA Test
13/4/22	Wed		10/5/22	Tue		10/6/22	Fri	Sriutsav2022	10/7/22	Sun	Bakrid
14/4/22	Thu	Ambedkar Jayanathi	11/5/22	Wed		11/6/22	Sat	Sriutsav2022	* 11/7/22	Mon	
15/4/22	Fri	Good Friday	12/5/22	Thu	Interview skills	12/6/22	Sun		12/7/22	Tue	
16/4/22	Sat		13/5/22	Fri		* 13/6/22	Mon	HODs Meeting	13/7/22	Wed	
17/4/22	Sun		14/5/22	Sat		14/6/22	Tue	Deptl Staff Meeting	14/7/22	Thu	IA Test - III(6thsem)
* 18/4/22	Mon	HODs Meeting	15/5/22	Sun		15/6/22	Wed		15/7/22	Fri	IA Test - III(6thsem)
19/4/22	Tue	Deptl Staff Meeting	* 16/5/22	Mon	HODs Meeting	16/6/22	Thu	Expert Lecture	16/7/22	Sat	IA Test - III(6thsem)
20/4/22	Wed		17/5/22	Tue	Deptl Staff Meeting	17/6/22	Fri	IA Test-II(6&8thsem)			
21/4/22	Thu		18/5/22	Wed		18/6/22	Sat	IA Test - II(6thsem)	VIII Semester		
22/4/22	Fri		19/5/22	Thu	Expert Lecture	19/6/22	Sun		Total No of Working Days: 73		
23/4/22	Sat		20/5/22	Fri	IA Test-I(6&8thsem)	* 20/6/22	Mon	IA Test - II(6thsem)	* Last Working Day: 30/06/2022		
24/4/22	Sun		21/5/22	Sat	IA Test-I(6thsem)	21/6/22	Tue	Deptl Staff Meeting	Theory Examinations: 04/07/2022		
* 25/4/22	Mon	HODs Meeting	22/5/22	Sun		22/6/22	Wed		Project/Internship Viva: 22/07/2022		
26/4/22	Tue	Deptl Staff Meeting	* 23/5/22	Mon	IA Test - I(6thsem)	23/6/22	Thu	Symposium			
27/4/22	Wed		24/5/22	Tue	Deptl Staff Meeting	24/6/22	Fri	Industrial Visit	VI Semester		
28/4/22	Thu	Extempore	25/5/22	Wed		25/6/22	Sat		Total No of Working Days: 87		
29/4/22	Fri		26/5/22	Thu	Aptitude Test	26/6/22	Sun		Last Working Day: 16/07/2022		
30/4/22	Sat		27/5/22	Fri		* 27/6/22	Mon	HODs Meeting	Practical Examinations: 18/07/2022		
			28/5/22	Sat	Cycle Rally	28/6/22	Tue	Deptl Staff Meeting	Theory Examinations: 01/08/2022		
			29/5/22	Sun		29/6/22	Wed	Workshop	Go Green		
			* 30/5/22	Mon	HODs Meeting	30/6/22	Thu	Closure for VIII Sem*	There is no Planet B		
			31/5/22	Tue	Deptl Staff Meeting				No of Working Days: 14		
No of Working Days: 22			No of Working Days: 25			No of Working Days: 26					

(Prof SHANMUKASWAMY CV)
Head of the Department

(Dr Narayana Viswanath)
Principal



SHRIDEVI INSTITUTE OF ENGINEERING AND TECHNOLOGY, TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE & ENGG
Academic year Academic year 2021-22, EVEN Semester



CLASS TIME TABLE

W.E.F: 04/04/2022

Class: VIII Sem [2018 Scheme]

Lecture Hall:109

TIME DAY	8:45 A.M To 9:40 A.M	9:40 A.M To 10:35 A.M	10:35 A.M To 10:55 A.M	10:55 A.M To 11:50 A.M	11:50 A.M To 12:45 P.M	12:45P.M To 1:45 P.M	1:45 P.M To 2:40 P.M	2:40 P.M To 3:35 P.M	3:35 P.M To 4:30 P.M
	MONDAY	PROJECT WORK Phase -II					LUNCH BREAK	PROJECT WORK Phase -II	
TUESDAY	PROJECT WORK Phase -II					PROJECT WORK Phase -II			
WEDNESDAY	PROJECT WORK Phase -II					PROJECT WORK Phase -II			
THURSDAY	IOT [MSC]	IOT [MSC]	Tea Break	SAN [RS]	SAN [RS]	Dept. Activty			
FRIDAY	SAN [RS]	SAN [RS]		IOT [MSC]	IOT [MSC]	Activity Points Events			
SATURDAY	IOT [MSC]	SAN [RS]		Internship Seminar	Technical Seminar				

COURSE CODE

18CS81
18CS82
18CSP83
18CSS84
18CSI85

NAME OF THE COURSE

Internet of Things
Storage Area Networks
Project work phase -2
Technical Seminar
Internship

NAME OF THE STAFF

Chethan M S
Mr.Suthan R
Mr.Suthan R
Dr. Charan KV
Mr. Kiran G M

COUNSELOR

A1: Mr.Renukaradhya PC
A2: Mrs .Ayesha Khanum(*CT)

[Mrs.Veena N D]
Time Table Co-Ordinator

[Prof. C V Shanmugaswamy]
Head , Dept. of CSE

[Dr. Narendra Vissanath]
Principal

PRINCIPAL
SIET, TUMAKURU.

STORAGE AREA NETWORKS			
(Effective from the academic year 2018 -2019)			
SEMESTER – VII			
Course Code	18CS822	CIE Marks	40
Number of Contact Hours/Week	3:0:0	SEE Marks	60
Total Number of Contact Hours	40	Exam Hours	03
CREDITS –3			
Course Learning Objectives: This course (18CS822) will enable students to:			
<ul style="list-style-type: none"> • Evaluate storage architectures, • Define backup, recovery, disaster recovery, business continuity, and replication • Examine emerging technologies including IP-SAN • Understand logical and physical components of a storage infrastructure • Identify components of managing and monitoring the data center • Define information security and identify different storage virtualization technologies 			
Module 1			Contact Hours
Storage System: Introduction to Information Storage: Information Storage, Evolution of Storage Architecture, Data Center Infrastructure, Virtualization and Cloud Computing. Data Center Environment: Application Database Management System (DBMS), Host (Compute), Connectivity, Storage, Disk Drive Components, Disk Drive Performance, Host Access to Data, Direct-Attached Storage, Storage Design Based on Application Textbook1 : Ch.1.1 to 1.4, Ch.2.1 to 2.10 RBT: L1, L2			08
Module 2			
Data Protection - RAID : RAID Implementation Methods, RAID Array Components, RAID Techniques, RAID Levels, RAID Impact on Disk Performance, RAID Comparison. Intelligent Storage Systems : Components of an Intelligent Storage System, Types of Intelligent Storage Systems. Fibre Channel Storage Area Networks - Fibre Channel: Overview, The SAN and Its Evolution, Components of FC SAN. Textbook1 : Ch.3.1 to 3.6, Ch. 4.1, 4.3, Ch. 5.1 to 5.3 RBT: L1, L2			08
Module 3			
IP SAN and FCoE: iSCSI, FCIP, Network-Attached Storage: General-Purpose Servers versus NAS Devices, Benefits of NAS, File Systems and Network File Sharing, Components of NAS, NAS I/O Operation, NAS Implementations, NAS File-Sharing Protocols, Factors Affecting NAS Performance Textbook1 : Ch.6.1, 6.2, Ch. 7.1 to 7.8 RBT: L1, L2			08
Module 4			
Introduction to Business Continuity: Information Availability, BC Terminology, BC Planning Life Cycle, Failure Analysis, Business Impact Analysis, BC Technology Solutions, Backup and Archive: Backup Purpose, Backup Considerations, Backup Granularity, Recovery Considerations, Backup Methods, Backup Architecture, Backup and Restore Operations, Backup Topologies, Backup in NAS Environments Textbook1 : Ch.9.1 to 9.6, Ch. 10.1 to 10.9 RBT: L1, L2			08
Module 5			
Local Replication: Replication Terminology, Uses of Local Replicas, Replica Consistency, Local Replication Technologies, Tracking Changes to Source and Replica, Restore and Restart Considerations, Creating Multiple Replicas. Remote Replication: Modes of Remote			08

<p>Replication, Remote Replication Technologies. Securing the Storage Infrastructure: Information Security Framework, Risk Triad, Storage Security Domains. Security Implementations in Storage Networking Textbook1 : Ch.11.1 to 11.7, Ch. 12.1, 12.2, Ch. 14.1 to 14.4 RBT: L1, L2</p>	
<p>Course Outcomes: The student will be able to :</p> <ul style="list-style-type: none"> • Identify key challenges in managing information and analyze different storage networking technologies and virtualization • Explain components and the implementation of NAS • Describe CAS architecture and types of archives and forms of virtualization • Illustrate the storage infrastructure and management activities 	
<p>Question Paper Pattern:</p> <ul style="list-style-type: none"> • The question paper will have ten questions. • Each full Question consisting of 20 marks • There will be 2 full questions (with a maximum of four sub questions) from each module. • Each full question will have sub questions covering all the topics under a module. • The students will have to answer 5 full questions, selecting one full question from each module. 	
<p>Textbooks:</p> <ol style="list-style-type: none"> 1. EMC Education Services, "Information Storage and Management", Wiley India Publications, 2009. ISBN: 9781118094839 	
<p>Reference Books:</p> <ol style="list-style-type: none"> 1. Paul Massiglia, Richard Barker, "Storage Area Network Essentials: A Complete Guide to Understanding and Implementating SANs Paperback", 1st Edition, Wiley India Publications, 2008 	


 PRINCIPAL
 SIET, TUMAKURU.

LESSON PLAN (OCT-FEB 2022) MICROSCHEDULE

SUBJECT	STORAGE AREA NETWORKS	STAFF NAME	Mr. SUTHAN R
SUBJECT CODE	18CS822	SEM/SEC	VIII
IA Marks (CIE)	40 (Average of three tests for 30 marks and 10 marks for assignment)	Maximum Exam Marks (SEE)	60 (Question paper will be set and evaluated for 100 marks and later reduced to 60)

MODULE 1					
Sl. No.	DATE	DAY	LESSON PLANNED	LESSON COVERED	REMARKS
1	16/04/22	SAT	Storage System: Introduction to Information Storage: Information Storage,	Covered	
2	21/04/22	THU	Evolution of Storage Architecture,	Covered	
3	21/04/22	THU	Data Center Infrastructure	Covered	
4	22/04/22	FRI	Virtualization and Cloud Computing	Covered	
5	22/04/22	FRI	Data Center Environment: Application Database Management System (DBMS),	Covered	
6	23/04/22	SAT	Host (Compute), Connectivity, Storage,	Covered	
7	28/04/22	THU	Disk Drive Components	Covered	
8	28/04/22	THU	Disk Drive Performance	Covered	
9	29/04/22	FRI	Host Access to Data,	Covered	
10	29/04/22	FRI	Direct-Attached Storage	Covered	

SUMMARY

PLANNED DATE	16.04.2022	TO: 29.04.2022	
ACTUAL CLASSES TAKEN	FROM: 16.04.2022	TO: 29.04.2022	
NUMBER OF CLASSES	ALLOCATED: 10	TAKEN: 10	
CONTENT COVERED FOR IA	IA 1: ✓	IA 2:	IA 3:
VALUE ADDITION TO THE MODULE	ASSIGNMENTS: ✓	TUTORIALS:	QP DISCUSSION: ✓
	QUIZ:	SEMINARS:	ANY OTHER:


Suthan R
Staff Incharge



Prof. C V Shanmukaswamy
HOD, CSE

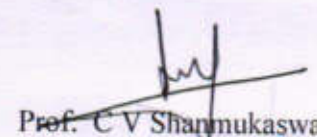

Dr. Narendra Viswanath
Principal
SIRI TUMAKURU

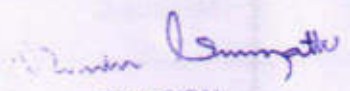
MODULE II					
Sl. No.	DATE	DAY	LESSON PLANNED	LESSON COVERED	REMARKS
11	30/04/22	SAT	Data Protection - RAID : RAID Implementation Methods	Covered	
12	04/05/22	THU	RAID Array Components, RAID Techniques	Covered	
13	04/05/22	THU	RAID Impact on Disk Performance	Covered	
14	05/05/22	FRI	RAID Comparison. Intelligent Storage Systems : Components of an Intelligent Storage System	Covered	
15	05/05/22	FRI	Components of an Intelligent Storage System	Covered	
16	06/05/22	SAT	Types of Intelligent Storage Systems.	Covered	
17	12/05/22	THU	Fibre Channel Storage Area Networks - Fibre Channel: Overview	Covered	
18	12/05/22	THU	The SAN and Its Evolution	Covered	
19	13/05/22	FRI	Components of FC SAN.	Covered	
20	13/05/22	FRI	Components of FC SAN.	Covered	

SUMMARY

PLANNED DATE	FROM: 30.04.2022	TO: 13.05.2022	
ACTUAL CLASSES TAKEN	FROM: 30.04.22	TO: 13.05.22	
NUMBER OF CLASSES	ALLOCATED: 10	TAKEN: 10	
CONTENT COVERED FOR IA	IA 1: ✓ 70%	IA 2: ✓ 30%	IA 3:
VALUE ADDITION TO THE MODULE	ASSIGNMENTS: ✓	TUTORIALS:	QP DISCUSSION: ✓
	QUIZ:	SEMINARS:	ANY OTHER: 1


Suthan R
Staff Incharge


Prof. C V Shanmukaswamy
HOD, CSE


PRINCIPAL
SIB. TUMAKURU
Dr. Narendra Viswanath
Principal


MODULE III

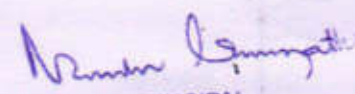
Sl No.	DATE	DAY	LESSON PLANNED	LESSON COVERED	REMARKS
21	14/05/22	SAT	IP SAN and FCoE: iSCSI, FCIP,	Covered	
22	19/05/22	THU	Network-Attached Storage: General-Purpose Servers versus NAS Devices,	Covered	
23	19/05/22	THU	Benefits of NAS,	Covered	
24	21/05/22	SAT	File Systems and Network File Sharing	Covered	
25	26/05/22	THU	File Systems and Network File Sharing	Covered	
26	26/05/22	THU	Components of NAS, ,	Covered	
27	27/05/22	FRI	NAS I/O Operation	Covered	
28	27/05/22	FRI	NAS Implementations	Covered	
29	28/05/22	SAT	NAS File-Sharing Protocols	Covered	
30	02/06/22	THU	Affecting NAS Performance	Covered	

SUMMARY

PLANNED DATE	FROM: 14.05.2022	TO: 02.06.2022	
ACTUAL CLASSES TAKEN	FROM: 14.05.22	TO: 2.06.22	
NUMBER OF CLASSES	ALLOCATED: 10	TAKEN: 10	
CONTENT COVERED FOR IA	IA 1:	IA 2: ✓	IA 3:
VALUE ADDITION TO THE MODULE	ASSIGNMENTS: ✓	TUTORIALS:	QP DISCUSSION: ✓
	QUIZ:	SEMINARS:	ANY OTHER:


Suthan R
Staff Incharge


Prof. C V Shanmukaswamy
HOD, CSE

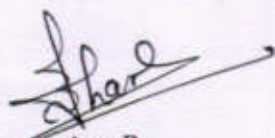

PRINCIPAL
SIET, TUMAKURU
Dr. Narendra Viswanath
Principal


MODULE IV

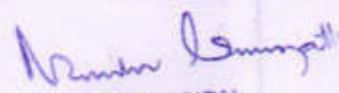
Sl. No.	DATE	DAY	LESSON PLANNED	LESSON COVERED	REMARKS
31	02/06/22	THU	Introduction to Business Continuity: Information Availability	Covered	
32	03/06/22	FRI	BC Terminology, BC Planning Life Cycle	Covered	
33	03/06/22	FRI	Failure Analysis, Business Impact Analysis	Covered	
34	04/06/22	SAT	BC Technology Solutions	Covered	
35	09/06/22	THU	Backup and Archive: Backup Purpose	Covered	
36	09/06/22	THU	Backup Considerations, Backup Granularity	Covered	
37	10/06/22	FRI	Recovery Considerations, Backup Methods	Covered	
38	10/06/22	FRI	Backup Architecture	Covered	
39	11/06/22	SAT	Backup and Restore Operations, Backup Topologies, Backup in NAS Environments	Covered	

SUMMARY

PLANNED DATE	FROM: 02.06.2022	TO: 11.06.2022	
ACTUAL CLASSES TAKEN	FROM: 02.06.22	TO: 11.06.22	
NUMBER OF CLASSES	ALLOCATED: 10	TAKEN: 10	
CONTENT COVERED FOR IA	IA 1:	IA 2: ✓ 50%	IA 3:
VALUE ADDITION TO THE MODULE	ASSIGNMENTS: ✓	TUTORIALS:	QP DISCUSSION: ✓
	QUIZ:	SEMINARS:	ANY OTHER:


Suthan R
Staff Incharge



Prof. C V Shammukaswamy
HOD, CSE



PRINCIPAL
SIET., TUMAKURU
Dr. Narendra Viswanath
Principal

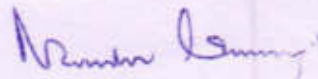
MODULE V					
Sl. No.	DATE	DAY	LESSON PLANNED	LESSON COVERED	REMARKS
40	16/06/22	THU	Local Replication: Replication Terminology, Uses of Local Replicas,	Covered	
41	16/06/22	THU	Replica Consistency , Local Replication Technologies	Covered	
42	23/06/22	THU	Tracking Changes to Source and Replica , Restore and Restart Considerations	Covered	
43	23/06/22	THU	Creating Multiple Replicas, Remote Replication: Modes of Remote Replication	Covered	
44	24/06/22	FRI	Remote Replication Technologies	Covered	
45	24/06/22	FRI	Securing the Storage Infrastructure: Information Security Framework,	Covered	
46	25/06/22	SAT	Information Security Framework,	Covered	
47	25/06/22	SAT	Risk Triad,	Covered	EXTRA
48	25/06/22	SAT	Storage Security Domains	Covered	EXTRA
49	30/06/22	THU	Security Implementations in Storage Networking	Covered	
50	30/06/22	THU	REVISION	Covered	

SUMMARY

PLANNED DATE	FROM: 16.06.2022	TO: 30.06.2022	
ACTUAL CLASSES TAKEN	FROM: 16.06.22	TO: 30.6.22	
NUMBER OF CLASSES	ALLOCATED: 10	TAKEN: 10	
CONTENT COVERED FOR IA	IA 1:	IA 2:	IA 3: ✓
VALUE ADDITION TO THE MODULE	ASSIGNMENTS: ✓	TUTORIALS:	QP DISCUSSION: ✓
	QUIZ:	SEMINARS:	ANY OTHER:


Suthan R
Staff Incharge


Prof. C V Shanmukaswamy
HOD, CSE


PRINCIPAL
SIET, TUMAKURU
Dr. Narendra Viswanath
Principal

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
LESSON PLAN (APR-JUN 2022) MACROSCHEDULE

SUBJECT	STORAGE AREA NETWORKS	STAFF NAME	Mr. SUTHAN R
SUBJECT CODE	18CS822	SEM/SEC	VIII
IA Marks (CIE)	40 (Average of three tests for 30 marks and 10 marks for assignment)	Maximum Exam Marks (SEE)	60 (Question paper will be set and evaluated for 100 marks and later reduced to 60)

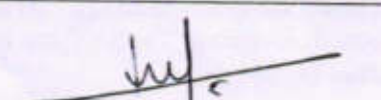
Course Outcomes or COs

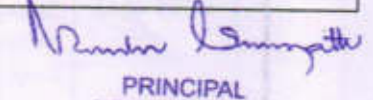
- CO1: Evaluate storage architectures
- CO2: Define backup, recovery, disaster recovery, business continuity, and replication
- CO3: Examine emerging technologies including IP-SAN
- CO4: Understand logical and physical components of a storage infrastructure
- CO5: Identify components of managing and monitoring the data center
- CO6: Define information security and identify different storage virtualization technologies

Sl. No.	DATE	MODULE LESSON PLAN	ADDITIONAL SOURCES
1.	16.04.2022 to 29.04.2022	Module 1 Storage System: Introduction to Information Storage: Information Storage, Evolution of Storage Architecture, Data Center Infrastructure, Virtualization and Cloud Computing. Data Center Environment: Application Database Management System (DBMS), Host (Compute), Connectivity, Storage, Disk Drive Components, Disk Drive Performance, Host Access to Data, Direct-Attached Storage, Storage Design Based on Application	https://www.youtube.com/watch?v=0IAPZzGSbME&list=PLDN4rrl48XKpZk03iYFI-O29szjTrs_O https://www.youtube.com/watch?v=GQNT0v5zKhE&list=PLrjkTqL3jnm8wGQvNhgdm2gkoa8CXcmI https://www.youtube.com/watch?v=twE1eiO7gEE&list=PL-JvKqQx2Atd--1Gs3WB8nmWOWRbEM7WW
2.	30.04.2022 to 13.05.2022	Module 2 Data Protection - RAID : RAID Implementation Methods, RAID Array Components, RAID Techniques, RAID Levels, RAID Impact on Disk Performance, RAID Comparison. Intelligent Storage Systems : Components of an Intelligent Storage System, Types of Intelligent Storage Systems. Fibre Channel Storage Area Networks - Fibre Channel: Overview, The SAN and Its Evolution, Components of FC SAN.	https://www.youtube.com/watch?v=2Rr2tW9zvRg https://www.youtube.com/watch?v=YAE1FsYGotA https://www.youtube.com/watch?v=PfEmF07S4hw

3.	14.05.2022 to 02.06.2022	Module 3 IP SAN and FCoE: iSCSI, FCIP, Network-Attached Storage: General-Purpose Servers versus NAS Devices, Benefits of NAS, File Systems and Network File Sharing, Components of NAS, NAS I/O Operation, NAS Implementations, NAS File-Sharing Protocols, Factors Affecting NAS Performance	https://www.youtube.com/watch?v=ARvQcqJ_-NY https://www.youtube.com/watch?v=IMOpX3q9NJI https://www.youtube.com/watch?v=GQNT0v5zKhE&list=PLrjkTqI3jnm8wGOyNhgdm2gk_oa8CXCml
4.	02.06.2022 to 11.06.2022	Module 4: Introduction to Business Continuity: Information Availability, BC Terminology, BC Planning Life Cycle, Failure Analysis, Business Impact Analysis, BC Technology Solutions, Backup and Archive: Backup Purpose, Backup Considerations, Backup Granularity, Recovery Considerations, Backup Methods, Backup Architecture, Backup and Restore Operations, Backup Topologies, Backup in NAS Environments	https://www.youtube.com/watch?v=5dRGRueKU3M https://www.youtube.com/watch?v=5dRGRueKU3M&list=PLJULIlvHz0rE83NKhnq7acXYIeA0o1dXb https://www.youtube.com/watch?v=IVR2u9Isx18&list=PLdo5W4Nhv31aBrJE1WS4MR9LR_fbmZrAQu https://www.youtube.com/watch?v=oNI0rf2P9gE https://www.youtube.com/watch?v=Gc4mWrmJBsw
5.	16.06.2022 to 30.06.2022	MODULE-5: Local Replication: Replication Terminology, Uses of Local Replicas, Replica Consistency, Local Replication Technologies, Tracking Changes to Source and Replica, Restore and Restart Considerations, Creating Multiple Replicas. Remote Replication: Modes of Remote Replication, Remote Replication Technologies. Securing the Storage Infrastructure: Information Security Framework, Risk Triad, Storage Security Domains. Security Implementations in Storage Networking	https://www.youtube.com/watch?v=DKCbsiDBN6c https://www.youtube.com/watch?v=xFv_Hl4B83A https://www.youtube.com/watch?v=Lm378j0CuYM&list=PL8xmnXn7pVtwcQRTWxhyGA7OfmGcNBsbv https://www.youtube.com/watch?v=3RBNPc0_Q6g https://www.youtube.com/watch?v=1FEP_sNb62k https://www.youtube.com/watch?v=PfkBS9qIMRE https://www.youtube.com/watch?v=nLmhmb6NzcM


Suthan R
Staff Incharge



Prof. C V Shanmukaswamy
HOD, CSE

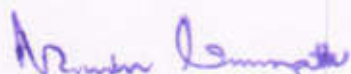

PRINCIPAL
SIET, TUMAKURU
Dr. Narendra Viswanath
Principal

STUDENT LIST FOR THE ACADEMIC YEAR 2021-22(EVEN SEM)

CLASS: VIII SEM

Sl No	USN	Name
1	1SV17CS011	CHAITRA M S
2	1SV17CS015	GAGANASHREE T U
3	1SV17CS018	JUNAID ULLA KHAN
4	1SV17CS024	MANASA V
5	1SV17CS027	NAVYA S
6	1SV17CS034	RAGHU RAM GK
7	1SV18CS001	ABDULLAH
8	1SV18CS003	AMULYA J M
9	1SV18CS004	AYUSH RANJAN TIWARI
10	1SV18CS005	BASAVARAJA
11	1SV18CS007	BHAVYA H P
12	1SV18CS008	CHANDRASHEKARA T
13	1SV18CS011	DHARMANA HARIKA
14	1SV18CS014	ENCHARA M
15	1SV18CS015	GAGANA N
16	1SV18CS017	GANYA KUMAR G R
17	1SV18CS019	HADA AMAL KHAN
18	1SV18CS021	KEERTHIPRASAD B K
19	1SV18CS022	KUSHAL KUMAR D
20	1SV18CS023	LAVANYA T A
21	1SV18CS024	LISHA SHREE NAYAKA S
22	1SV18CS025	MANORANJAN P M
23	1SV18CS026	MURFUA FATHIMA
24	1SV18CS028	MEGHANA G S
25	1SV18CS029	NANDA T M
26	1SV18CS030	PAVAN KUMAR DURGAD
27	1SV18CS031	PRAGNA H S
28	1SV18CS032	PRAJWAL C
29	1SV18CS033	PRIYADARSHINI R
30	1SV18CS038	SHRADDHA S
31	1SV18CS042	SUSHMA H S
32	1SV18CS043	THUNGASHREE
33	1SV18CS045	VIJAYALAXMI
34	1SV18CS046	VIVEKANAND MATH
35	1SV19CS400	SHIREESHA HEGDE H R
36	1SV19CS401	VEENA L G
37	1SV17CS005	ANIL GOWDA
38	1SV17CS008	B RAMESH
39	1SV15CS074	RAJEEVA N P
40	1SV15CS088	SHIVAYOGI B S


(Prof. C V Shanmuka Swamy)
HOD, CSE


PRINCIPAL
SIET, TUMKURU.

Module – 1 :Storage System

Introduction

Information is increasingly important in our daily lives. We have become information dependents of the twenty-first century, living in an on-command, on-demand world that means we need information when and where it is required. We access the Internet every day to perform searches, participate in social networking, send and receive e-mails, share pictures and videos, and scores of other applications. Equipped with a growing number of content-generating devices, more information is being created by individuals than by businesses.

Information created by individuals gains value when shared with others. Figure 1-1 depicts this virtuous cycle of information.

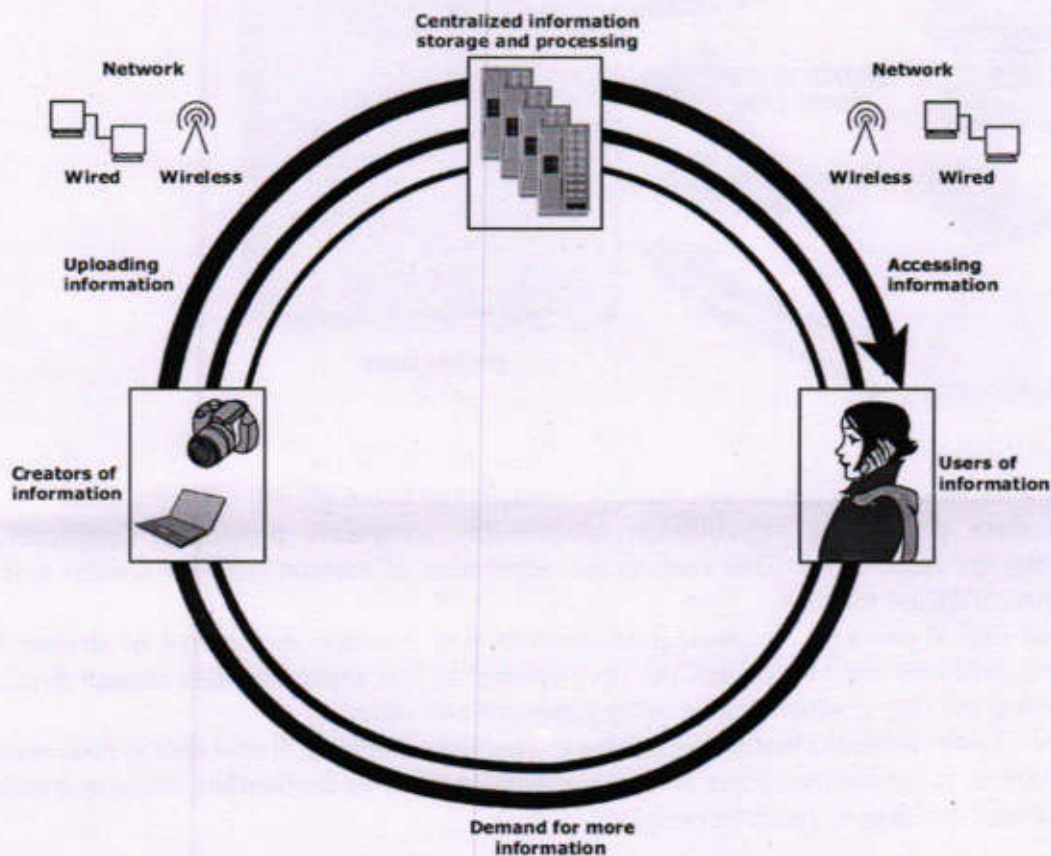


Figure 1-1: Virtuous cycle of information

Chapter Objective

This chapter describes the evolution of information storage architecture from simple direct-attached models to complex networked topologies. It introduces the information lifecycle management (ILM) strategy, which aligns the information technology (IT) infrastructure with business priorities.

1.1 Information Storage

Businesses use data to derive information that is critical to their day-to-day operations. Storage is a repository that enables users to store and retrieve this digital data.

1.1.1 Data

Data is a collection of raw facts from which conclusions may be drawn. Handwritten letters, a printed book, a family photograph, a movie on video tape, printed and duly signed copies of mortgage papers, a bank's ledgers, and an account holder's passbooks are all examples of data.

The data can be generated using a computer and stored in strings of 0s and 1s, as shown in Figure 1-2. Data in this form is called *digital data* and is accessible by the user only after it is processed by a computer.

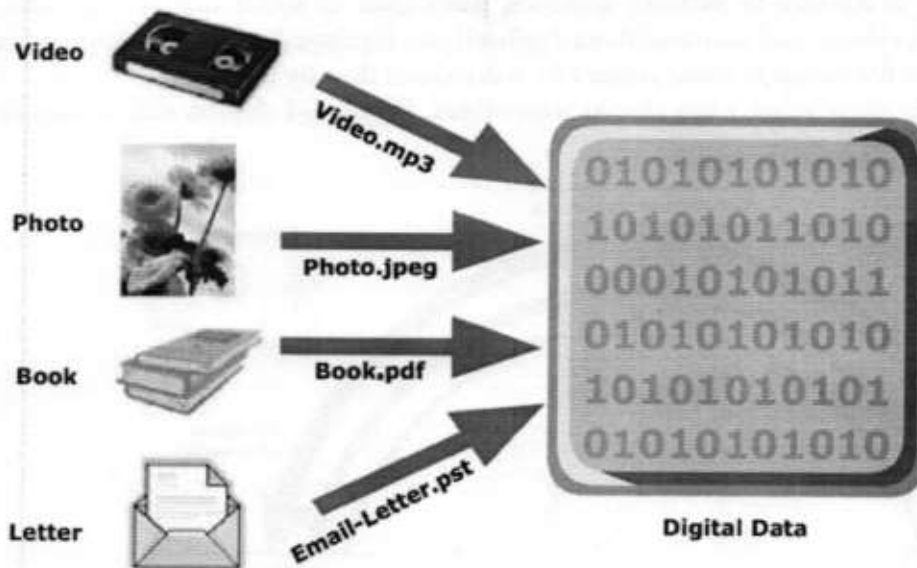


Figure 1-2: Digital data

The following is a list of some of the factors that have contributed to the growth of digital data:

- 1. Increase in data processing capabilities:** Modern-day computers provide a significant increase in processing and storage capabilities. This enables the conversion of various types of content and media from conventional forms to digital formats.
- 2. Lower cost of digital storage:** Technological advances and decrease in the cost of storage devices have provided low-cost solutions and encouraged the development of less expensive data storage devices. This cost benefit has increased the rate at which data is being generated and stored.
- 3. Affordable and faster communication technology:** The rate of sharing digital data is now much faster than traditional approaches. A handwritten letter may take a week to reach its destination, whereas it only takes a few seconds for an e-mail message to reach its recipient.

The importance and the criticality of data vary with time. Most of the data created holds significance in the short-term but becomes less valuable over time. This governs the type of data storage solutions used. Individuals store data on a variety of storage devices, such as hard disks, CDs, DVDs, or Universal Serial Bus (USB) flash drives.

1.1.2 Types of Data

Data can be classified as structured or unstructured (see Figure 1-3) based on how it is stored and managed. Structured data is organized in rows and columns in a rigidly defined format so that applications can retrieve and process it efficiently. Structured data is typically stored using a database management system (DBMS). Data is unstructured if its elements cannot be stored in rows and columns, and is therefore difficult to query and retrieve by business applications. For example, customer contacts may be stored in various forms such as sticky notes, e-mail messages, business cards, or even digital format files such as .doc, .txt, and .pdf.

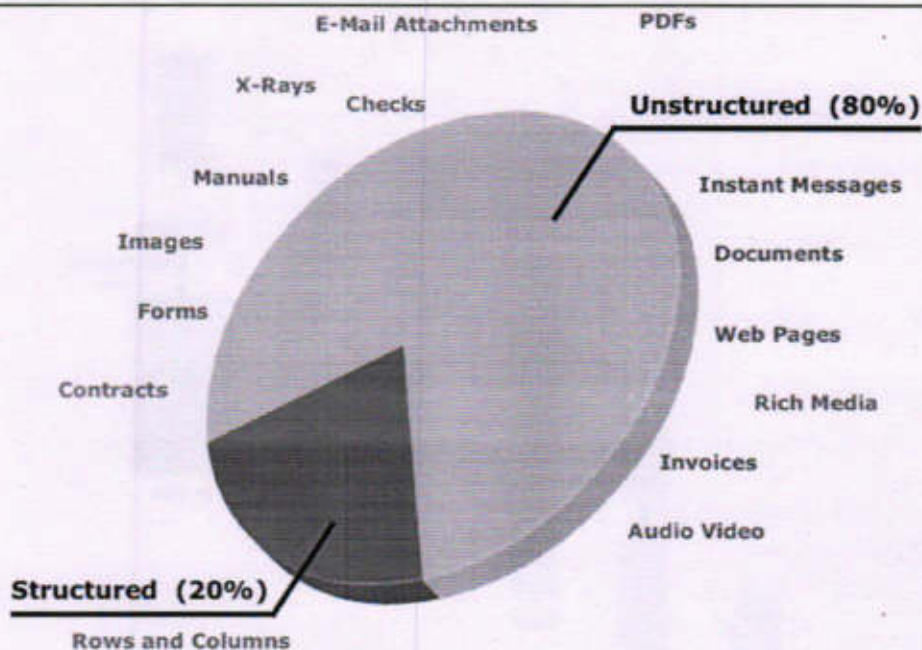


Figure 1-3: Types of data

1.1.3 Information

Information is the intelligence and knowledge derived from data. Data, whether structured or unstructured, does not fulfill any purpose for individuals or businesses unless it is presented in a meaningful form. Businesses need to analyze data for it to be of value.

Effective data analysis not only extends its benefits to existing businesses, but also creates the potential for new business opportunities by using the information in creative ways.

Example: Job portal. In order to reach a wider set of prospective employers, job seekers post their résumés on various websites offering job search facilities. These websites collect the résumés and post them on centrally accessible locations for prospective employers. In addition, companies post available positions on job search sites. Job-matching software matches keywords from résumés to keywords in job postings. In this manner, the job search engine uses data and turns it into information for employers and job seekers.

1.1.4 Storage

Data created by individuals or businesses must be stored so that it is easily accessible for further processing. In a computing environment, devices designed for storing data are termed *storage devices* or simply *storage*.

Devices such as memory in a cell phone or digital camera, DVDs, CD-ROMs, and hard disks in personal computers are examples.

1.2 Evolution of Storage Technology and Architecture

Historically, organizations had centralized computers (mainframe) and information storage devices (tape reels and disk packs) in their data center. The evolution of open systems and the affordability and ease of deployment that they offer made it possible for business units/departments to have their own servers and storage. In earlier implementations of open systems, the storage was typically internal to the server.

Originally, there were very limited policies and processes for managing the servers and the data created. To overcome these challenges, storage technology evolved from non-intelligent internal storage to intelligent networked storage (see Figure 1-4).

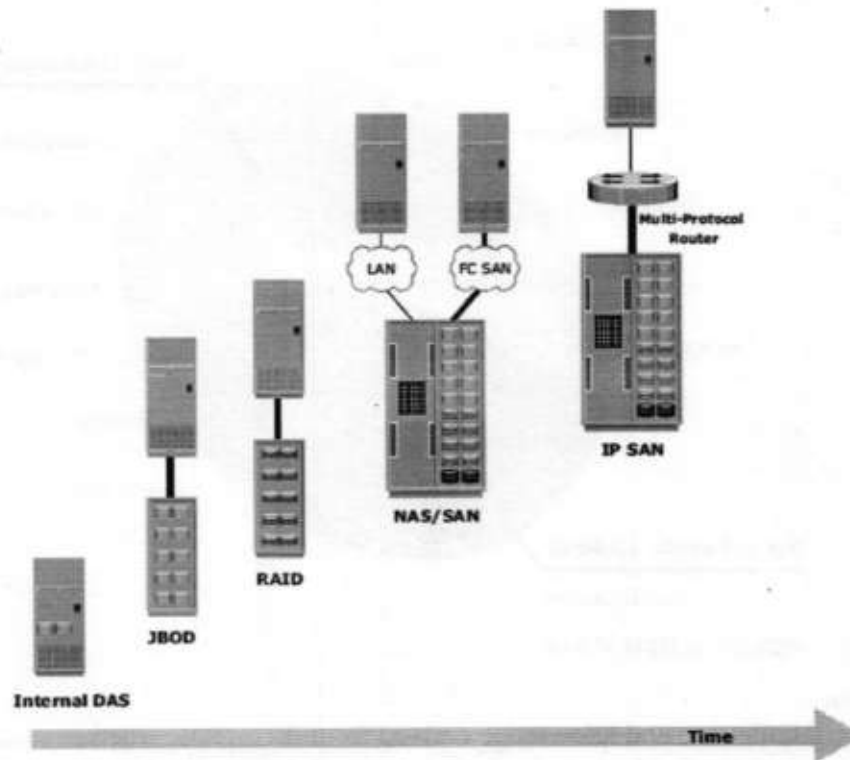


Figure 1-4: Evolution of storage architectures

The technology evolution includes:

1. **Redundant Array of Independent Disks (RAID):** This technology was developed to address the cost, performance, and availability requirements of data. It continues to evolve today and is used in all storage architectures such as DAS, SAN, and so on.
2. **Direct-attached storage (DAS):** This type of storage connects directly to a server (host) or a group of servers in a cluster. Storage can be either internal or external to the server. External DAS alleviated the challenges of limited internal storage capacity.
3. **Storage area network (SAN):** This is a dedicated, high-performance *Fibre Channel (FC)* network to facilitate *block-level* communication between servers and storage. Storage is partitioned and assigned to a server for accessing its data. SAN offers scalability, availability, performance, and cost benefits compared to DAS.
4. **Network-attached storage (NAS):** This is dedicated storage for *file serving* applications. Unlike a SAN, it connects to an existing communication network (LAN) and provides file access to heterogeneous clients. Because it is purposely built for providing storage to file server applications, it offers higher scalability, availability, performance, and cost benefits compared to general purpose file servers.
5. **Internet Protocol SAN (IP-SAN):** One of the latest evolutions in storage architecture, IP-SAN is a convergence of technologies used in SAN and NAS. IP-SAN provides block-level communication across a local or wide area network (LAN or WAN), resulting in greater consolidation and availability of data.

1.3 Data Center Infrastructure

Organizations maintain data centers to provide centralized data processing capabilities across the enterprise. Data centers store and manage large amounts of mission-critical data.

The data center infrastructure includes 1) computers, 2) storage systems, 3) network devices, 4) dedicated power backups, 5) and environmental controls (such as air conditioning and fire suppression).

1.3.1 Core Elements

Five core elements are essential for the basic functionality of a data center:

1. **Application:** An application is a computer program that provides the logic for computing operations. Applications, such as an order processing system, can be layered on a database, which in turn uses operating system services to perform read/write operations to storage devices.
2. **Database:** More commonly, a database management system (DBMS) provides a structured way to store data in logically organized tables that are interrelated. A DBMS optimizes the storage and retrieval of data.
3. **Server and operating system:** A computing platform that runs applications and databases.
4. **Network:** A data path that facilitates communication between clients and servers or between servers and storage.
5. **Storage array:** A device that stores data persistently for subsequent use.

Example: Figure 1-5 shows an order processing system that involves the five core elements of a data center and illustrates their functionality in a business process.

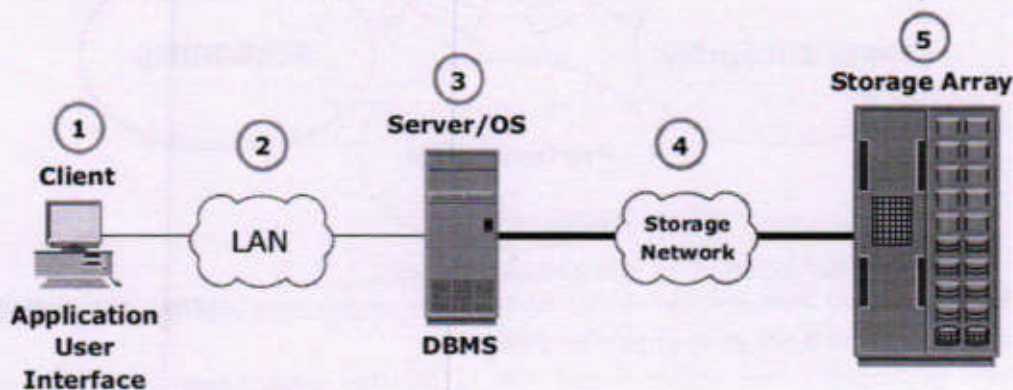


Figure 1-5: Example of an order processing system

Step 1: A customer places an order through the AUI of the order processing application software located on the client computer.

Step 2: The client connects to the server over the LAN and accesses the DBMS located on the server to update the relevant information such as the customer name, address, payment method, products ordered, and quantity ordered.

Step 3: The DBMS uses the server operating system to read and write this data to the database located on physical disks in the storage array.

Step 4: The Storage Network provides the communication link between the server and the storage array and transports the read or write commands between them.

Step 5: The storage array, after receiving the read or write commands from the server, performs the necessary operations to store the data on physical disks.

1.3.2 Key Requirements for Data Center Elements

Uninterrupted operation of data centers is critical to the survival and success of a business. It is necessary to have a reliable infrastructure that ensures data is accessible at all times. While the requirements, shown in Figure 1-6, are applicable to all elements of the data center infrastructure, our focus here is on storage systems.

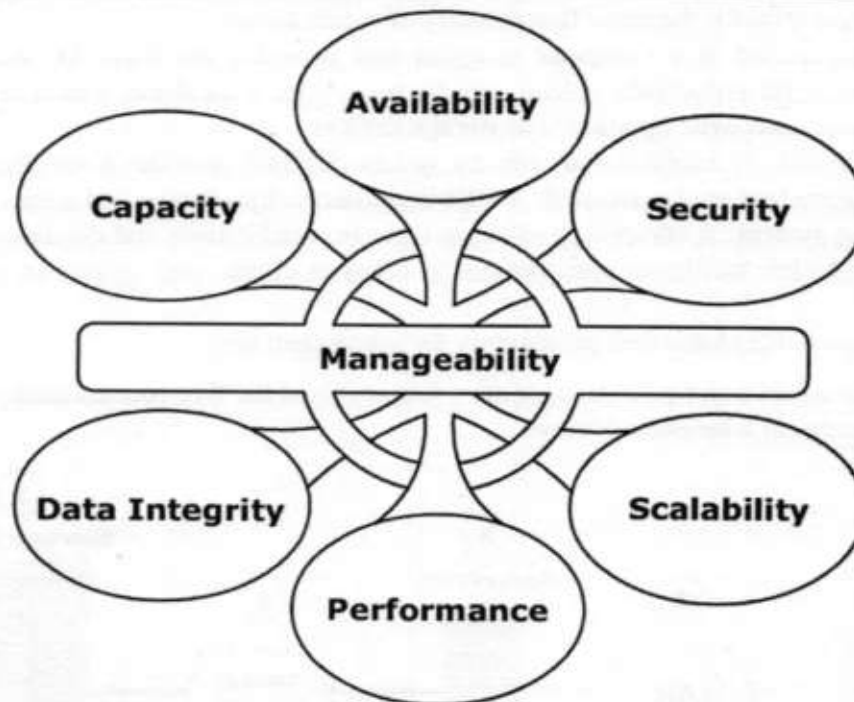


Figure 1-6: Key characteristics of data center elements

1 Availability: All data center elements should be designed to ensure accessibility. The inability of users to access data can have a significant negative impact on a business.

2 Security: Policies, procedures, and proper integration of the data center core elements that will prevent unauthorized access to information must be established. In addition to the security measures for client access, specific mechanisms must enable servers to access only their allocated resources on storage arrays.

3 Scalability: Data center operations should be able to allocate additional processing capabilities or storage on demand, without interrupting business operations. Business growth often requires deploying more servers, new applications, and additional databases. The storage solution should be able to grow with the business.

4 Performance: All the core elements of the data center should be able to provide optimal performance and service all processing requests at high speed. The infrastructure should be able to support performance requirements.

5 Data integrity: Data integrity refers to mechanisms such as error correction codes or parity bits which ensure that data is written to disk exactly as it was received. Any variation in data during its retrieval implies corruption, which may affect the operations of the organization.

6 Capacity: Data center operations require adequate resources to store and process large amounts of data efficiently. When capacity requirements increase, the data center must be able to provide additional capacity without interrupting availability, or, at the very least, with minimal disruption.

Capacity may be managed by reallocation of existing resources, rather than by adding new resources.

7 Manageability: A data center should perform all operations and activities in the most efficient manner. Manageability can be achieved through automation and the reduction of human (manual) intervention in common tasks.

1.3.3 Managing Storage Infrastructure

Managing a modern, complex data center involves many tasks. Key management activities include:

1 Monitoring is the continuous collection of information and the review of the entire data center infrastructure. The aspects of a data center that are monitored include security, performance, accessibility, and capacity.

2 Reporting is done periodically on resource performance, capacity, and utilization. Reporting tasks help to establish business justifications and chargeback of costs associated with data center operations.

3 Provisioning is the process of providing the hardware, software, and other resources needed to run a data center. Provisioning activities include capacity and resource planning. **Capacity planning** ensures that the user's and the application's future needs will be addressed in the most cost-effective and controlled manner. **Resource planning** is the process of evaluating and identifying required resources, such as personnel, the facility (site), and the technology.

1.4 Key Challenges in Managing Information

In order to frame an effective information management policy, businesses need to consider the following key challenges of information management:

1 Exploding digital universe: The rate of information growth is increasing exponentially. Duplication of data to ensure high availability and repurposing has also contributed to the multifold increase of information growth.

2 Increasing dependency on information: The strategic use of information plays an important role in determining the success of a business and provides competitive advantages in the marketplace.

3 Changing value of information: Information that is valuable today may become less important tomorrow. The value of information often changes over time.

1.5 Information Lifecycle

The *information lifecycle* is the "change in the value of information" over time. When data is first created, it often has the highest value and is used frequently. As data ages, it is accessed less frequently and is of less value to the organization.

For example, in a sales order application, the value of the information changes from the time the order is placed until the time that the warranty becomes void (see Figure 1-7). The value of the information is highest when a company receives a new sales order and processes it to deliver the product.

After order fulfillment, the customer or order data need not be available for real-time access. The company can transfer this data to less expensive secondary storage with lower accessibility and availability requirements unless or until a warranty claim or another event triggers its need.

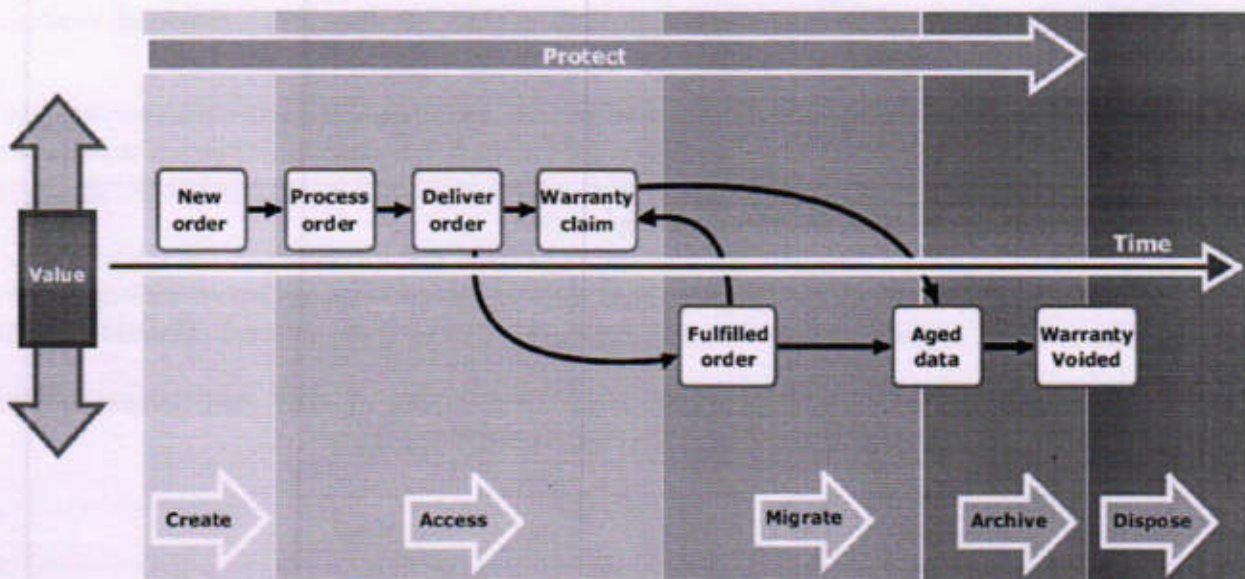


Figure 1-7: Changing value of sales order information

1.5.1 Information Lifecycle Management

Today's business requires data to be protected and available 24 × 7. Data centers can accomplish this with the optimal and appropriate use of storage infrastructure.

Information lifecycle management (ILM) is a proactive strategy that enables an IT organization to effectively manage the data throughout its lifecycle, based on predefined business policies. This allows an IT organization to optimize the storage infrastructure for maximum return on investment.

An ILM strategy should include the following characteristics:

1 Business-centric: It should be integrated with key processes, applications, and initiatives of the business to meet both current and future growth in information.

2 Centrally managed: All the information assets of a business should be under the purview of the ILM strategy.

3 Policy-based: The implementation of ILM should not be restricted to a few departments. ILM should be implemented as a policy and encompass all business applications, processes, and resources.

4 Heterogeneous: An ILM strategy should take into account all types of storage platforms and operating systems.

5 Optimized: Because the value of information varies, an ILM strategy should consider the different storage requirements and allocate storage resources based on the information's value to the business.

1.5.2 ILM Implementation

The process of developing an ILM strategy includes four activities—classifying, implementing, managing, and organizing:

1 Classifying data and applications on the basis of business rules and policies to enable differentiated treatment of information

2 Implementing policies by using information management tools, starting from the creation of data and ending with its disposal

3 Managing the environment by using integrated tools to reduce operational complexity

4 Organizing storage resources in tiers to align the resources with data classes, and storing information in the right type of infrastructure based on the information's current value.

Implementing ILM across an enterprise is an ongoing process. Figure 1-8 illustrates a three-step road map to enterprise-wide ILM.

Step 1 the goal is to implement a storage networking environment. Storage architectures offer varying levels of protection and performance and this acts as a foundation for future policy-based information management in Steps 2 and 3. The value of tiered storage platforms can be exploited by allocating appropriate storage resources to the applications based on the value of the information processed.

Step 2 takes ILM to the next level, with detailed application or data classification and linkage of the storage infrastructure to business policies. These classifications and the resultant policies can be automatically executed using tools for one or more applications, resulting in better management and optimal allocation of storage resources.

Step 3 of the implementation is to automate more of the applications or data classification and policy management activities in order to scale to a wider set of enterprise applications.

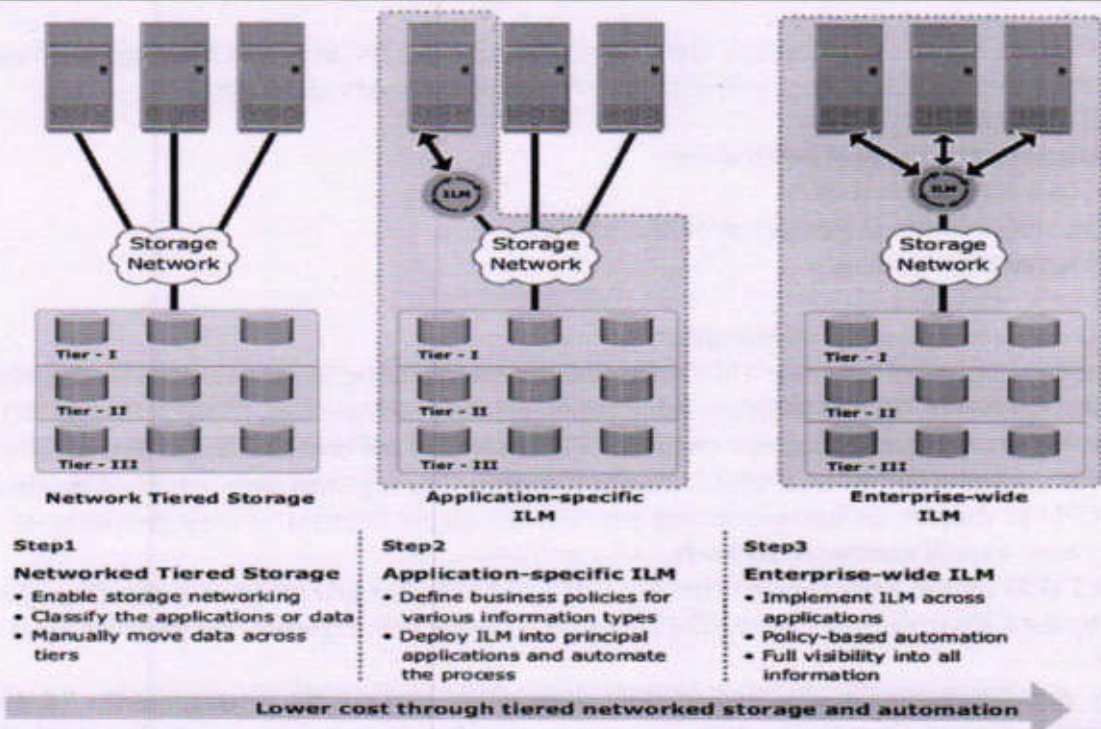


Figure 1-8: Implementation of ILM

1.5.3 ILM Benefits

Implementing an ILM strategy has the following key benefits that directly address the challenges of information management:

- 1 **Improved utilization** by using tiered storage platforms and increased visibility of all enterprise information.
- 2 **Simplified management** by integrating process steps and interfaces with individual tools and by increasing automation.
- 3 **A wider range of options** for backup, and recovery to balance the need for business continuity.
- 4 **Maintaining compliance** by knowing what data needs to be protected for what length of time.
- 5 **Lower Total Cost of Ownership (TCO)** by aligning the infrastructure and management costs with information value. As a result, resources are not wasted, and complexity is not introduced by managing low-value data at the expense of high-value data.

2 Storage System Environment

Storage, as one of the core elements of a data center, is recognized as a distinct resource and it needs focus and specialization for its implementation and management. The data flows from an application to storage through various components collectively referred as a *storage system environment*. The three main components in this environment are the host, connectivity, and storage. These entities, along with their physical and logical components, facilitate data access.

Chapter Objective

This chapter details the storage system environment and focuses primarily on storage. It provides details on various hardware components of a disk drive, disk geometry, and the fundamental laws that govern disk performance. The connectivity between the host and storage facilitated by bus technology and interface protocols is also explained.

2.1 Components of a Storage System Environment

2.1.1 Host

Users store and retrieve data through applications. The computers on which these applications run are referred to as *hosts*. Hosts can range from simple laptops to complex clusters of servers.

Physical Components

A host has three key physical components:

- Central processing unit (CPU)
- Storage, such as internal memory and disk devices
- Input/Output (I/O) devices

CPU

The CPU consists of four main components:

- **Arithmetic Logic Unit (ALU):** This is the fundamental building block of the CPU. It performs arithmetical and logical operations such as addition, subtraction, and Boolean functions (AND, OR, and NOT).
- **Control Unit:** A digital circuit that controls CPU operations and coordinates the functionality of the CPU.
- **Register:** A collection of high-speed storage locations. The registers store intermediate data that is required by the CPU to execute an instruction and provide fast access because of their proximity to the ALU. CPUs typically have a small number of registers.
- **Level 1 (L1) cache:** Found on modern day CPUs, it holds data and program instructions that are likely to be needed by the CPU in the near future. The L1 cache is slower than registers, but provides more storage space.

Storage

Memory and storage media are used to store data, either persistently or temporarily. Memory modules are implemented using semiconductor chips, whereas storage devices use either magnetic or optical media.

There are two types of memory on a host:

- **Random Access Memory (RAM):** This allows direct access to any memory location and can have data written into it or read from it. RAM is volatile; this type of memory requires a constant supply of power to maintain memory cell content. Data is erased when the system's power is turned off or interrupted.
- **Read-Only Memory (ROM):** Non-volatile and only allows data to be read from it. ROM holds data for execution of internal routines, such as system startup.

I/O Devices

I/O devices enable sending and receiving data to and from a host. This communication may be one of the following types:

- **User to host communications:** Handled by basic I/O devices, such as the keyboard, mouse, and monitor. These devices enable users to enter data and view the results of operations.
- **Host to host communications:** Enabled using devices such as a Network Interface Card (NIC) or modem.
- **Host to storage device communications:** Handled by a *Host Bus Adaptor (HBA)*. HBA is an application-specific integrated circuit (ASIC) board that performs I/O interface functions between the host and the storage, relieving the CPU from additional I/O processing workload.

2.1.2 Connectivity

Connectivity refers to the interconnection between hosts or between a host and any other peripheral devices, such as printers or storage devices. The components of connectivity in a storage system environment can be classified as physical and logical. The *physical components* are the hardware elements that connect the host to storage and the *logical components* of connectivity are the protocols used for communication between the host and storage.

Physical Components of Connectivity

The three physical components of connectivity between the host and storage are Bus, Port, and Cable.

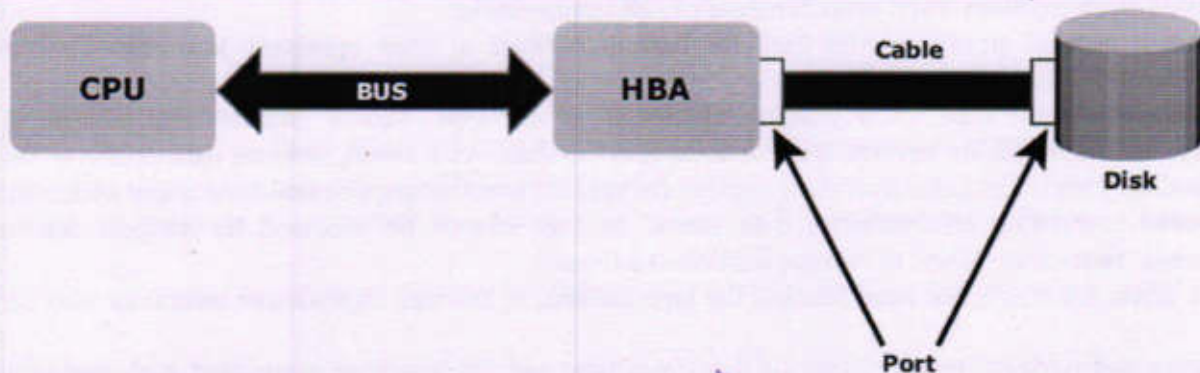


Figure 2-1: Physical components of connectivity

1. The *bus* is the collection of paths that facilitates data transmission from one part of a computer to another, such as from the CPU to the memory.
2. The *port* is a specialized outlet that enables connectivity between the host and external devices.
3. *Cables* connect hosts to internal or external devices using copper or fiber optic media.

Physical components communicate across a bus by sending bits (control, data, and address) of data between devices.

These bits are transmitted through the bus in either of the following ways:

- **Serially:** Bits are transmitted sequentially along a single path. This transmission can be unidirectional or bidirectional.
- **In parallel:** Bits are transmitted along multiple paths simultaneously. Parallel can also be bidirectional.

Bus, as conduits of data transfer on the computer system, can be classified as follows:

- **System bus:** The bus that carries data from the processor to memory.
- **Local or I/O bus:** A high-speed pathway that connects directly to the processor and carries data between the peripheral devices, such as storage devices and the processor.

Logical Components of Connectivity

PCI
PCI is a specification that standardizes how PCI expansion cards, such as network cards or modems, exchange information with the CPU. PCI provides the interconnection between the CPU and attached devices. The plug-and-play functionality of PCI enables the host to easily recognize and configure new cards and devices. The width of a PCI bus can be 32 bits or 64 bits. A 32-bit PCI bus can provide a throughput of 133 MB/s. *PCI Express* is an enhanced version of PCI bus with considerably higher throughput and clock speed.

IDE/ATA

IDE/ATA is the most popular interface protocol used on modern disks. This protocol offers excellent performance at relatively low cost. Details of IDE/ATA are provided in Chapter 5.

SCSI

SCSI has emerged as a preferred protocol in high-end computers. This interface is far less commonly used than IDE/ATA on personal computers due to its higher cost. SCSI was initially used as a parallel interface, enabling the connection of devices to a host. SCSI has been enhanced and now includes a wide variety of related technologies and standards. Chapter 5 provides details of SCSI.

2.1.3 Storage

A storage device uses magnetic or solid state media.

1. **Disks, tapes, and diskettes** use magnetic media.
2. **CD-ROM** is an example of a storage device that uses optical media

3. **Removable flash memory card** is an example of solid state media.

4. **Tapes** are a popular storage media used for backup because of their relatively low cost. Tape has the following limitations:

- Data is stored on the tape linearly along the length of the tape. Search and retrieval of data is done sequentially, invariably taking several seconds to access the data. As a result, random data access is slow and time consuming. This limits tapes as a viable option for applications that require real-time, rapid access to data.

- In a shared computing environment, data stored on tape cannot be accessed by multiple applications simultaneously, restricting its use to one application at a time.

- On a tape drive, the read/write head touches the tape surface, so the tape degrades or wears out after repeated use.

- The storage and retrieval requirements of data from tape and the overhead associated with managing tape media are significant.

5. **Optical disk storage** is popular in small, single-user computing environments. It is frequently used by individuals to store photos or as a backup medium on personal/laptop computers. It is also used as a distribution medium for single applications, such as games, or as a means of transferring small amounts of data from one self-contained system to another. Optical disks have limited capacity and speed, which limits the use of optical media as a business data storage solution.

2.2 Disk Drive Components

A disk drive uses a rapidly moving arm to read and write data across a flat platter coated with magnetic particles. Data is transferred from the magnetic platter through the R/W head to the computer. Several platters are assembled together with the R/W head and controller, most commonly referred to as a *hard disk drive (HDD)*. Data can be recorded and erased on a magnetic disk any number of times.

Key components of a disk drive are *platter, spindle, read/write head, actuator arm assembly, and controller* (Figure 2-2)

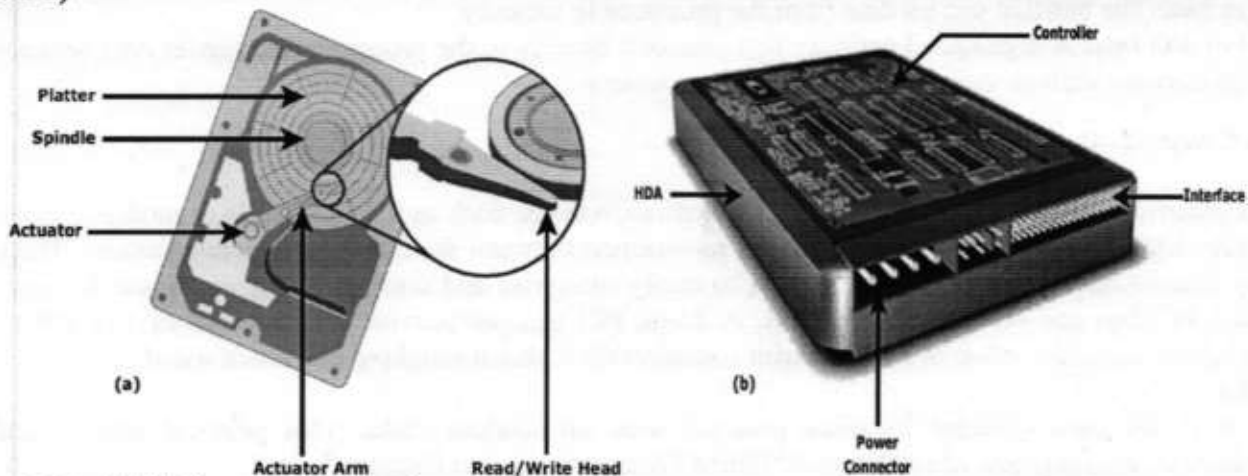


Figure 2-2: Disk Drive Components

2.2.1 Platter

A typical HDD consists of one or more flat circular disks called *platters* (Figure 2-3). The data is recorded on these platters in binary codes (0s and 1s). The set of rotating platters is sealed in a case, called a *Head Disk Assembly (HDA)*. The data is encoded by polarizing the magnetic area, or domains, of the disk surface. Data can be written to or read from both surfaces of the platter.

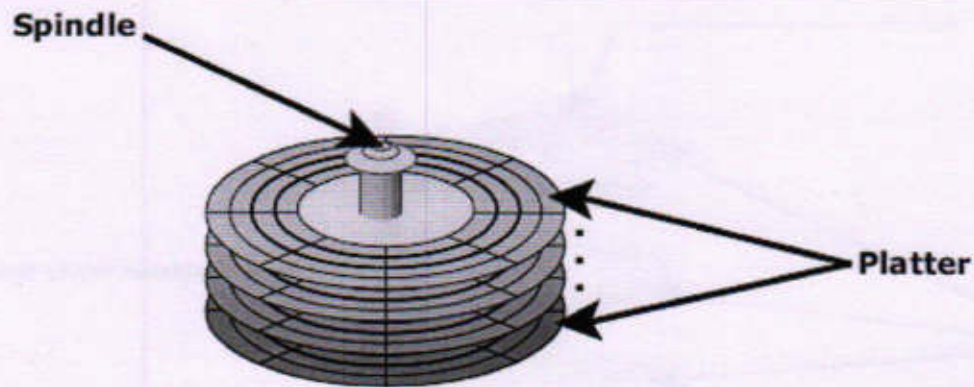


Figure 2-3: Spindle and platter

2.2.2 Spindle

A spindle connects all the platters, as shown in Figure 2-3, and is connected to a motor. The motor of the spindle rotates with a constant speed. The disk platter spins at a speed of several thousands of revolutions per minute (rpm). Disk drives have spindle speeds of 7,200 rpm, 10,000 rpm, or 15,000 rpm. Disks used on current storage systems have a platter diameter of 3.5" (90 mm).

2.2.3 Read/Write Head

Read/Write (R/W) heads, shown in Figure 2-4, read and write data from or to a platter. Drives have two R/W heads per platter, one for each surface of the platter. The R/W head changes the magnetic polarization on the surface of the platter when writing data. While reading data, this head detects magnetic polarization on the surface of the platter. During reads and writes, the R/W head senses the magnetic polarization and never touches the surface of the platter.

The logic on the disk drive ensures that heads are moved to the landing zone before they touch the surface. If the drive malfunctions and the R/W head accidentally touches the surface of the platter outside the landing zone, a *head crash* occurs. In a head crash, the magnetic coating on the platter is scratched and may cause damage to the R/W head. A head crash generally results in data loss.

2.2.4 Actuator Arm Assembly

The R/W heads are mounted on the *actuator arm assembly* (refer to Figure 2-2 [a]), which positions the R/W head at the location on the platter where the data needs to be written or read. The R/W heads for all platters on a drive are attached to one actuator arm assembly and move across the platters simultaneously.

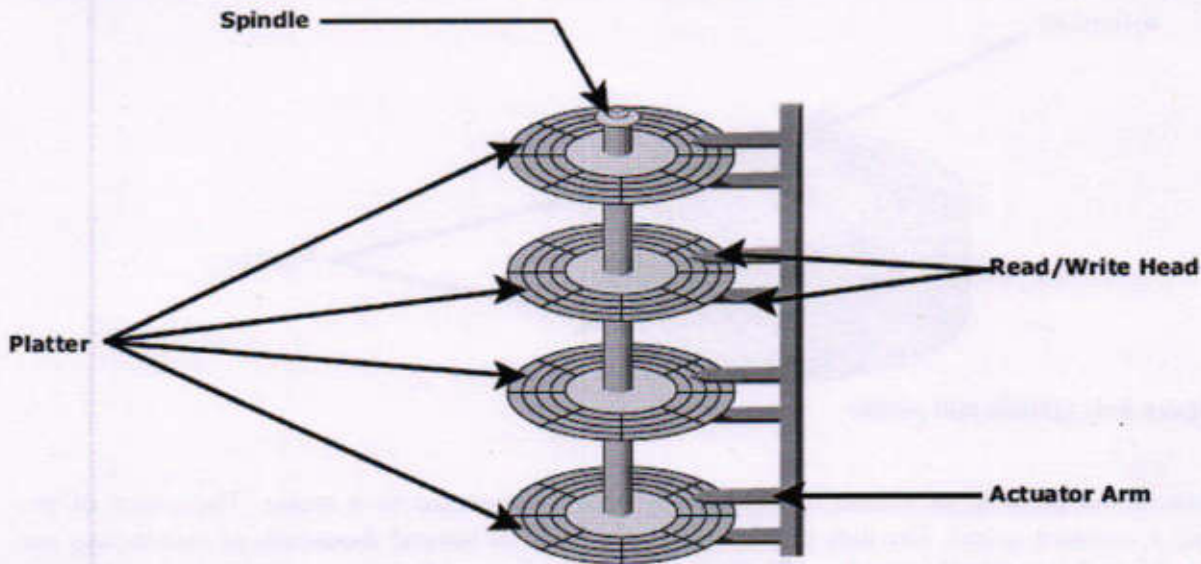


Figure 2-4: Actuator arm assembly

2.2.5 Controller

The *controller* (see Figure 2-2 [b]) is a printed circuit board, mounted at the bottom of a disk drive. It consists of a microprocessor, internal memory, circuitry, and firmware. The firmware controls power to the spindle motor and the speed of the motor. It also manages communication between the drive and the host.

2.2.6 Physical Disk Structure

Data on the disk is recorded on *tracks*, which are concentric rings on the platter around the spindle, as shown in Figure 2-5. The tracks are numbered, starting from zero, from the outer edge of the platter. The number of *tracks per inch (TPI)* on the platter (or the *track density*) measures how tightly the tracks are packed on a platter. Each track is divided into smaller units called *sectors*. A sector is the smallest, individually addressable unit of storage.

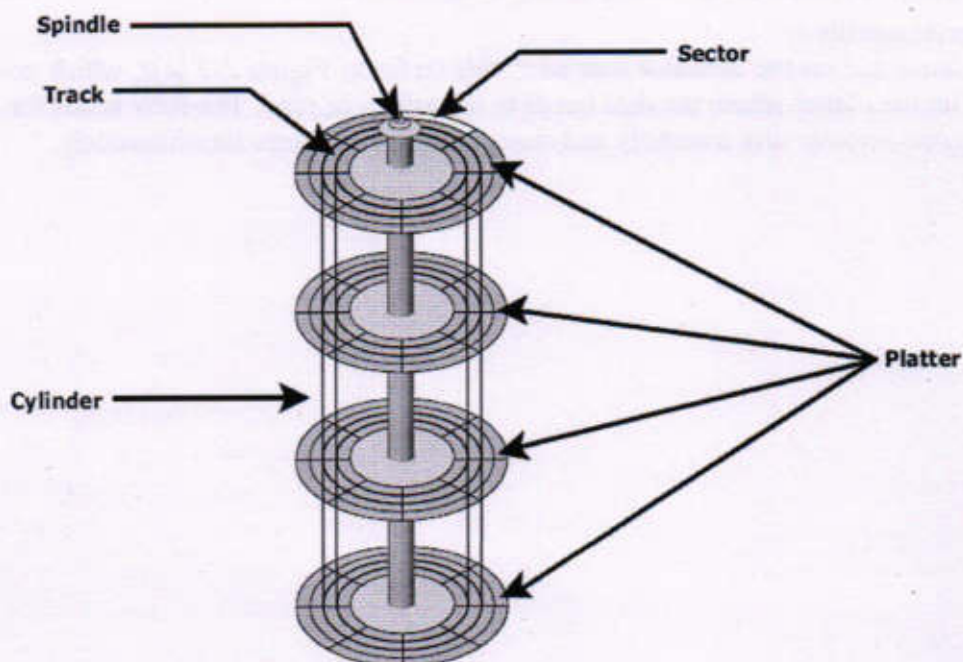


Figure 2-5: Disk structure: sectors, tracks, and cylinders

2.2.7 Zoned Bit Recording

Zone bit recording utilizes the disk efficiently. As shown in Figure 2-6 (b), this mechanism groups tracks into zones based on their distance from the center of the disk. The zones are numbered, with the outermost zone being zone 0. An appropriate number of sectors per track are assigned to each zone, so a zone near the center of the platter has fewer sectors per track than a zone on the outer edge.

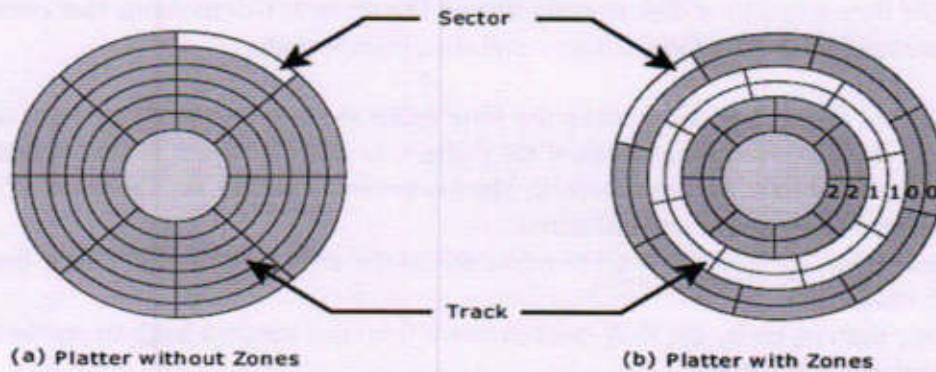


Figure 2-6: Zoned bit recording

2.2.8 Logical Block Addressing

Earlier drives used physical addresses consisting of the *cylinder, head, and sector (CHS)* number to refer to specific locations on the disk, as shown in Figure 2-7 (a), and the host operating system had to be aware of the geometry of each disk being used. *Logical block addressing (LBA)*, shown in Figure 2-7 (b), simplifies addressing by using a linear address to access physical blocks of data. The disk controller translates LBA to a CHS address, and the host only needs to know the size of the disk drive in terms of the number of blocks.

In Figure 2-7 (b), the drive shows eight sectors per track, eight heads, and four cylinders. This means a total of $8 \times 8 \times 4 = 256$ blocks, so the block number ranges from 0 to 255. Each block has its own unique address. Assuming that the sector holds 512 bytes, a 500 GB drive with a formatted capacity of 465.7 GB will have in excess of 976,000,000 blocks.

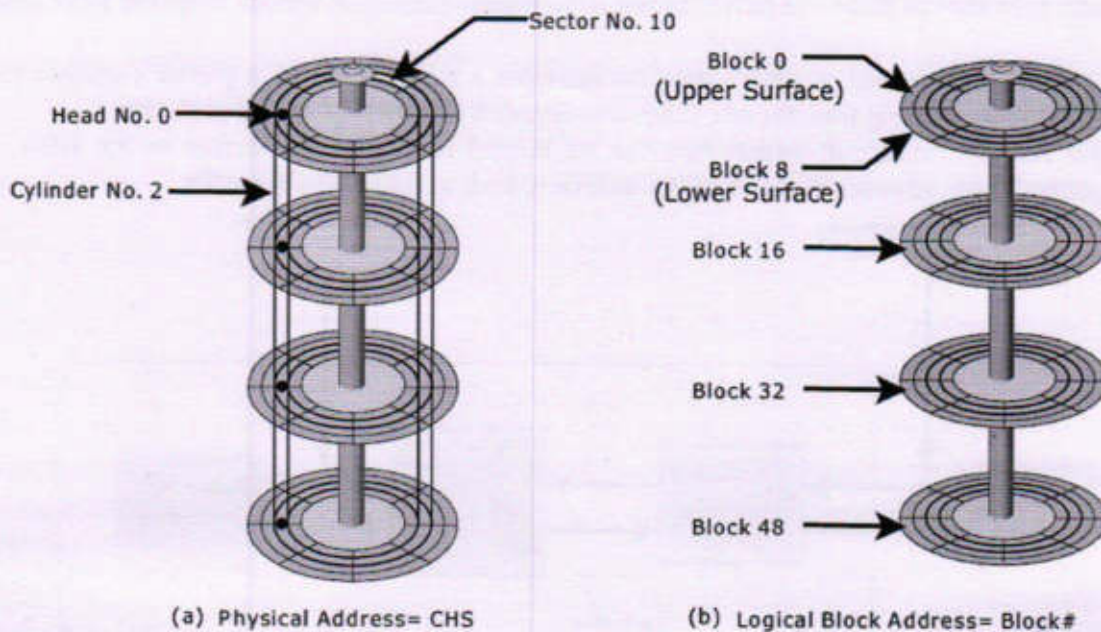


Figure 2-7: Physical address and logical block address

2.3 Disk Drive Performance

A disk drive is an electromechanical device that governs the overall performance of the storage system environment.

The various factors that affect the performance of disk drives are discussed in this section.

2.3.1 Disk Service Time

Disk service time is the time taken by a disk to complete an I/O request. Components that contribute to service time on a disk drive are *seek time*, *rotational latency*, and *data transfer rate*.

1. Seek Time

The *seek time* (also called *access time*) describes the time taken to position the R/W heads across the platter with a radial movement (moving along the radius of the platter). In other words, it is the time taken to reposition and settle the arm and the head over the correct track. The lower the seek time, the faster the I/O operation. Disk vendors publish the following seek time specifications:

■ **Full Stroke:** The time taken by the R/W head to move across the entire width of the disk, from the innermost track to the outermost track.

■ **Average:** The average time taken by the R/W head to move from one random track to another, normally listed as the time for one-third of a full stroke.

■ **Track-to-Track:** The time taken by the R/W head to move between adjacent tracks.

2. Rotational Latency

To access data, the actuator arm moves the R/W head over the platter to a particular track while the platter spins to position the requested sector under the R/W head. The time taken by the platter to rotate and position the data under the R/W head is called *rotational latency*. This latency depends on the rotation speed of the spindle and is measured in milliseconds.

3. Data Transfer Rate

The *data transfer rate* (also called *transfer rate*) refers to the average amount of data per unit time that the drive can deliver to the HBA. In a *read operation*, the data first moves from disk platters to R/W heads, and then it moves to the drive's internal *buffer*. Finally, data moves from the buffer through the interface to the host HBA. In a *write operation*, the data moves from the HBA to the internal buffer of the disk drive through the drive's interface. The data then moves from the buffer to the R/W heads. Finally, it moves from the R/W heads to the platters.

Internal transfer rate is the speed at which data moves from a single track of a platter's surface to internal buffer (cache) of the disk. Internal transfer rate takes into account factors such as the seek time.

External transfer rate is the rate at which data can be moved through the interface to the HBA. External transfer rate is generally the advertised speed of the interface, such as 133 MB/s for ATA.

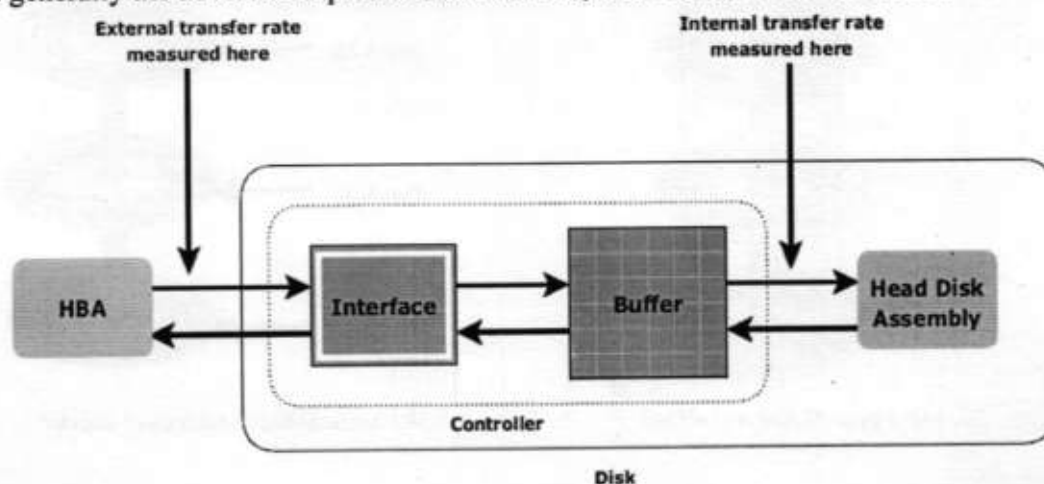


Figure 2-8: Data transfer rate

2.4 Fundamental Laws Governing Disk Performance

To understand the laws of disk performance, a disk can be viewed as a black box consisting of two elements:

■ **Queue:** The location where I/O request waits before it is processed by the I/O controller.

■ **Disk I/O Controller:** Processes I/Os that are waiting in the queue one by one.

The I/O requests arrive at the controller at the rate generated by the application. This rate is also called the *arrival rate*. These requests are held in the I/O queue, and the I/O controller processes them one by one, as shown in Figure 2-9. The I/O arrival rate, the queue length, and the time taken by the I/O controller to process each request determines the performance of the disk system, which is measured in terms of response time.



Figure 2-9: I/O processing

Little's Law is a fundamental law describing the relationship between the number of requests in a queue and the response time. The law states the following relation (numbers in parentheses indicate the equation number for crossreferencing):

$$N = a \times R \text{ ----- (1)}$$

where

“N” is the total number of requests in the queuing system (requests in the queue + requests in the I/O controller)

“a” is the arrival rate, or the number of I/O requests that arrive to the system per unit of time

“R” is the average response time or the turnaround time for an I/O request — the total time from arrival to departure from the system

The **utilization law** is another important law that defines the I/O controller utilization. This law states the relation:

$$U = a \times R_s \text{ ----- (2)}$$

where

“U” is the I/O controller utilization

“R_s” is the *service time*, or the average time spent by a request on the controller. 1/R_s is the *service rate*.

From the arrival rate “a”, the average inter-arrival time, R_a, can be computed as:

$$R_a = 1/a \text{ ----- (3)}$$

Consequently, **utilization** can be defined as the ratio of the service time to the average inter-arrival time, and is expressed as:

$$U = R_s / R_a \text{ ----- (4)}$$

The value of this ratio varies between 0 and 1.

In the following equation, the term average response rate (S) can be defined as the reciprocal of the average response time (R), and is derived as follows:

S = service rate – arrival rate

Consequently,

$$\begin{aligned} R &= 1 / (\text{service rate} - \text{arrival rate}) \\ R &= 1 / (1/R_s - 1/R_a) \\ &= 1 / (1/R_s - a) && \text{(from eq. 3)} \\ &= R_s / (1 - a \times R_s) \\ R &= R_s / (1 - U) \text{ ----- (5)} && \text{(from eq. 2)} \end{aligned}$$

As a result,

$$\text{Average response time (R)} = \text{service time} / (1 - \text{utilization}) \quad \text{(from equation 2)}$$

Utilization (U) can also be used to represent the average number of I/O requests on the controller, as shown in the following:

Number of requests in the queue (N_Q) = Number of requests in the system (N)

– Number of requests on the controller or utilization (U). Number of requests in a queue is also referred to as *average queue size*.

$$\begin{aligned}
 N_q &= N - U \\
 &= a \times R - U && \text{(from eq. 1)} \\
 &= a \times (R_s / (1 - U)) - U && \text{(from eq. 5)} \\
 &= (R_s / R_a) / (1 - U) - U && \text{(from eq. 3)} \\
 &= U / (1 - U) - U && \text{(from eq. 4)} \\
 &= U (1 / (1 - U) - 1) \\
 &= U^2 / (1 - U) \text{ ----- (6)}
 \end{aligned}$$

The time spent by a request in the queue is equal to the time spent by a request in the system, or the average response time minus the time spent by a request on the controller for processing:

$$\begin{aligned}
 &= R_s / (1 - U) - R_s \text{ (from eq. 5)} \\
 &= U \times R_s / (1 - U) \\
 &= U \times \text{avg. response time} \\
 &= \text{Utilization} \times R \text{ ----- (7)}
 \end{aligned}$$

Ex: Consider a disk I/O system in which an I/O request arrives at a rate of 100 I/Os per second. The service time, R_s , is 8 ms. The following measures of disk performance can be computed using the relationships developed above — utilization of I/O controller (U), total response time (R), average queue size [$U^2 / (1 - U)$] and total time spent by a request in a queue ($U \times R$), as follows:

Arrival rate (a) = 100 I/O/s; consequently, the arrival time

$$R_a = 1/a = 10 \text{ ms}$$

$$R_s = 8 \text{ ms (given)}$$

1. Utilization (U) = $R_s / R_a = 8 / 10 = 0.8$ or 80%
2. Response time (R) = $R_s / (1 - U) = 8 / (1 - 0.8) = 40 \text{ ms}$
3. Average queue size = $U^2 / (1 - U) = (0.8)^2 / (1 - 0.8) = 3.2$
4. Time spent by a request in a queue = $U \times R$, or the total response timeservice time = 32 ms

Now, if controller power is doubled, the service time is halved; consequently, $R_s = 4 \text{ ms}$ in this scenario.

1. Utilization (U) = $4 / 10 = 0.4$ or 40%
2. Response time (R) = $4 / (1 - 0.4) = 6.67 \text{ ms}$
3. Average queue size = $(0.4)^2 / (1 - 0.4) = 0.26$
4. Time spent by a request in a queue = $0.4 \times 6.67 = 2.67 \text{ ms}$

As a result, it can be concluded that by reducing the service time (the sum of seek time, latency, and internal transfer rate) or utilization by half, the response time can be reduced drastically (almost six times in the preceding example). The relationship between utilization and response time is shown in Figure 2-10.

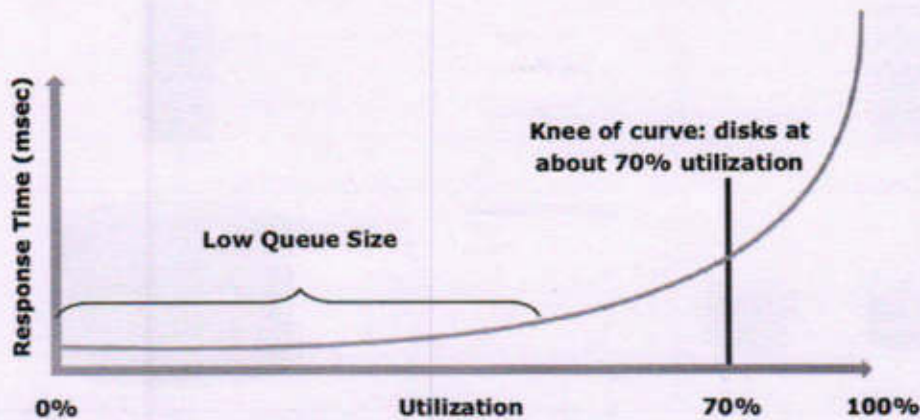


Figure 2-10: Utilization vs. Response time

2.5 Logical Components of the Host

The logical components of a host consist of the software applications and protocols that enable data communication with the user as well as the physical components. Following are the logical components of a host:

- Operating system
- Device drivers
- Volume manager
- File system
- Application

2.5.1 Operating System

An *operating system* controls all aspects of the computing environment. It works between the application and physical components of the computer system. One of the services it provides to the application is data access. The operating system also monitors and responds to user actions and the environment. It organizes and controls hardware components and manages the allocation of hardware resources.

2.5.2 Device Driver

A *device driver* is special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a hard drive. A device driver enables the operating system to recognize the device and to use a standard interface (provided as an *application programming interface*, or *API*) to access and control devices. Device drivers are hardware dependent and operating system specific.

2.5.3 Volume Manager

Disk partitioning was introduced to improve the flexibility and utilization of HDDs. In partitioning, an HDD is divided into logical containers called *logical volumes (LVs)* (see Figure 2-11).

Concatenation is the process of grouping several smaller physical drives and presenting them to the host as one logical drive (see Figure 2-11).

The evolution of *Logical Volume Managers (LVMs)* enabled the dynamic extension of file system capacity and efficient storage management. LVM is software that runs on the host computer and manages the logical and physical storage. LVM is an optional, intermediate layer between the file system and the physical disk. It can aggregate several smaller disks to form a larger virtual disk or to partition a larger-capacity disk into virtual, smaller-capacity disks, which are then presented to applications. The LVM provides optimized storage access and simplifies storage resource management. It hides details about the physical disk and the location of data on the disk; and it enables administrators to change the storage allocation without changing the hardware, even when the application is running.

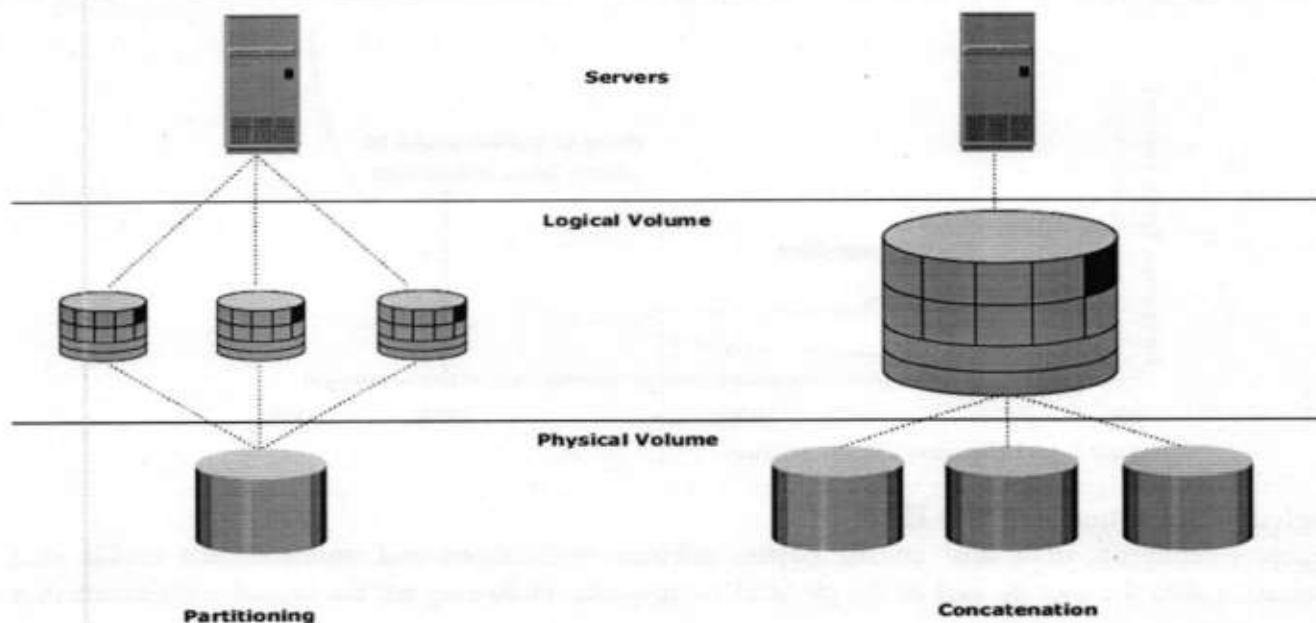


Figure 2-11: Disk partitioning and concatenation

The basic LVM components are the *physical volumes*, *volume groups*, and *logical volumes*. In LVM terminology, each physical disk connected to the host system is a *physical volume (PV)*. LVM converts the physical storage provided by the physical volumes to a logical view of storage, which is then used by the operating system and applications. A *volume group* is created by grouping together one or more physical volumes. A unique *physical volume identifier (PVID)* is assigned to each physical volume when it is initialized for use by the LVM.

Logical volumes are created within a given volume group. A logical volume can be thought of as a virtual disk partition, while the volume group itself can be thought of as a disk. A volume group can have a number of logical volumes. The size of a logical volume is based on a multiple of the physical extents.

2.5.4 File System

A *file* is a collection of related records or data stored as a unit with a name. A *file system* is a hierarchical structure of files. File systems enable easy access to data files residing within a disk drive, a disk partition, or a logical volume. A file system needs host-based logical structures and software routines that control access to files. It provides users with the functionality to create, modify, delete, and access files. A file system organizes data in a structured hierarchical manner via the use of directories, which are containers for storing pointers to multiple files. All file systems maintain a pointer map to the directories, subdirectories, and files that are part of the file system. Some of the common file systems are as follows:

- FAT 32 (File Allocation Table) for Microsoft Windows
- NT File System (NTFS) for Microsoft Windows
- UNIX File System (UFS) for UNIX
- Extended File System (EXT2/3) for Linux

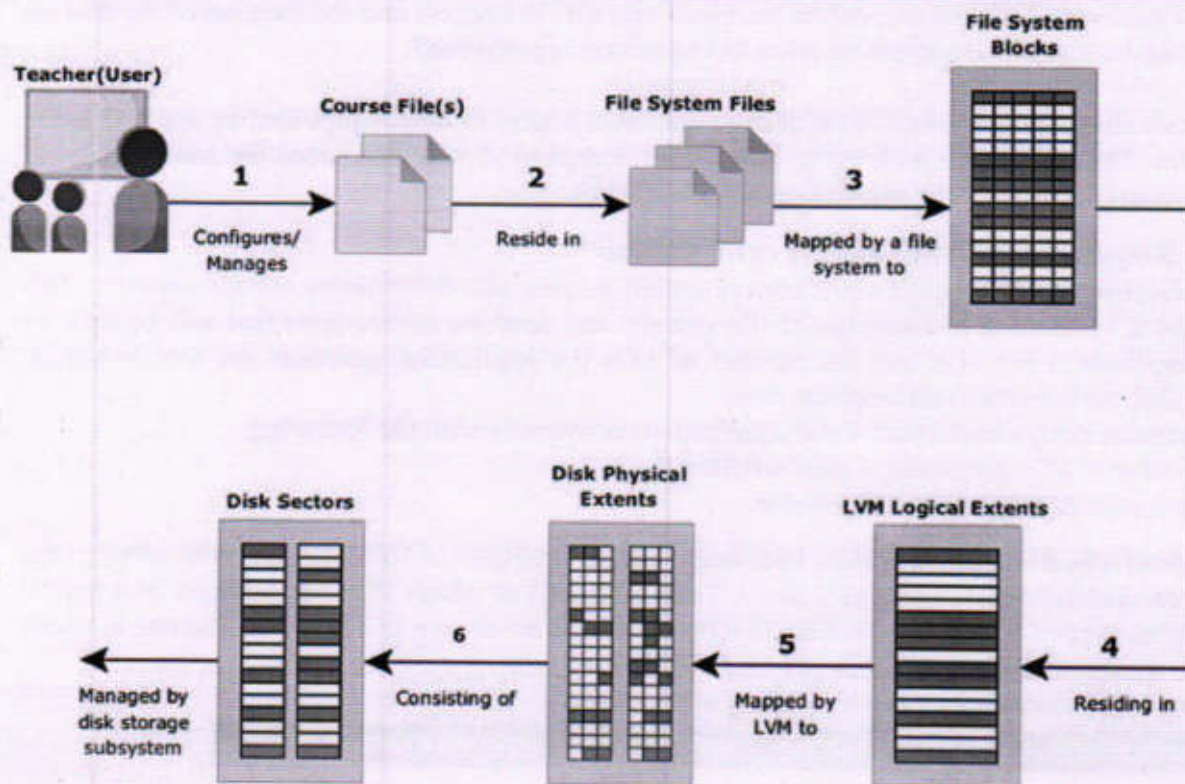


Figure 2-12: Process of mapping user files to disk storage

Figure 2-12 shows the following process of mapping user files to the disk storage subsystem with an LVM:

1. Files are created and managed by users and applications.
2. These files reside in the file systems.
3. The file systems are then mapped to units of data, or file system blocks.
4. The file system blocks are mapped to logical extents.
5. These in turn are mapped to disk physical extents either by the operating system or by the LVM.
6. These physical extents are mapped to the disk storage subsystem.

2.5.5 Application

An *application* is a computer program that provides the logic for computing operations. It provides an interface between the user and the host and among multiple hosts. Conventional business applications using databases have a three-tiered architecture — the application user interface forms the front-end tier; the computing logic forms, or the application itself is, the middle tier; and the underlying databases that organize the data form the back-end tier. The application sends requests to the underlying operating system to perform read/write (R/W) operations on the storage devices. Applications can be layered on the database, which in turn uses the OS services to perform R/W operations to storage devices. These R/W operations (I/O operations) enable transactions between the front-end and back-end tiers.

Data access can be classified as block-level or file-level depending on whether the application uses a logical block address or the file name and a file record identifier to read from and write to a disk.

Block-Level Access

Block-level access is the basic mechanism for disk access. In this type of access, data is stored and retrieved from disks by specifying the logical block address. The block address is derived based on the geometric configuration of the disks. Block size defines the basic unit of data storage and retrieval by an application.

Databases, such as Oracle and SQL Server, define the block size for data access and the location of the data on the disk in terms of the logical block address when an I/O operation is performed.

File-Level Access

File-level access is an abstraction of block-level access. File-level access to data is provided by specifying the name and path of the file. It uses the underlying block-level access to storage and hides the complexities of logical block addressing (LBA) from the application and the DBMS.

2.6 Application Requirements and Disk Performance

The analysis of application storage requirements conventionally begins with determining storage capacity. This can be easily estimated by the size and number of file systems and database components that will be used by applications. The application I/O size and the number of I/Os the application generates are two important measures affecting disk performance and response time.

Consequently, the storage design and layout for an application commences with the following:

1. Analyzing the number of I/Os generated at peak workload
2. Documenting the application I/O size or block size.

Consider an example of a SCSI controller (SCSI interface) with a throughput of 160 MB/s and disk service time $R_s = 0.3$ ms. The computation of the rate ($1 / [R_s + \text{Transfer time}]$) at which I/Os are serviced in a typical database I/O with block sizes 4 KB, 8 KB, 16 KB, 32 KB, and 64 KB are shown in Table 2-1. The rate at which the application I/Os are serviced is termed I/Os per second (IOPS).

Table 2-1: Maximum IOPS Performed by SCSI Controller

BLOCK SIZE	TRANSFER TIME (MS)	IOPS = $1 / (R_s + \text{TRANSFER TIME})$
4 KB	4 KB / 160 MB = 0.025	$1 / (0.3 + 0.025) = 3,076$
8 KB	8 KB / 160 MB = 0.05	$1 / (0.3 + 0.05) = 2,857$
16 KB	16 KB / 160 MB = 0.1	$1 / (0.3 + 0.1) = 2,500$
32 KB	32 KB / 160 MB = 0.2	$1 / (0.3 + 0.2) = 2,000$
64 KB	64 KB / 160 MB = 0.4	$1 / (0.3 + 0.4) = 1,428$

As a result, the number of IOPS per controller depends on the I/O block size and ranges from 1,400 (for 64 KB) to 3,100 (for 4 KB).

The disk service time (RS) is a key measure of disk performance; R_s along with disk utilization rate (U) determines the I/O response time for applications. As shown earlier in this chapter, the total disk service time (RS) is the sum of seek time (E), rotational latency (L), and the internal transfer time (X):

$$R_s = E + L + X$$

E is determined based on the randomness of the I/O request. L and X are measures provided by disk vendors as technical specifications of the disk.

Consider an example with the following specifications provided for a disk:

- Average seek time of 5 ms in a random I/O environment, or $E = 5$ ms.
- Disk rotation speed of 15,000 rpm — from which rotational latency (L) can be determined, which is one half of the time taken for a full rotation or $L = (0.5 / 15,000 \text{ rpm})$ expressed in ms).
- 40 MB/s internal data transfer rate, from which the internal transfer time (X) is derived based on the block size of the I/O — for example, an I/O with a block size of 32 KB or $X = 32 \text{ KB} / 40 \text{ MB}$.

Consequently, $R_s = 5 \text{ ms} + (0.5 / 15,000) + 32 \text{ KB} / 40 \text{ MB} = 7.8 \text{ ms}$.

The maximum number of I/Os serviced per second or IOPS = $1 / R_s$.

In other words, for an I/O with a block size of 32 KB and $R_s = 7.8 \text{ ms}$, the maximum IOPS will be $1 / (7.8 \times 10^{-3}) = 128 \text{ IOPS}$.

Table 2-2: Maximum IOPS Performed by Disk Drive

BLOCK SIZE	RS = E+L+X	IOPS = 1/RS
4 KB	5 ms + (0.5 / 15,000 rpm) + 4K / 40MB = 5 + 2 + 0.1 = 7.1	140
8 KB	5 ms + (0.5 / 15,000 rpm) + 8K / 40MB = 5 + 2 + 0.2 = 7.2	139
16 KB	5 ms + (0.5 / 15,000 rpm) + 16K / 40MB = 5 + 2 + 0.4 = 7.4	135
32 KB	5 ms + (0.5 / 15,000 rpm) + 32K / 40MB = 5 + 2 + 0.8 = 7.8	128
64 KB	5 ms + (0.5 / 15,000 rpm) + 64K / 40MB = 5 + 2 + 1.6 = 8.6	116

For the example in Table 2-2, the I/O response time (R) for an I/O with a block size of 64 KB will be approximately 215 ms when the controller works close to 96 percent utilization, as shown here:

$$\begin{aligned} R &= R_s / (1 - U) \\ &= 8.6 / (1 - 0.96) \\ &= 215 \text{ ms} \end{aligned}$$

If an application demands a faster response time, then the utilization for the disks should be maintained below 70 percent, or the knee of the curve, after which the response time increases exponentially.

The total number of disks required (N) for an application is computed as follows:

If C is the number of disks required to meet the capacity and I is the number of disks required for meeting IOPS, then

$$N = \text{Max}(C, I)$$

Disk vendors publish the disk potential in terms of IOPS based on the benchmark they carry out for different block sizes and application environments.

Consider an example in which the capacity requirements for an application are 1.46 TB. The peak workload generated by the application is estimated at 9,000 IOPS. The vendor specifies that a 146 GB, 15,000-rpm drive is capable of a maximum of 180 IOPS (U = 70%).

In this example, the number of disks required to meet the capacity requirements will be only 1.46 TB / 146 GB = 10 disks. To meet 9,000 IOPS, 50 disks will be required (9,000 / 180). As a result, the number of disks required to meet the application demand will be Max (10, 50) = 50 disks.

Storage System.

Evolution of Storage Architecture

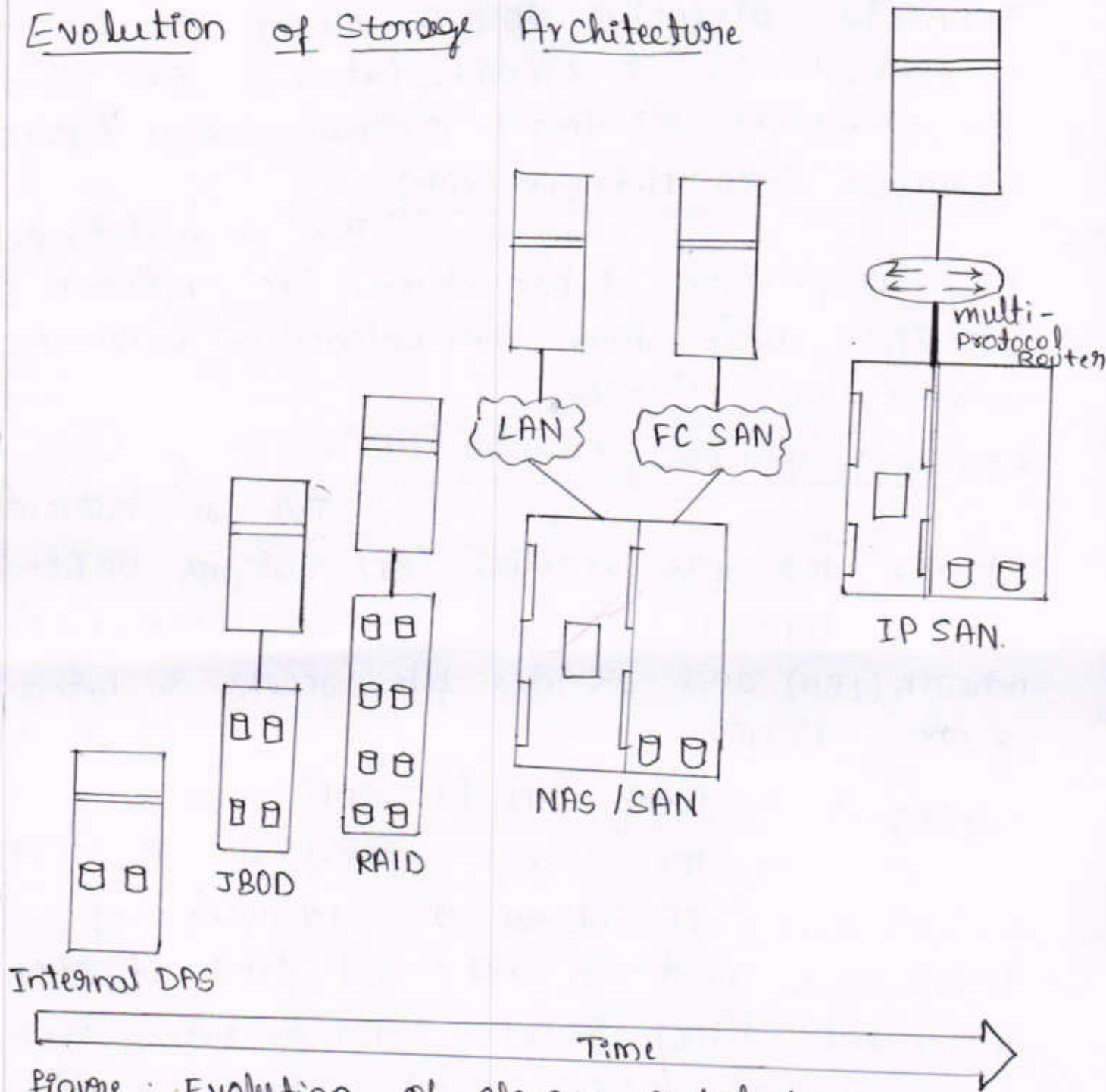


Figure : Evolution of storage architectures.

* Redundant Array of Independent Disks (RAID):

This technology was developed to address the cost, performance and availability requirements of data. It continues to evolve today and is used in all storage architecture such as DAS, SAN, and so on.

* Direct-attached storage (DAS): This type of storage connects directly to a server or a group of servers in a cluster. External DAS alleviated the challenges of limited internal storage capacity.

* Storage area Network (SAN): This is a dedicated, high-performance Fibre channel (FC) network to facilitate block-level communication between servers and storage.

* Network-attached storage (NAS): This is dedicated storage for file serving applications. Unlike a SAN, it connects to an existing communication network (LAN) and provides file access to heterogeneous clients.

* Internet Protocol SAN (IP-SAN): One of the latest evolutions in storage architecture, IP-SAN is a convergence of technologies used in SAN and NAS. IP-SAN provides block-level communication across a local or wide area network (LAN or WAN), resulting in greater consolidation and availability of data.

Storage technology and architecture continues to evolve, which enables organizations to consolidate, protect, optimize and leverage.

organizations maintain data centers to provide centralized data processing capabilities across the enterprise. Data centers store and manage large amounts of mission-critical data.

Core Elements.

Five core elements are essential for the basic functionality of a data center.

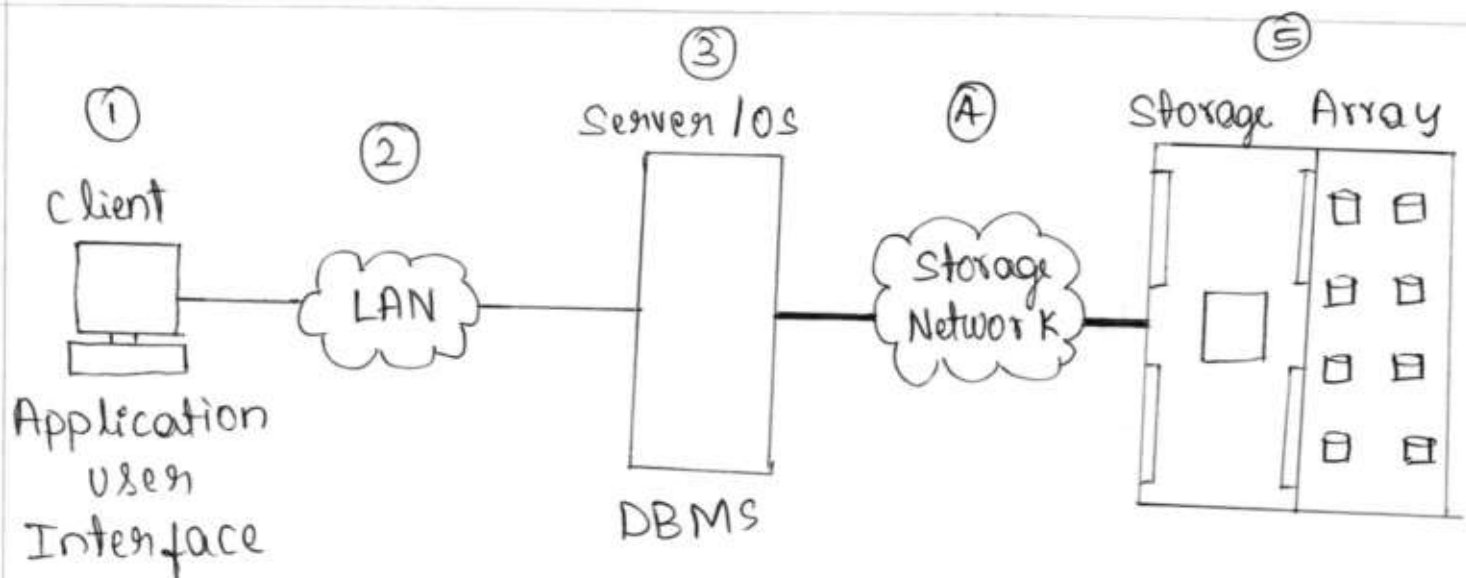
1] Application: An application is a computer program that provides the logic for computing operations. Applications, such as an order processing system, can be layered on a database, which in turn uses operating system services to perform read/write operations to storage devices.

2] Database: More commonly, a database management system (DBMS) provides a structured way to store data in logically organized tables that are interrelated.

3] Servers and operating system: A computing platform that runs applications and databases.

4] Network: A data path that facilitates communication between clients and servers or between servers and storage.

5] Storage array: A device that stores data persistently for subsequent use.



- ① A customer places an order through the GUI of the order processing application software located on the client computer.
- ② The client connects to the server over the LAN and accesses the DBMS located on the server to update the relevant information such as the customer name, address, payment method, products ordered, and quantity ordered.
- ③ The DBMS uses the server operating system to read and write this data to the database located on physical disks in the storage array.
- ④ The Storage Network provides the communication link between the server and the storage array & transports the read or write commands between them.
- ⑤ The storage array, after receiving the read or write commands from the server, performs the necessary operations to store the data on physical disks.

Key Requirements for Data Center Elements.

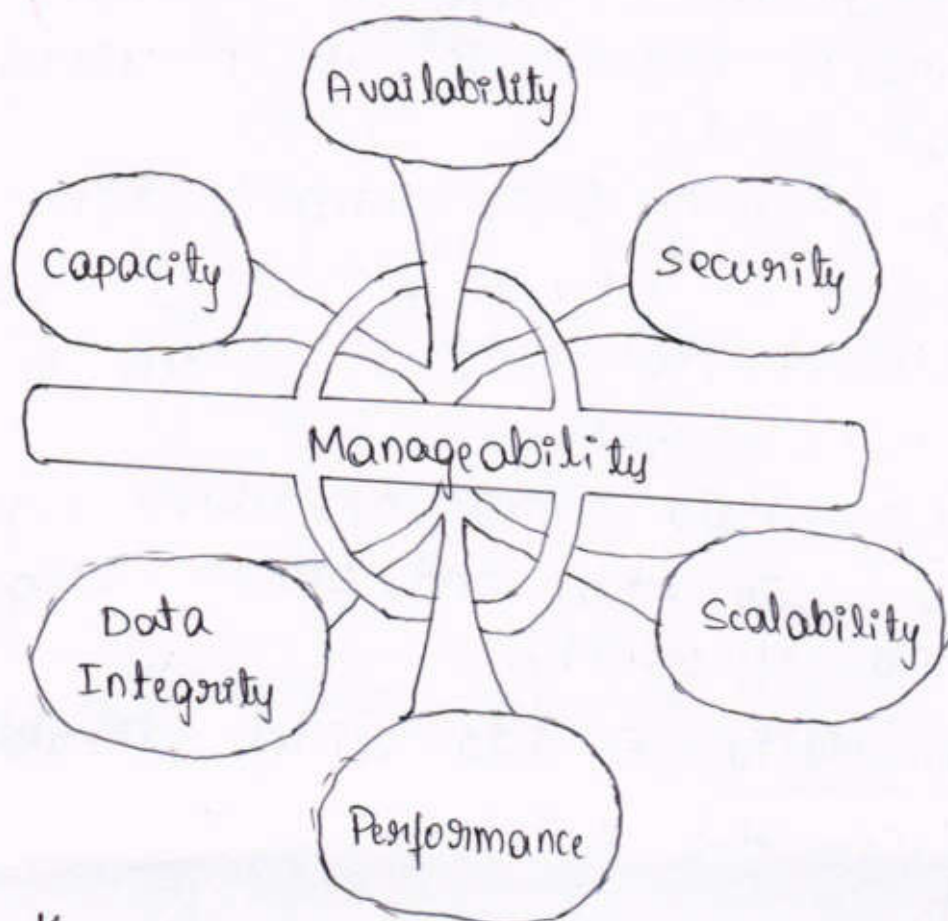


Figure: Key characteristics of data center elements.

- 1] Availability: All data center elements should be designed to ensure accessibility.
- 2] Security: Policies, procedures, and proper integration of the data center core elements that will prevent unauthorized access to information must be established.
- 3] Scalability: Data center operations should be able to allocate additional processing capabilities or storage on demand.

4] Performance: All the core elements of the data center should be able to provide optimal performance and service all processing requests at high speed.

5] Data integrity: Data integrity refers to mechanisms such as error correction codes or parity bits which ensure that data is written to disk exactly as it was received.

6] Capacity: Data center operations require adequate resources to store and process large amounts of data efficiently.

7] Manageability: A data center should perform all operations and activities in the most efficient manner.

Managing Storage Infrastructure.

Managing a modern, complex data center involves many tasks. Key management activities include:

1] Monitoring: is the continuous collection of information & the review of the entire data center infrastructure.

2] Reporting is done periodically on resource performance, capacity, and utilization.

3] Provisioning: is the process of providing the hardware, software and other resources needed to run a data center. provisioning activities include capacity and resource planning.

Virtualization :- It is a technique of abstracting physical resource such as compute, storage and network and making them appear as logical resource.

* Common examples of virtualization are virtual memory used on compute systems and partitioning of raw disks

* Virtualization enables group of physical resources and providing an combined view of the material resource capabilities.

* Virtualization resources can be created and provisioned from the shared physical resources.

* In today's fast-paced and competitive environment, organizations must be agile and flexible to meet changing market requirements. This leads to rapid expansion and upgrade of resources while meeting budgets.

Cloud Computing :

addresses these challenges efficiently

* Cloud computing enables individuals or business to use IT resources as a service over the networks.

- * It provides highly scalable and flexible computing that enables provisioning of resources on demand.
- * Users can scale up or scale down the demand of computing resources, including storage capacity, with minimal management effort or service provider interaction.
- * cloud computing enables consumption based metering: therefore, consumers pay only for the resources they use such as CPU hours used, amount of data transferred, and gigabytes of data stored.

The Various Software Components that are Essential parts of a host System.

① Operating System.

- * It works between the application and the physical components of a compute system.
- * One of the services it provides to the application is data access.
- * The Operating system also monitors and responds to user actions and the environment.
- * It organizes & controls hardware components & manages the allocation of hardware resources.
- * It provides basic security for the access & usage of all managed resources.
- * An operating sly also performs basic storage management tasks such as the file sly, device drivers.
- * In a virtualized compute environment, the virtualization layer works b/w the operating sly & the hardware resources.
- * In a typical implementation, the OS works as a guest and performs only the activities related to application interaction.
- * Hardware management functions are handled by the virtualization layer.

Memory virtualization:

- * Memory virtualization enables multiple applications & processes, whose aggregate memory requirement is greater than the available physical memory, to run on a host without impacting each other.
- * Memory virtualization is an operating system feature that virtualizes the physical memory (RAM) of a host.
- * It creates virtual memory with an address space larger than the physical memory space present in the computer system.
- * The operating system utility that manages the virtual memory is known as the virtual memory manager (VMM).
- * The VMM manages the virtual-to-physical memory mapping & fetches data from the disk storage when a process references a virtual address that points to data at the disk storage.
- * The space used by the VMM on the disk is known as a swap space. A swap space (also known as page file or swap file) is a portion of the disk drive that appears to be physical memory to the operating system.
- * In a virtual memory implementation, the memory of a system is divided into contiguous blocks of fixed-size pages.
- * A process known as paging moves inactive physical memory pages onto the swap file & brings them back to the physical memory when required.

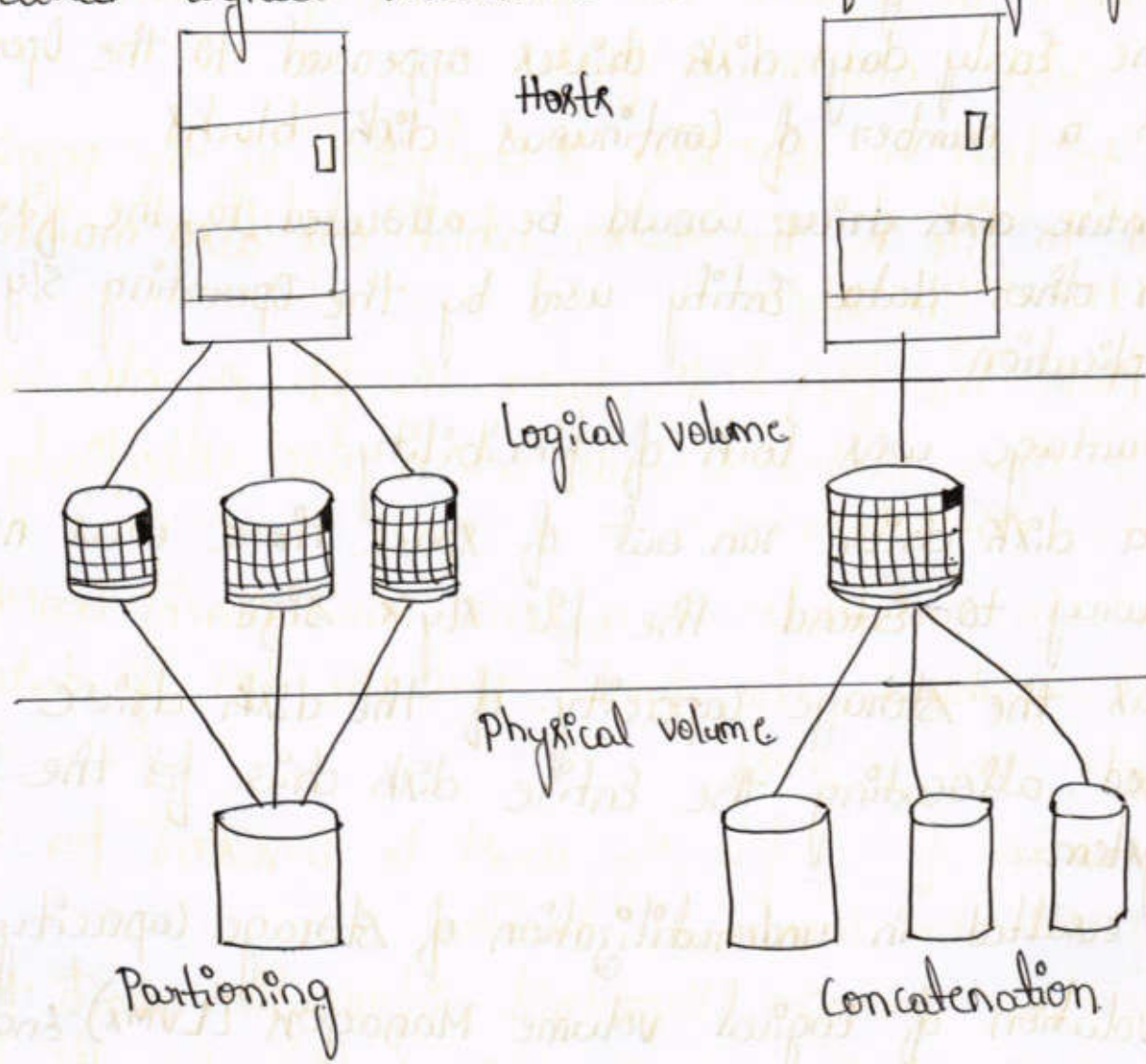
② Device Driver

- * A device driver is special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a disk drive.
- * A device driver enables the operating system to recognize the device and to access and control devices.
- * Device drivers are hardware-dependent and operating system-specific.

③ Volume Manager

- * In the early days, disk drives appeared to the operating system as a number of continuous disk blocks.
- * The entire disk drive would be allocated to the file system or other data entity used by the operating system or application.
- * disadvantage was lack of flexibility.
- * when a disk drive ran out of space, there was no easy way to extend the file system's size.
- * Also, as the storage capacity of the disk drive increased, allocating the entire disk drive for the file system.
- * often resulted in underutilization of storage capacity.
- * The evolution of Logical Volume Managers (LVMs) enabled dynamic extension of file system capacity & efficient storage management.

- * The LVM is software that runs on the computer and manages logical and physical storage.
- * It can partition a larger-capacity disk into virtual, smaller-capacity volumes (the process is called partitioning) or aggregate several smaller disks to form a larger virtual volume. (The process is called concatenation).
- * Disk partitioning was introduced to improve the flexibility and utilization of disk drives. In partitioning, a disk drive is divided into logical components called logical volumes (LVM) (see following figure).



Disk partitioning and Concatenation.

④ File System.

- * A File is a collection of related records or data stored as a unit with a name.
- * A file system is a hierarchical structure of files.
- * A file system enables easy access to data files residing within a disk drive, a disk partition, or a logical volume.
- * A file system contains logical structures & procedures that control access to files.
- * It provides users with the functionality to create, modify, delete, and access files.
- * Access to files on the disks is controlled by the permissions assigned to file by the owner, which are also maintained by the file system.
- * Apart from the files & directories, the file system also includes a no. of other related records, which are collectively called the metadata.
- * For ex, the metadata in a UNIX environment consists of the superblock, the inodes, and the list of data blocks free and in use.
- * The metadata of a file system must be consistent for the file system to be considered healthy.
- * A superblock contains important information about the file system, such as the file system type, creation & modification dates, size and layout.

* It also contains the count of available resources (such as the number of free blocks, inodes, & so on) & a standard indicating the increase grade of the file sly.

* An inode is associated with every file & directory & contains information such as the file length, ownership, access privileges, time of last access/modification, no. of links, and the address of the data.

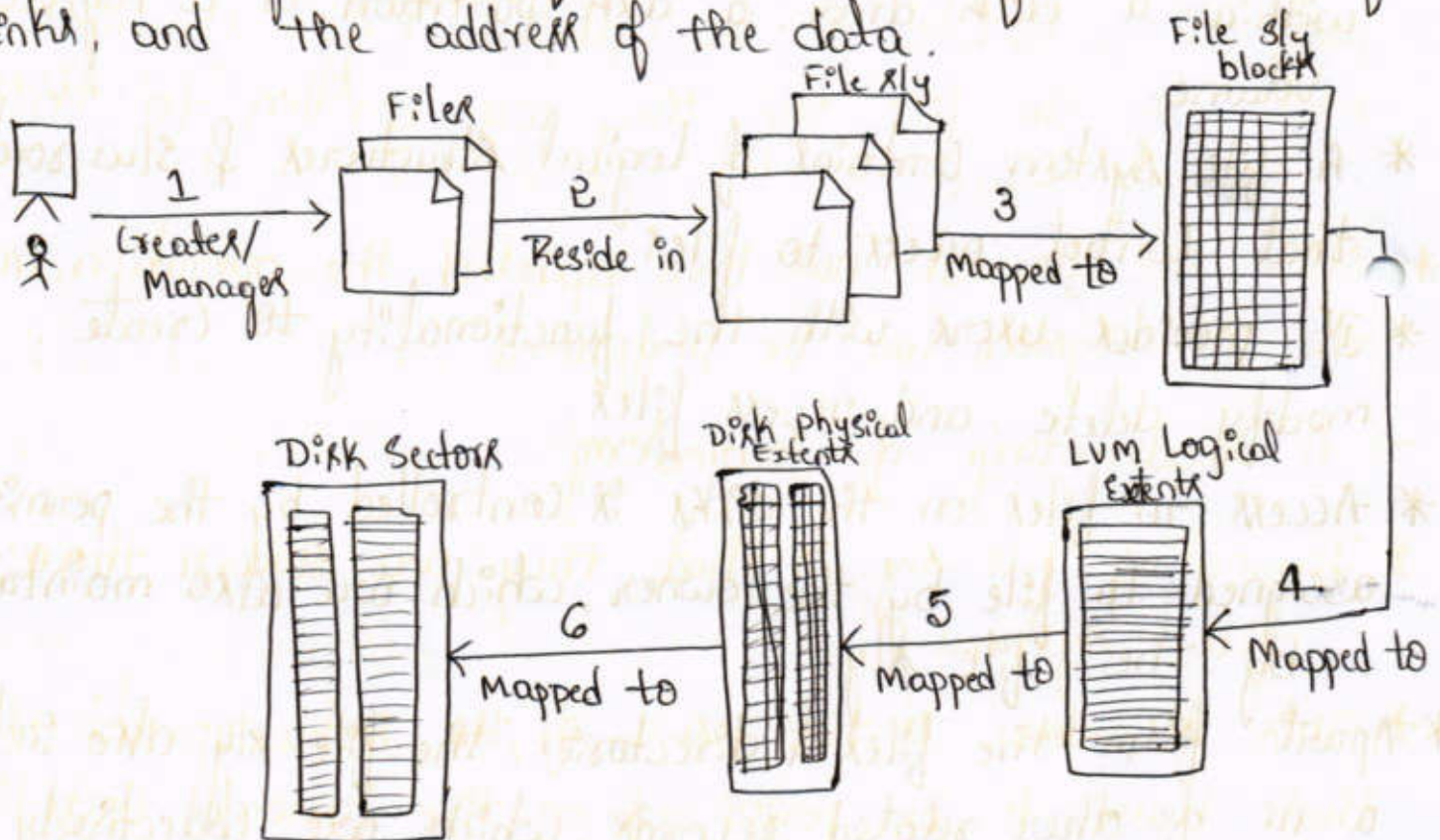


Fig: process of mapping user files to disk storage.

* A file system can be either a journaling file sly or a non journaling file system.

* Non journaling file system cause a potential loss of files because they use separate buffers to update their data and metadata.

* If the system crashes during the write process, the metadata or data might be lost or corrupted.

* Sly reboots, the file system attempts to update the metadata structures by examining & repairing them.

- * If there is insufficient information to re-create the wanted or original structure, the files might be misplaced or lost, resulting in corrupted file sy.
- * A journaling file sy uses a separate area called a log or journal.
- * This journal might contain all the data to be written or just the metadata to be updated. Before changes are made to the file sy, they are written to this separate area.
- * After the journal has been updated, the operation on the file system can be performed.

→ A disadvantage of journaling.

- * Journaling file sy is that they are slower than other file systems.
- * This slowdown is the result of the extra operations that have to be performed on the journal each time the file system is changed.

⑤ Compute virtualization.

- * It enables multiple operating sy to run concurrently on single or clustered physical machines.
- * This technique enables creating portable virtual compute systems called virtual machines (VM).
- * Each VM runs on operating sy and application instance in an inaccessible manner.

- * Compute virtualization is achieved by a virtualization layer that resides between the hardware & virtual machines. This layer is also called the hypervisor.
- * The hypervisor provides hardware resources, such as CPU, memory and I/O to all the virtual machines.
- * Within a physical server, a large number of virtual machines can be created depending on the H/W capabilities of the physical server.
- * Physical servers often face resource - conflict issues when two or more applications running on the server have conflicting requirements.
- * These issues are further compounded with an application high-availability requirements. As a result, the servers are limited to serve only one application at a time, as shown in the following figure.

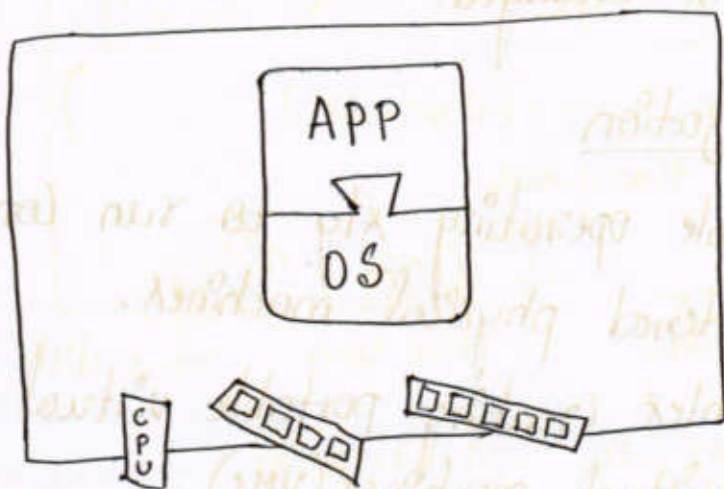


Fig : Before Compute virtualization.

* on the other hand, many applications do not take full advantage of the hardware capabilities available to them.

* Consequently, resources such as processors, memory, and storage remain underutilized. Compute virtualization enables users to overcome these challenges as show in the following figure by allowing multiple operating systems and applications to run on a single physical machine.

* This technique significantly improved server utilization and provides server consolidation.

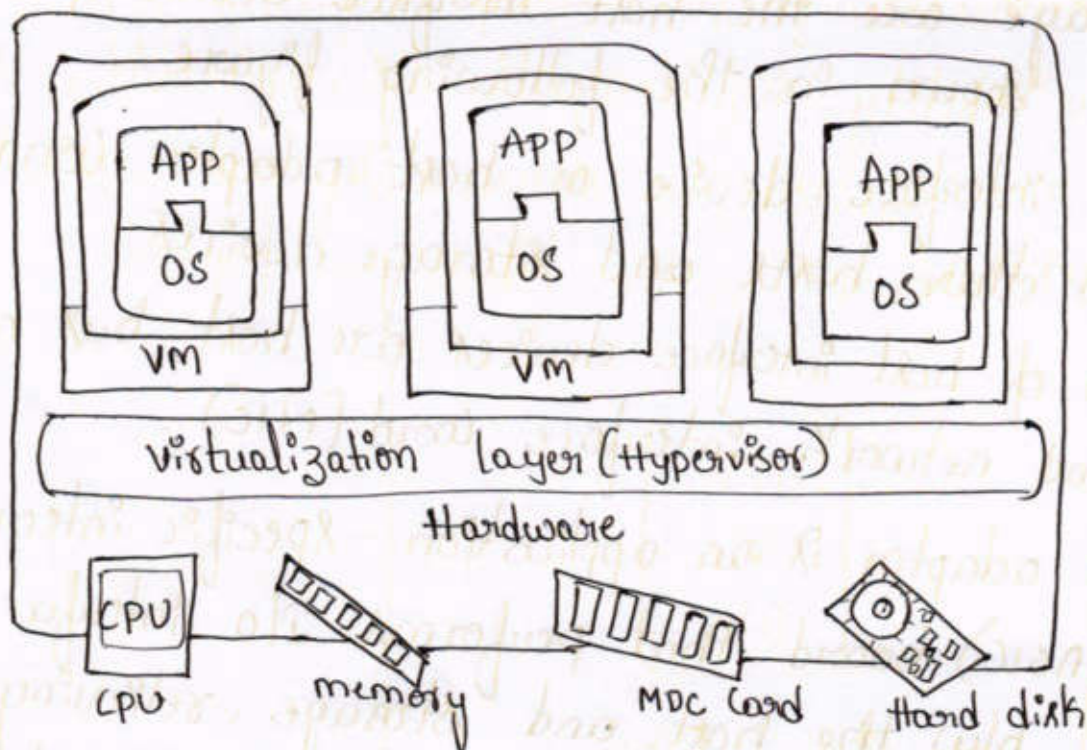


Fig : After Compute Virtualization.

⑤ Connectivity:

* Connectivity refers to the interconnection between hosts or between a host and peripheral devices, such as printers or storage devices.

* Connectivity and communication between host and storage are enabled using physical components and interface protocols.

Physical Components of Connectivity:

* The physical components of connectivity are the I/O elements that connect the host to storage.

* Three physical components of connectivity b/w the host and storage are the host interface device, port & cable as shown in the following figure.

* A host interface device or host adapter connects a host to other hosts and storage devices.

* Examples of host interface devices are host bus adapter (HBA) and network interface card (NIC).

* Host bus adapter is an application-specific integrated circuit (ASIC) board that performs I/O interface functions b/w the host and storage, relieving the CPU from additional I/O processing workload.

* A host typically contains multiple HBAs.

- * A port is a specialized outlet that enables connectivity between the host and external devices.
- * An HBA may contain one or more ports to connect the host to internal or external devices using copper or fibre optic media.

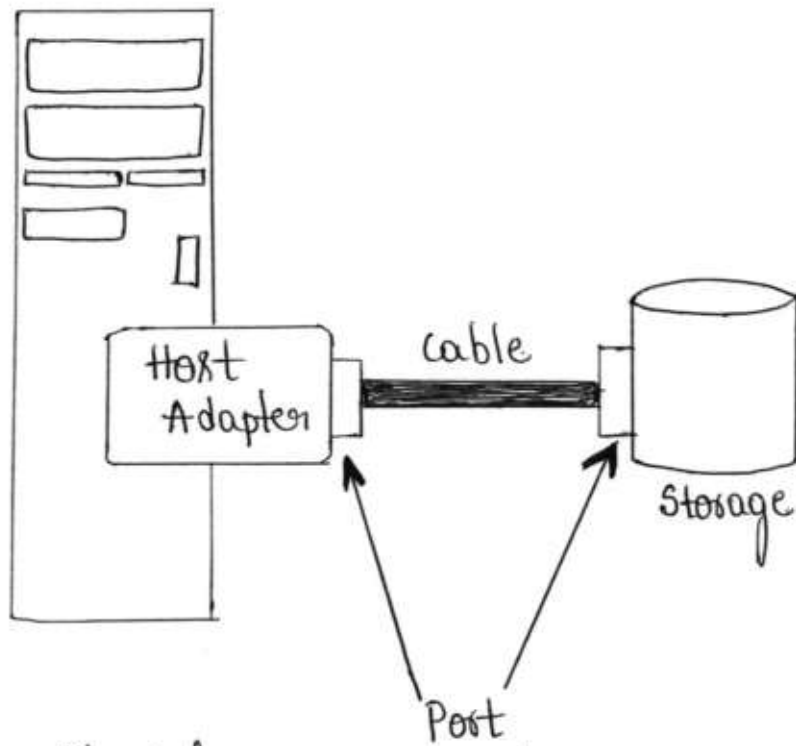


Fig: Physical Components of Connectivity.

Physical components of Connectivity:

The physical components of Connectivity are the HW elements that connect the host to storage.

Three physical components of connectivity b/w the host and storage are the host interface device, port, and cable as shown in the following figure

A host interface device or host adapter connects a host to other hosts and storage devices.

Examples of host interface devices are host bus adapter (HBA) and network interface card (NIC)

Host bus adapter is an application-specific integrated circuit (ASIC) board that performs I/O interface functions b/w the host and storage, relieving the CPU from additional I/O processing workload.

A host typically contains multiple HBAs.

A port is a specialized outlet that enables connectivity b/w the host and external devices.

An HBA may contain one or more ports to connect the host to the storage device. Cables connect hosts to internal or external devices using copper or fibre optic media.

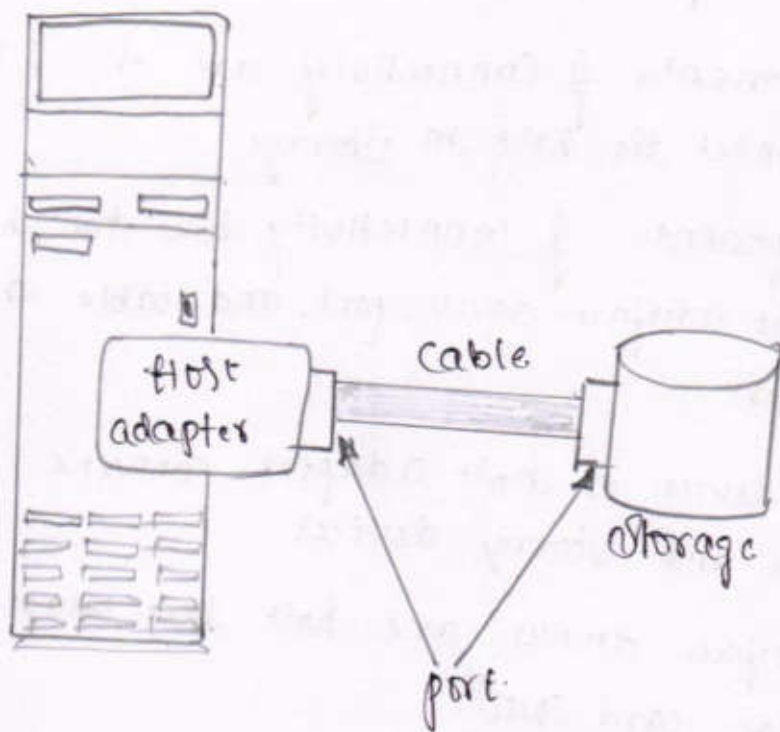


Fig. physical components of connectivity.

Interface protocol:

A protocol enables communication b/w the host and storage.

Protocols are implemented using interface devices (or controllers) at both source and destination.

The popular interface protocols used for host to storage communications are integrated device electronics/advanced technology attachment (IDE/ATA), Small Computer System interface (SCSI), fibre channel. The popular interface protocols used for host to storage communications are IDE/ATA, Small Computer System interface, FC and SP.

* IDE/ATA and Serial ATA:

It is a popular interface protocol standard used for connecting storage devices, such as disk drives and CD-ROM drives.

Internet Protocol (IP):

IP is a new protocol that has been traditionally used for host-to-host traffic.

With the emergence of new technologies, an IP now has become a viable option for host-to-host communication.

IP offers several advantages in terms of cost & maturity & enables organizations to leverage their existing IP-based network.

Storage:

Storage is a core component in a data center.

A storage device uses magnetic, optic, or solid state media.

Disks, tapes and diskettes use magnetic media, whereas CD/DVD uses optical media for storage.

In the past, tapes were the most popular storage option for backups because of their low cost.

Tapes have various limitations in terms of performance and management.

Data is stored on the tape linearly along the length of the tape. Search & retrieval of data are done sequentially and it invariably takes several seconds to access the data. As a result, random data access is slow & time consuming.

This protocol supports parallel transmission and therefore is also known as parallel ATA (PATA) or simply ATA. SATA has a variety of standards and names.

The Ultra ATA/133 version of ATA supports a throughput of 133MB per second.

The Serial version of this protocol supports single bit serial transmission & is known as Serial ATA.

SCSI and Serial SCSI:

SCSI has emerged as a preferred connectivity protocol in high-end computers.

This protocol supports parallel transmission and offers improved performance, scalability, and compatibility compared to ATA.

SCSI supports up to 16 devices on a single bus & provides data transfer rates up to 640MB/s.

A newer version of Serial SCSI supports a data transfer rate up to 6GB/s.

Fibre channel:

It is a widely used protocol for high speed communication to the storage device.

The fibre channel interface provides gigabit network speed. It provides a serial data transmission that operates over copper wire & optical fiber.

The latest version of the FC interface allows transmission of data up to 16GB/s.

On a shared Computing environment, data stored on tape cannot be accessed by multiple applications simultaneously restricting its use to one application at a time.

On a tape drive, the read/write head touches the tape surface, so the tape degrades or wears out after repeated use.

The storage & retrieval requirements of data from the tape & the overhead associated with managing the tape media are significant.

Due to these limitations & availability of low-cost disk drives, tapes are no longer a preferred choice as a backup destination for enterprise class data centers.

Optical disc storage is popular in small, single-user computing environments.

It is frequently used by individuals to store photos or as a backup medium on personal or laptop computers.

It is also used as a distribution medium for small applications. Such as games, or as a means to transfer small amounts of data from one computer system to another.

Optical discs have limited capacity & speed, which limit the use of optical media as a business data storage solution.

3.1 Implementation of RAID

There are two types of RAID implementation, hardware and software. Both have their merits and demerits.

3.1.1 Software RAID

Software RAID uses host-based software to provide RAID functions. It is implemented at the operating-system level and does not use a dedicated hardware controller to manage the RAID array.

Software RAID implementations offer cost and simplicity benefits when compared with hardware RAID.

Limitations of software RAID:

1. **Performance:** Software RAID affects overall system performance. This is due to the additional CPU cycles required to perform RAID calculations.
2. **Supported features:** Software RAID does not support all RAID levels.
3. **Operating system compatibility:** Software RAID is tied to the host operating system hence upgrades to software RAID or to the operating system should be validated for compatibility.

3.1.2 Hardware RAID

In hardware RAID implementations, a specialized hardware controller is implemented either on the host or on the array.

Controller card RAID is host-based hardware RAID implementation in which a specialized **RAID controller** is installed in the host and HDDs are connected to it. **RAID Controller** interacts with the hard disks using PCI bus.

Manufacturers integrate RAID controllers on motherboards. This reduces the overall cost of the system, but does not provide the flexibility required for high-end storage systems.

The external RAID controller is an array-based hardware RAID. It acts as an interface between host and disks. It presents storage volumes to host, which manage the drives using the supported protocol.

Key functions of RAID controllers are:

1. Management and control of disk aggregations
2. Translation of I/O requests between logical disks and physical disks
3. Data regeneration in the event of disk failures.

3.2 RAID Array Components

A RAID array is an enclosure that contains a number of HDDs and the supporting hardware and software to implement RAID.

HDDs inside a RAID array are contained in smaller sub-enclosures. These sub-enclosures, or **physical arrays**, hold a fixed number of HDDs, and also include other supporting hardware, such as power supplies.

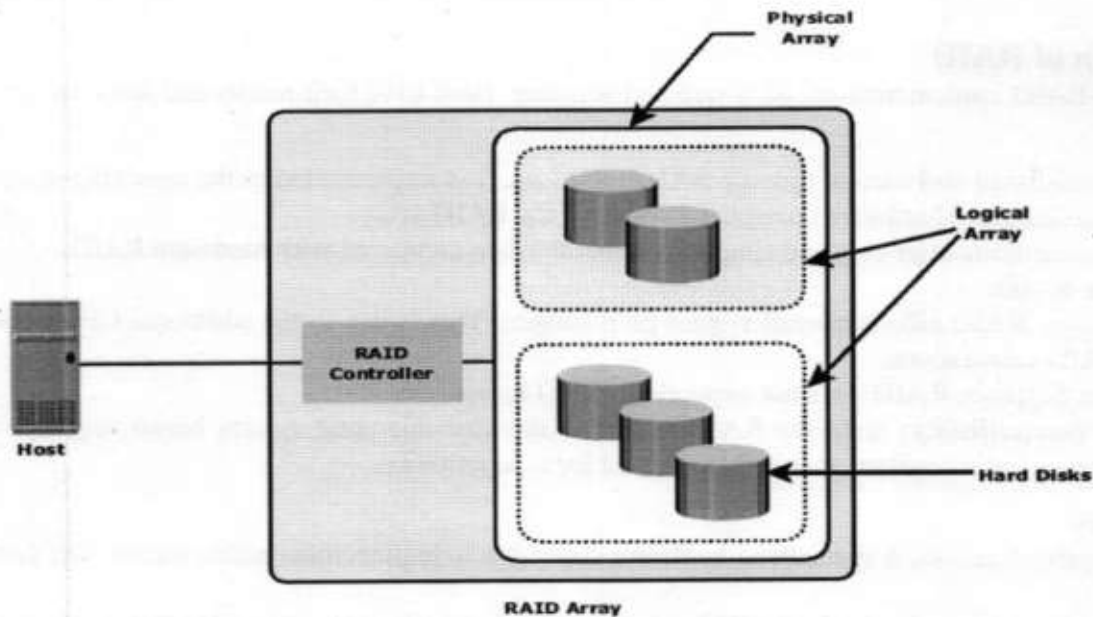


Figure 3-1: Components of RAID array

A subset of disks within a RAID array can be grouped to form logical associations called **logical arrays**, also known as a **RAID set** or a **RAID group** (see Figure 3-1). Logical arrays are comprised of **logical volumes (LV)**. The operating system recognizes the LVs as if they are physical HDDs managed by the RAID controller.

3.3 RAID Levels

RAID levels are defined on the basis of 1) **striping**, 2) **mirroring**, and 3) **parity** techniques. These techniques determine the data availability and performance characteristics of an array. Some RAID arrays use one technique, whereas others use a combination of techniques.

Table 3-1: Raid Levels

LEVELS	BRIEF DESCRIPTION
RAID 0	Striped array with no fault tolerance
RAID 1	Disk mirroring
RAID 3	Parallel access array with dedicated parity disk
RAID 4	Striped array with independent disks and a dedicated parity disk
RAID 5	Striped array with independent disks and distributed parity
RAID 6	Striped array with independent disks and dual distributed parity
Nested	Combinations of RAID levels. Example: RAID 1 + RAID 0

3.3.1 Striping

A RAID set is a group of disks. Within each disk, a predefined number of contiguously addressable disk blocks are defined as **strips**. The set of aligned strips that spans across all the disks within the RAID set is called a **stripe** (Figure 3-2).

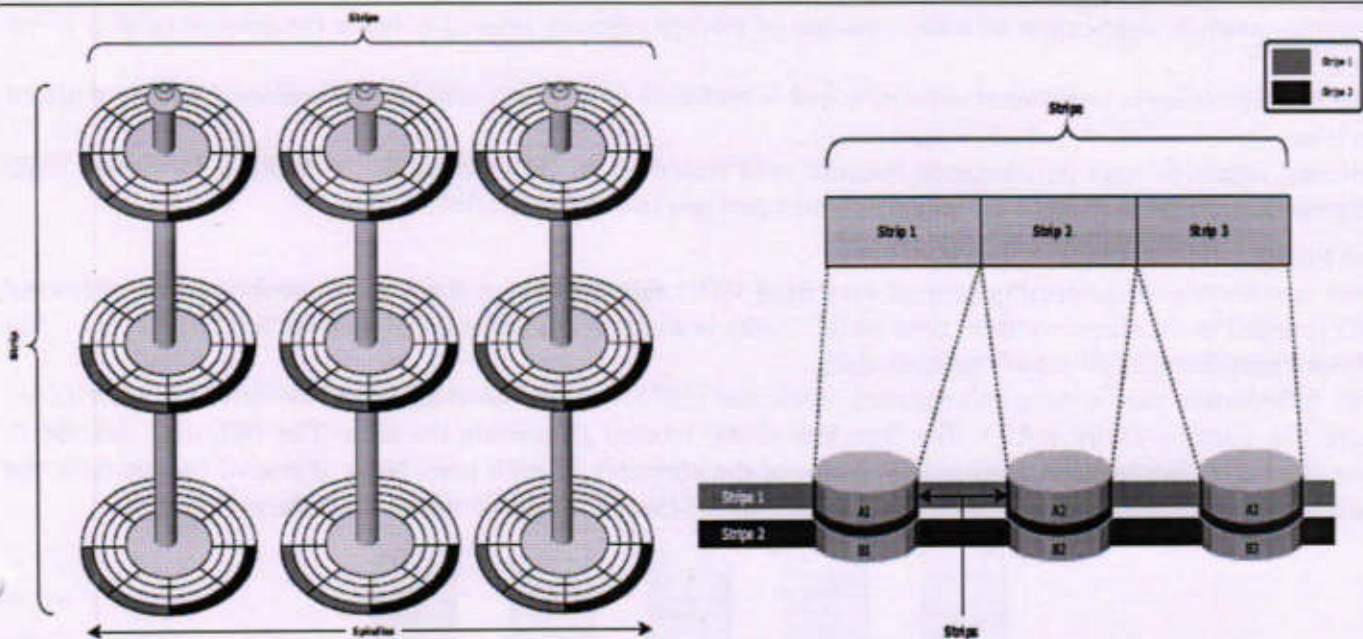


Figure 3-2: Striping

Strip size (also called *stripe depth*) describes the number of blocks in a *stripe*, and is the maximum amount of data that can be written to or read from a single HDD in the set. All strips in a stripe have the same number of blocks.

Stripe width refers to the number of data strips in a stripe.

Striped RAID does not protect data unless parity or mirroring is used. Striping significantly improves I/O performance.

3.3.2 Mirroring

Mirroring is a technique whereby data is stored on two different HDDs, yielding two copies of data. In the event of one HDD failure, the data is intact on the surviving HDD (see Figure 3-3) And the controller continues to service the host's data requests from the surviving disk.

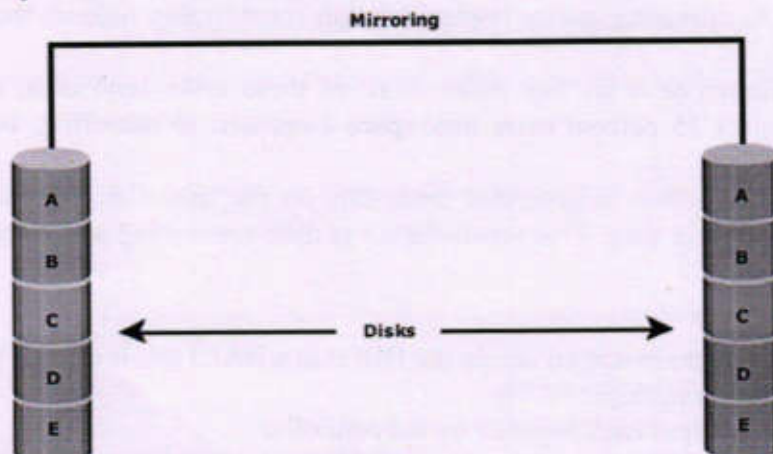


Figure 3-3: Mirrored disks in an array

When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair.

Mirroring provide complete data redundancy, and also enables faster recovery from disk failure.

Mirroring is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of data.

Mirroring involves duplication of data - amount of storage capacity needed is twice the amount of data being stored.

Therefore, mirroring is considered expensive and is preferred for mission-critical applications that cannot afford data loss.

Mirroring improves read performance because read requests can be serviced by both disks. However, write performance decreases, as each write request must perform two writes on the HDDs.

3.3.3 Parity

Parity is a method of protecting striped data from HDD failure without the cost of mirroring. An additional HDD is added to the stripe width to hold parity. Parity is a redundancy check that ensures full protection of data without maintaining a full set of duplicate data.

Parity information can be stored on separate, dedicated HDDs or distributed across all the drives in a RAID set. Figure 3-4 shows a parity RAID. The first four disks, labeled *D*, contain the data. The fifth disk, labeled *P*, stores the parity information, which is the sum of the elements in each row. Now, if one of the *D*s fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value.

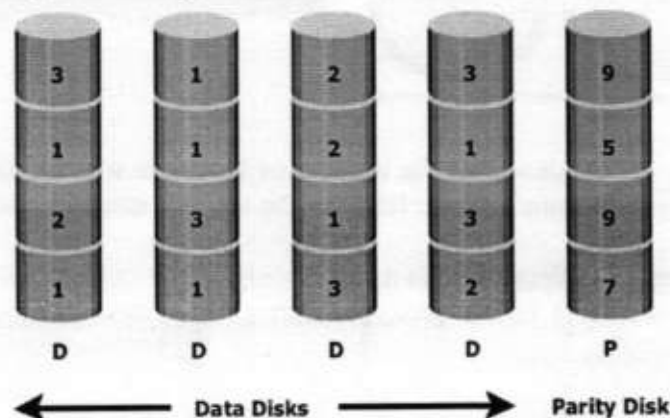


Figure 3-4: Parity RAID

The computation of parity is represented as a simple arithmetic operation on the data. Parity calculation is a *bitwise XOR* operation. Calculation of parity is a function of the RAID controller.

Advantage: Compared to mirroring, parity implementation considerably reduces the cost associated with data protection.

Consider a RAID configuration with five disks. Four of these disks hold data, and the fifth holds parity information. Parity requires 25 percent extra disk space compared to mirroring, which requires 100 percent extra disk space.

Disadvantage: Parity information is generated from data on the data disk. Therefore, parity is recalculated every time there is a change in data. This recalculation is time-consuming and affects the performance of the RAID controller.

3.3.4 RAID 0

In a RAID 0 configuration, data is striped across the HDDs in a RAID set. It utilizes the full storage capacity by distributing strips of data over multiple HDDs.

To read data, all the strips are put back together by the controller.

The stripe size is specified at a host level for software RAID and is vendor specific for hardware RAID.

Figure 3-5 shows RAID 0 on a storage array in which data is striped across 5 disks. When the number of drives in the array increases, performance improves because more data can be read or written simultaneously.

Adv: RAID 0 is used in applications that need high I/O throughput.

Disadv: RAID 0 does not provide data protection and availability in the event of drive failures.

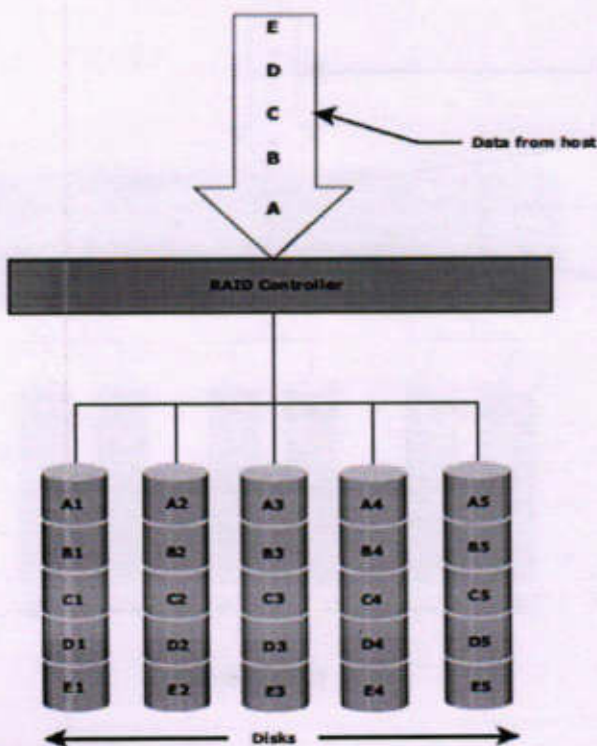


Figure 3-5: RAID 0

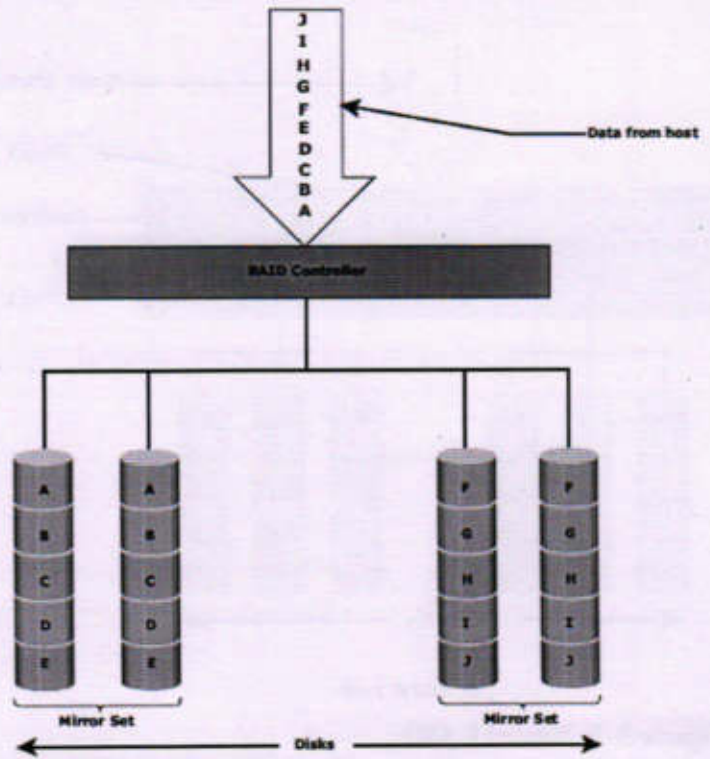


Figure 3-6: RAID 1

3.3.5 RAID 1

In a RAID 1 configuration, data is mirrored to improve fault tolerance (Figure 3-6). A RAID 1 group consists of at least two HDDs. As explained in mirroring, every write is written to both disks. In the event of disk failure, the impact on data recovery is the least among all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery and continuous operation. RAID 1 is suitable for applications that require high availability.

3.3.6 Nested RAID

Most data centers require data redundancy and performance from their RAID arrays.

RAID 0+1 and **RAID 1+0** combine the **performance benefits of RAID 0** with the **redundancy benefits of RAID 1**. They use striping and mirroring techniques and combine their benefits. These types of RAID require an even number of disks, the minimum being four (see Figure 3-7).

RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. Similarly, RAID 0+1 is also known as RAID 01 or RAID 0/1.

RAID 1+0 performs well for workloads that use small, random, write-intensive I/O.

Some applications that benefit from RAID 1+0 include the following:

1. High transaction rate Online Transaction Processing (OLTP)
2. Large messaging installations
3. Database applications that require high I/O rate, random access, and high availability.

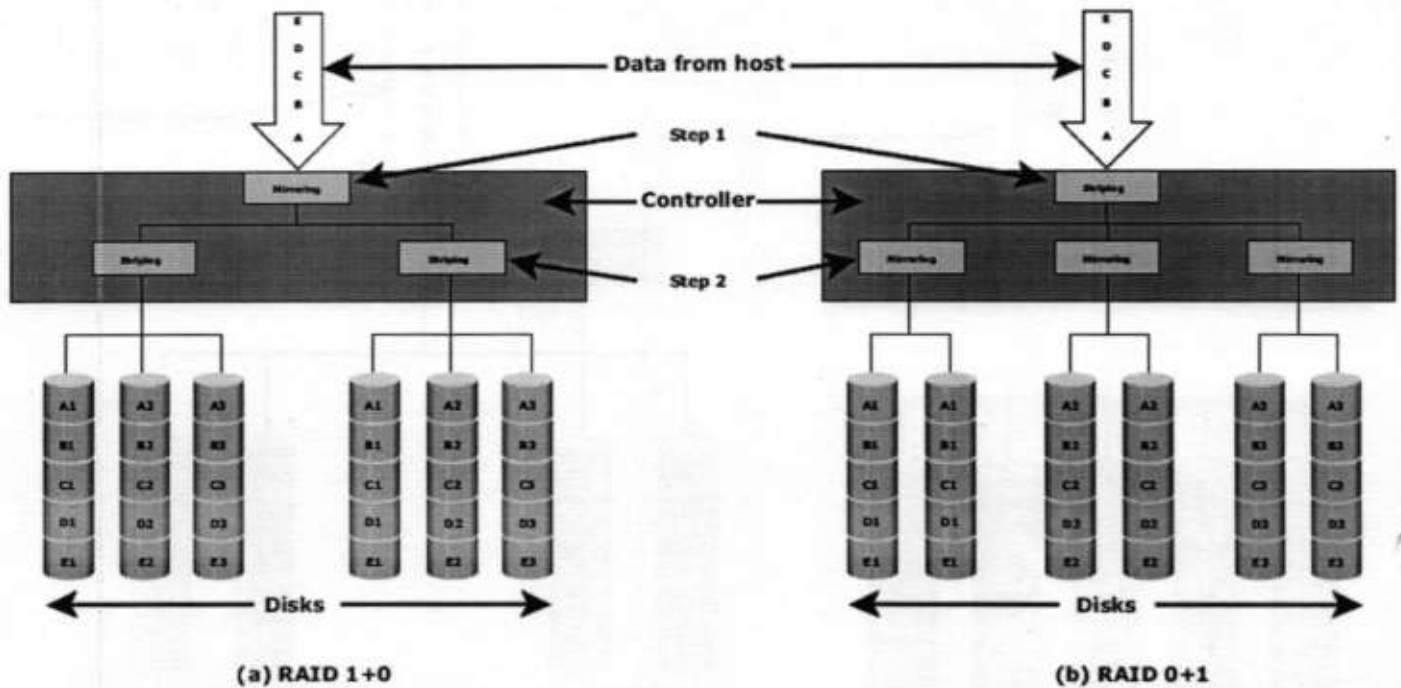


Figure 3-7: Nested RAID

A common misconception is that RAID 1+0 and RAID 0+1 are the same.

RAID 1+0 is also called striped mirror. The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of data are striped across multiple HDDs in a RAID set. When replacing a failed drive, only the mirror is rebuilt, i.e. the disk array controller uses the surviving drive in the mirrored pair for data recovery and continuous operation. Data from the surviving disk is copied to the replacement disk.

RAID 0+1 is also called mirrored stripe. The basic element of RAID 0+1 is a stripe. This means that the process of striping data across HDDs is performed initially and then the entire stripe is mirrored. If one drive fails, then the entire stripe is faulted. A rebuild operation copies the entire stripe, copying data from each disk in the healthy stripe to an equivalent disk in the failed stripe.

Disadv: This causes increased and unnecessary I/O load on the surviving disks and makes the RAID set more vulnerable to a second disk failure.

3.3.7 RAID 3

RAID 3 stripes data for high performance and uses parity for improved fault tolerance.

Parity information is stored on a dedicated drive so that data can be reconstructed if a drive fails.

For example, of five disks, four are used for data and one is used for parity. Therefore, the total disk space required is 1.25 times the size of the data disks.

RAID 3 **always** reads and writes complete stripes of data across all disks, as the drives operate in parallel. There are no partial writes that update one out of many strips. Figure 3-8 illustrates the RAID 3 implementation.

RAID 3 provides good bandwidth for the transfer of large volumes of data. RAID 3 is used in applications that involve large sequential data access, such as video streaming.

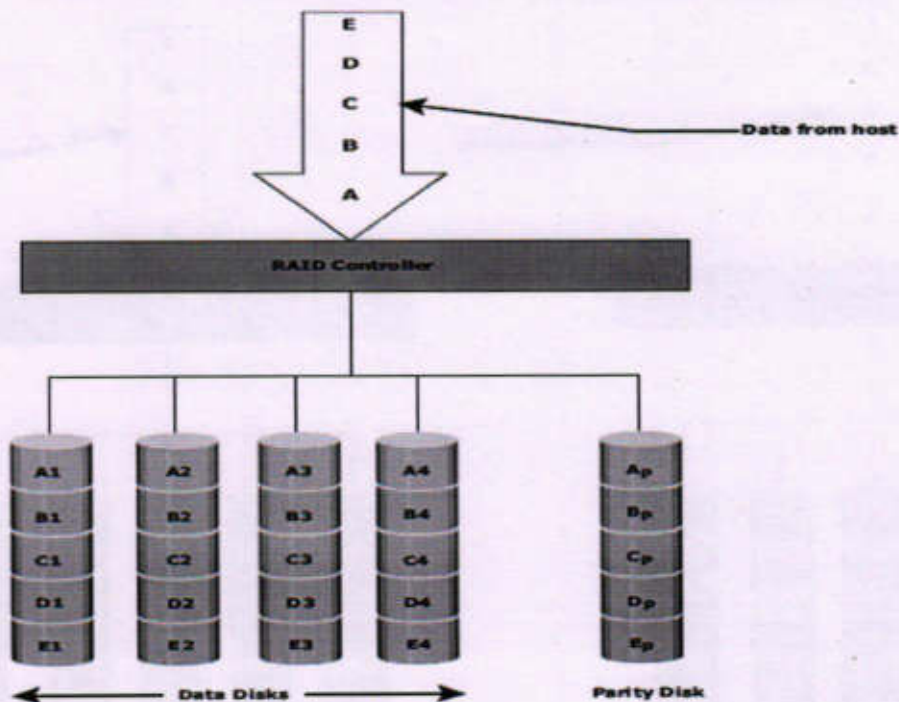


Figure 3-8: RAID 3

3.3.8 RAID 4

Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance (refer to Figure 3-8). Data is striped across all disks except the parity disk. Parity information is stored on a dedicated disk so that the data can be rebuilt if a drive fails. Striping is done at the block level.

Unlike RAID 3, data disks in RAID 4 can be accessed independently so that specific data elements can be read or written on single disk without read or write of an entire stripe. RAID 4 provides good read throughput and reasonable write throughput.

3.3.9 RAID 5

RAID 5 is a very versatile RAID implementation. It is similar to RAID 4 because it uses striping and the drives (strips) are independently accessible.

The difference between RAID 4 and RAID 5 is the **parity location**. In RAID 4, parity is written to a dedicated drive, creating a write bottleneck for the parity disk. In RAID 5, **parity is distributed** across all disks. The distribution of parity in RAID 5 overcomes the write bottleneck. Figure 3-9 illustrates the RAID 5 implementation.

RAID 5 is preferred for messaging, data mining, medium-performance media serving, and relational database management system (RDBMS) implementations in which database administrators (DBAs) optimize data access.

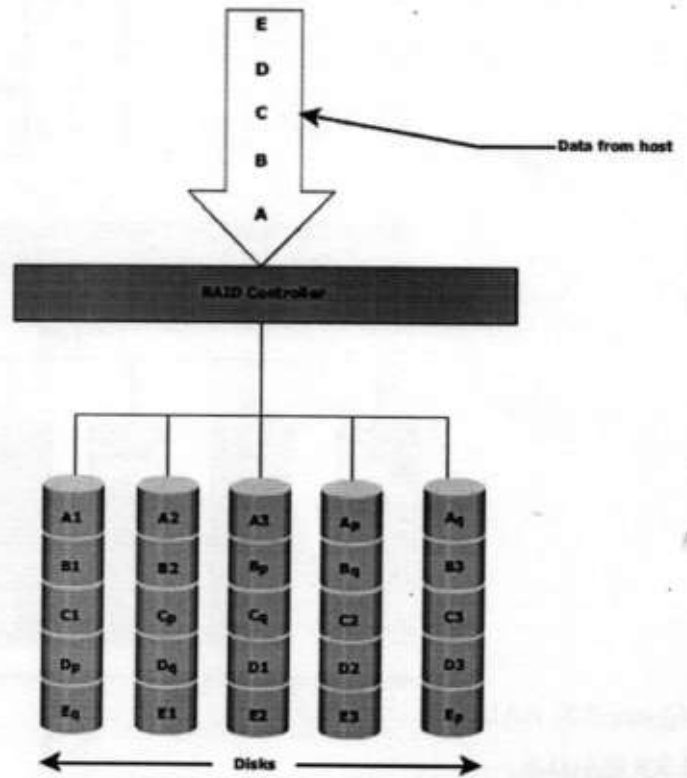
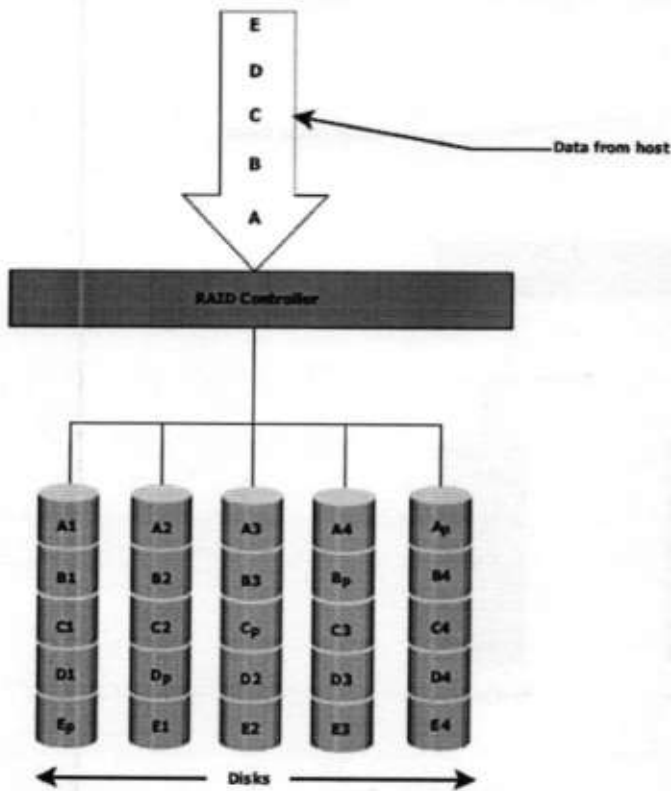


Figure 3-9: RAID 5

Figure 3-10: RAID 6

3.3.10 RAID 6

RAID 6 includes a second parity element to enable survival in the event of the failure of two disks in a RAID group (Figure 3-10). Therefore, a RAID 6 implementation requires at least four disks. RAID 6 distributes the parity across all the disks.

The write penalty in RAID 6 is more than that in RAID 5; therefore, RAID 5 writes perform better than RAID 6. The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.

3.4 RAID Comparison

Table 3-2: Comparison of Different RAID Types

RAID	MIN. DISKS	STORAGE EFFICIENCY %	COST	READ PERFORMANCE	WRITE PERFORMANCE	WRITE PENALTY
0	2	100	Low	Very good for both random and sequential read	Very good	No
1	2	50	High	Good. Better than a single disk.	Good. Slower than a single disk, as every write must be committed to all disks.	Moderate
3	3	$(n-1)*100/n$ where n= number of disks	Moderate	Good for random reads and very good for sequential reads.	Poor to fair for small random writes. Good for large, sequential writes.	High
4	3	$(n-1)*100/n$ where n= number of disks	Moderate	Very good for random reads. Good to very good for sequential writes.	Poor to fair for random writes. Fair to good for sequential writes.	High
5	3	$(n-1)*100/n$ where n= number of disks	Moderate	Very good for random reads. Good for sequential reads	Fair for random writes. Slower due to parity overhead. Fair to good for sequential writes.	High
6	4	$(n-2)*100/n$ where n= number of disks	Moderate but more than RAID 5	Very good for random reads. Good for sequential reads.	Good for small, random writes (has write penalty).	Very High
1+0 and 0+1	4	50	High	Very good	Good	Moderate

3.5 RAID Impact on Disk Performance

When choosing a RAID type, it is imperative to consider the impact to disk performance and application IOPS. In both **mirrored and parity RAID configurations**, every write operation translates into more I/O overhead for the disks which is referred to as **write penalty**.

Figure 3-11 illustrates a single write operation on RAID 5 that contains a group of five disks. Four of these disks are used for data and one is used for parity.

The parity (P) at the controller is calculated as follows:

$$E_p = E_1 + E_2 + E_3 + E_4 \text{ (XOR operations)}$$

Here, D1 to D4 is striped data across the RAID group of five disks.

Whenever controller performs a write I/O, parity must be computed by reading the old parity (E_p old) and the old data (E_4 old) from the disk, i.e. two read I/Os.

The new parity (E_p new) is computed as follows:

$$E_{p \text{ new}} = E_{p \text{ old}} - E_4 \text{ old} + E_4 \text{ new} \text{ (XOR operations)}$$

After computing the new parity, controller completes write I/O by writing the new data and new parity onto the disks, amounting to two write I/Os.

Therefore, controller performs two disk reads and two disk writes for every write operation, and the write penalty in RAID 5 implementations is 4.

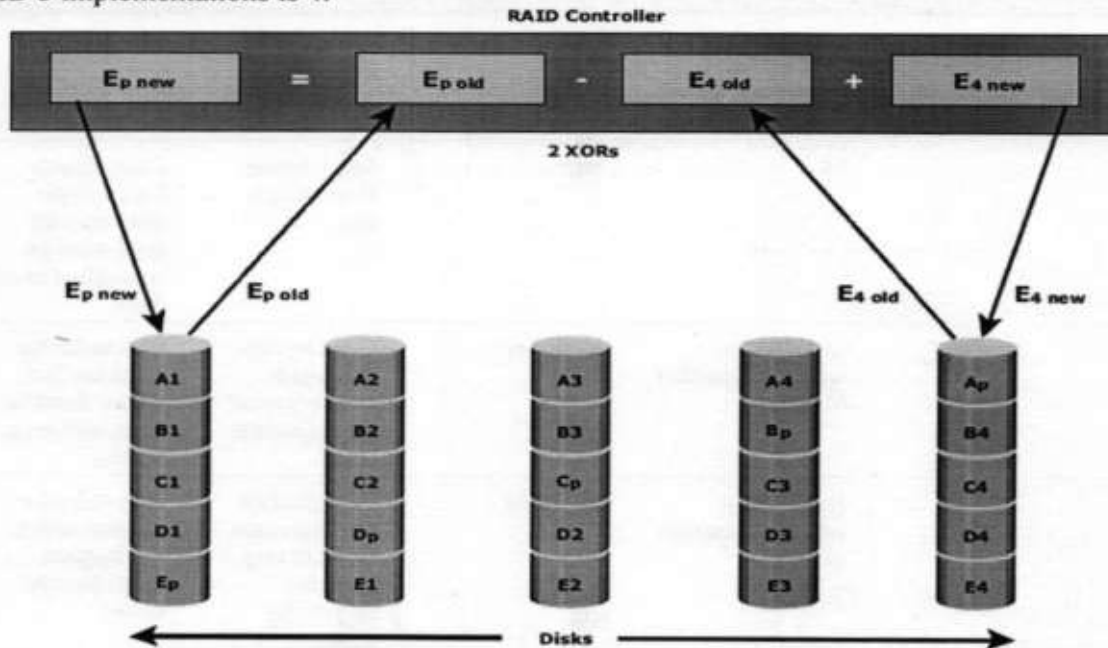


Figure 3-11: Write penalty in RAID 5

3.5.1 Application IOPS and RAID Configurations

When deciding the number of disks required for an application, it is important to consider the impact of RAID based on IOPS generated by the application. The total disk load should be computed by considering the type of RAID configuration and the ratio of read compared to write from the host.

The following example illustrates the method of computing the disk load in different types of RAID.

Consider an application that generates 5,200 IOPS, with 60 percent of them being reads.

The disk load in RAID 5 is calculated as follows:

$$\begin{aligned} \text{RAID 5 disk load} &= 0.6 \times 5,200 + 4 \times (0.4 \times 5,200) \text{ [because the write penalty for RAID 5 is 4]} \\ &= 3,120 + 4 \times 2,080 \\ &= 3,120 + 8,320 \\ &= 11,440 \text{ IOPS} \end{aligned}$$

The disk load in RAID 1 is calculated as follows:

$$\begin{aligned} \text{RAID 1 disk load} &= 0.6 \times 5,200 + 2 \times (0.4 \times 5,200) \text{ [because every write manifests as two writes to the disks]} \\ &= 3,120 + 2 \times 2,080 \\ &= 3,120 + 4,160 \\ &= 7,280 \text{ IOPS} \end{aligned}$$

Computed disk load determines the number of disks required for the application.

If in this example an HDD with a specification of a maximum 180 IOPS for the application needs to be used, the number of disks required to meet the workload for the RAID configuration as follows:

RAID 5: $11,440 / 180 = 64$ disks

RAID 1: $7,280 / 180 = 42$ disks (approximated to the nearest even number)

3.6 Hot Spares

A *hot spare* refers to a spare HDD in a RAID array that temporarily replaces a failed HDD of a RAID set. A hot spare takes the identity of the failed HDD in the array.

One of the following methods of data recovery is performed depending on the RAID implementation:

1. If parity RAID is used, then the data is rebuilt onto the hot spare from the parity and the data on the surviving HDDs in the RAID set.
2. If mirroring is used, then the data from the surviving mirror is used to copy the data.

When the failed HDD is replaced with a new HDD, one of the following takes place:

1. The hot spare replaces the new HDD permanently. This means that it is no longer a hot spare, and a new hot spare must be configured on the array.
2. When a new HDD is added to the system, data from the hot spare is copied to it. The hot spare returns to its idle state, ready to replace the next failed drive.

A hot spare should be large enough to accommodate data from a failed drive. System can implement multiple hot spares to improve data availability.

A hot spare can be configured as *automatic* or *user initiated*, which specifies how it will be used in the event of disk failure.

In an *automatic configuration*, when the recoverable error rates for a disk exceed a predetermined threshold, the disk subsystem tries to copy data from the failing disk to the hot spare automatically. If this task is completed before the damaged disk fails, then the subsystem switches to the hot spare and marks the failing disk as unusable.

Chapter 4 - Intelligent Storage System

Introduction

Business-critical applications require high levels of performance, availability, security, and scalability. A hard disk drive is a core element of storage that governs the performance of any storage system. RAID technology made an important contribution to enhancing storage performance and reliability, but hard disk drives even with a RAID implementation could not meet performance requirements of today's applications.

With advancements in technology, a new breed of storage solutions known as an *intelligent storage system* has evolved. The intelligent storage systems detailed in this chapter are the feature-rich RAID arrays that provide highly optimized I/O processing capabilities. These arrays have an operating environment that controls the management, allocation, and utilization of storage resources.

4.1 Components of an Intelligent Storage System

An intelligent storage system consists of four key components: *front end*, *cache*, *back end*, and *physical disks*. Figure 4-1 illustrates these components and their interconnections. An I/O request received from the host at the front-end port is processed through cache and the back end, to enable storage and retrieval of data from the physical disk. A read request can be serviced directly from cache if the requested data is found in cache.

4.1.1 Front End

The front end provides the interface between the storage system and the host. It consists of two components: **front-end ports** and **front-end controllers**.

1. **Front-end ports:** enable hosts to connect to the intelligent storage system. Each front-end port has processing logic that executes the appropriate transport protocol, such as SCSI, Fibre Channel, or iSCSI, for storage connections. Redundant ports are provided on the front end for high availability.
2. **Front-end controllers:** route data to and from cache via the internal data bus.

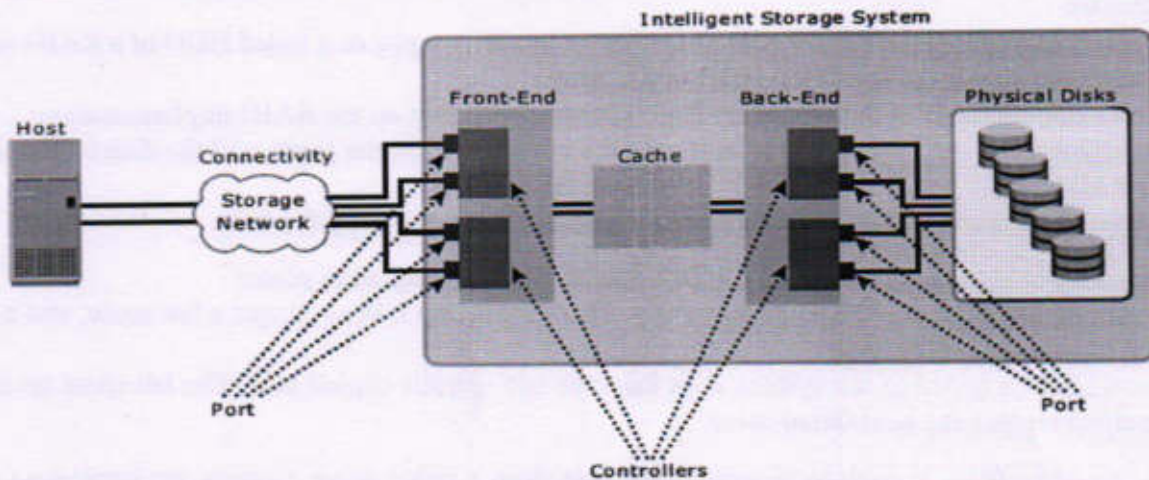


Figure 4-1: Components of an intelligent storage system

Front-End Command Queuing: Command queuing is a technique implemented on front-end controllers. It determines the execution order of received commands and can reduce unnecessary drive head movements and improve disk performance.

When a command is received for execution, the command queuing algorithms assigns a tag that defines a sequence in which commands should be executed. With command queuing, multiple commands can be executed concurrently based on the organization of data on the disk, regardless of the order in which the commands were received.

The most commonly used command queuing algorithms are as follows:

1. First In First Out (FIFO): This is the default algorithm where commands are executed in the order in which they are received (Figure 4-2 (a)). There is no reordering of requests for optimization; therefore, it is inefficient in terms of performance.

2. Seek Time Optimization: Commands are executed based on optimizing read/write head movements, which may result in reordering of commands. Without seek time optimization, the commands are executed in the order they are received. For example, as shown in Figure 4-2(a), the commands are executed in the order A, B, C and D. The radial movement required by the head to execute C immediately after A. With seek time optimization, the command execution sequence would be A, C, B and D, as shown in Figure 4-2(b).

3. Access Time Optimization: Commands are executed based on the combination of seek time optimization and an analysis of rotational latency for optimal performance.

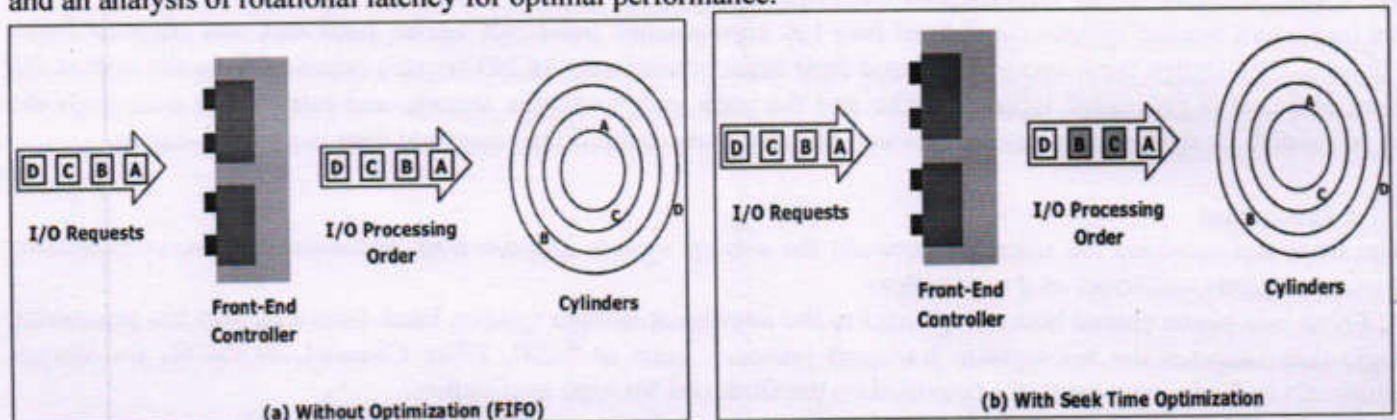


Figure 4-2: Front-end command queuing

4.1.2 Cache

Cache is an important component that enhances the I/O performance in an intelligent storage system. Cache is semiconductor memory where data is placed temporarily to reduce the time required to service I/O requests from the host.

Cache improves storage system performance by isolating hosts from the mechanical delays associated with physical disks, which are the slowest components of an intelligent storage system. Accessing data from cache takes less than a millisecond. Write data is placed in cache and then written to disk. After the data is securely placed in cache, the host is acknowledged immediately.

Structure of Cache: Cache is organized into pages or slots, which is the smallest unit of cache allocation. The size of a cache page is configured according to the application I/O size. Cache consists of the *data store* and *tag RAM*. The data store holds the data while tag RAM tracks the location of the data in the data store (see Figure 4-3) and in disk.

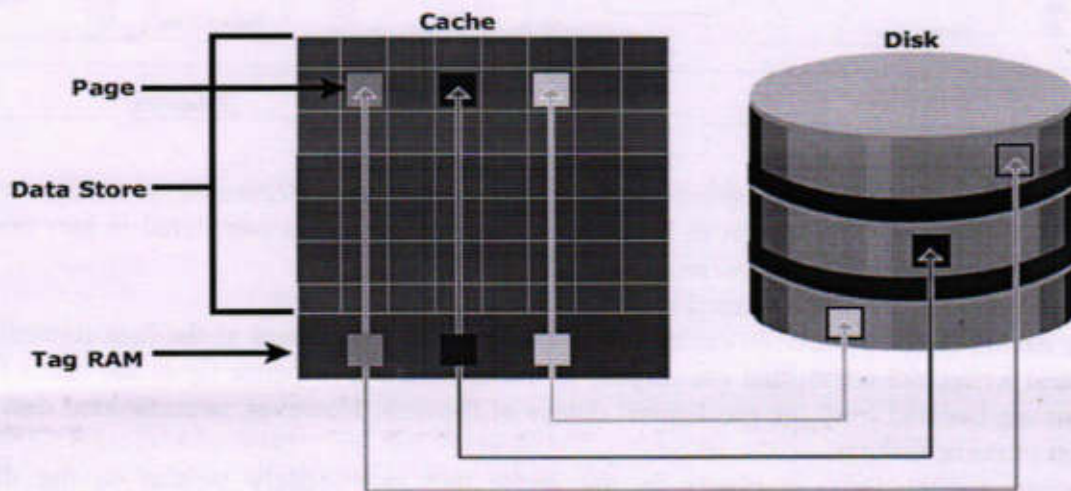


Figure 4-3: Structure of cache

Entries in tag RAM indicate where data is found in cache and where the data belongs on the disk. Tag RAM includes a *dirty bit* flag, which indicates whether the data in cache has been committed to the disk or not. It also contains time-based information, such as the time of last access, which is used to identify cached information that has not been accessed for a long period and may be freed up.

Read Operation with Cache: When a host issues a read request, the front-end controller accesses the tag RAM to determine whether the required data is available in cache.

If the requested data is found in the cache, it is called a **read cache hit** or **read hit** and data is sent directly to the host, without any disk operation (see Figure 4-4[a]). This provides a fast response time to the host (about a millisecond).

If the requested data is not found in cache, it is called a **read cache miss** or **read miss** and the data must be read from the disk (see Figure 4-4[b]).

The back-end controller accesses the appropriate disk and retrieves the requested data. Data is then placed in cache and is finally sent to the host through the front-end controller. Cache misses increase I/O response time.

A **pre-fetch**, or **read-ahead**, algorithm is used when read requests are sequential.

In a sequential read request, a contiguous set of associated blocks is retrieved. Several other blocks that have not yet been requested by the host can be read from the disk and placed into cache in advance. When the host subsequently requests these blocks, the read operations will be read hits. This process significantly improves the response time experienced by the host.

The intelligent storage system offers fixed and variable pre-fetch sizes.

In *fixed pre-fetch*, the intelligent storage system pre-fetches a fixed amount of data. It is most suitable when I/O sizes are uniform.

In *variable pre-fetch*, the storage system pre-fetches an amount of data in multiples of the size of the host request.

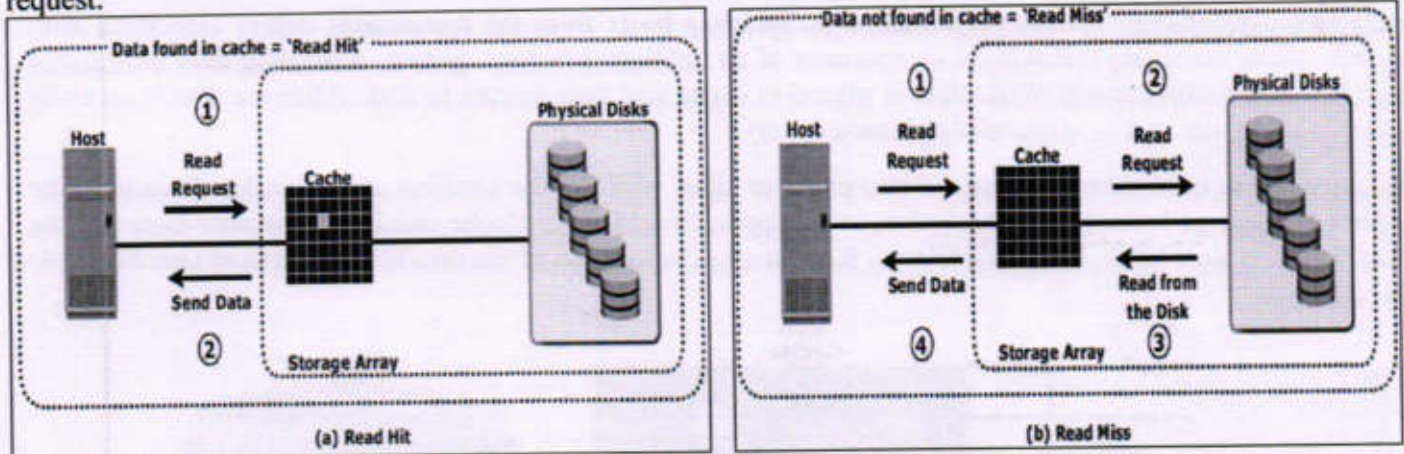


Figure 4-4: Read hit and read miss

Write Operation with Cache: Write operations with cache provide performance advantages over writing directly to disks. When an I/O is written to cache and acknowledged, it is completed in less time (from the host's perspective) than it would take to write directly to disk.

A write operation with cache is implemented in the following ways:

- 1. Write-back cache:** Data is placed in cache and an acknowledgment is sent to the host immediately. Later, data from several writes are committed (de-staged) to the disk. Write response times are much faster, as the write operations are isolated from the mechanical delays of the disk. However, uncommitted data is at risk of loss in the event of cache failures.
- 2. Write-through cache:** Data is placed in the cache and immediately written to the disk, and an acknowledgment is sent to the host. Because data is committed to disk as it arrives, the risks of data loss are low but write response time is longer because of the disk operations.

Cache Implementation: Cache can be implemented as either dedicated cache or global cache. With dedicated cache, separate sets of memory locations are reserved for reads and writes. In global cache, both reads and writes can use any of the available memory addresses. Cache management is more efficient in a global cache implementation, as only one global set of addresses has to be managed.

Cache Management: Cache is a finite and expensive resource that needs proper management. When all cache pages are filled, some pages have to be freed up to accommodate new data and avoid performance degradation. Various cache management algorithms are implemented in intelligent storage systems:

- 1. Least Recently Used (LRU):** An algorithm that continuously monitors data access in cache and identifies the cache pages that have not been accessed for a long time. LRU either frees up these pages or marks them for reuse. This algorithm is based on the assumption that data which hasn't been accessed for a while will not be requested by the host.
- 2. Most Recently Used (MRU):** In MRU, the pages that have been accessed most recently are freed up or marked for reuse. This algorithm is based on the assumption that recently accessed data may not be required for a while.

As cache fills, storage system must take action to **flush dirty pages** (data written into the cache but not yet written to the disk) to manage availability. Flushing is the process of committing data from cache to the disk. On the basis of the I/O access rate and pattern, high and low levels called *watermarks* are set in cache to manage the flushing process.

High watermark (HWM) is cache utilization level at which the storage system starts highspeed flushing of cache data.

Low watermark (LWM) is the point at which the storage system stops the high-speed or forced flushing and returns to idle flush behavior.

The cache utilization level, as shown in Figure 4-5, drives the mode of flushing to be used:

1. **Idle flushing:** Occurs continuously, at a modest rate, when the cache utilization level is between the high and low watermark.
2. **High watermark flushing:** Activated when cache utilization hits the high watermark. The storage system dedicates some additional resources to flushing. This type of flushing has minimal impact on host I/O processing.
3. **Forced flushing:** Occurs in the event of a large I/O burst when cache reaches 100 percent of its capacity, which significantly affects the I/O response time. In forced flushing, dirty pages are forcibly flushed to disk.

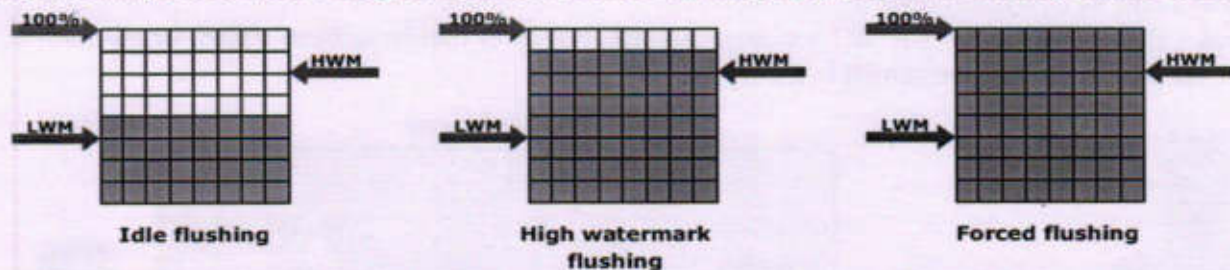


Figure 4-5: Types of flushing

Cache Data Protection: Cache is volatile memory, so a power failure or any kind of cache failure will cause the loss of data not yet committed to the disk. This risk of losing uncommitted data held in cache can be mitigated using *cache mirroring* and *cache vaulting*:

1. **Cache mirroring:** Each write to cache is held in two different memory locations on two independent memory cards. In the event of a cache failure, the write data will still be safe in the mirrored location and can be committed to the disk. Reads are staged from the disk to the cache; therefore, in the event of a cache failure, the data can still be accessed from the disk. As only writes are mirrored, this method results in better utilization of the available cache. In cache mirroring approaches, the problem of maintaining *cache coherency* is introduced. Cache coherency means data in two different cache locations must be identical at all times. It is the responsibility of the array operating environment to ensure coherency.

2. **Cache vaulting:** In this case, Storage vendors use a set of physical disks to dump the contents of cache during power failure. This is called cache vaulting and the disks are called vault drives. When power is restored, data from these disks is written back to write cache and then written to the intended disks.

Cache is exposed to the risk of uncommitted data loss due to power failure. This problem can be addressed in various ways: powering the memory with a battery until AC power is restored or using battery power to write the cache content to the disk. In the event of extended power failure, using batteries is not a viable option because in intelligent storage systems, large amounts of data may need to be committed to numerous disks.

4.1.3 Back End

The *back end* provides an interface between cache and the physical disks. It consists of two components: **back-end ports** and **back-end controllers**. The back end controls data transfers between cache and the physical disks. From cache, data is sent to the back end and then routed to the destination disk. Physical disks are connected to ports on the back end. The back end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage. The algorithms implemented on back-end controllers provide error detection and correction, along with RAID functionality.

4.1.4 Physical Disk

A physical disk stores data persistently. Disks are connected to the back-end with either SCSI or a Fibre Channel interface (discussed in subsequent chapters). An intelligent storage system enables the use of a mixture of SCSI or Fibre Channel drives and IDE/ATA drives.

Logical Unit Number: Physical drives or groups of RAID protected drives can be logically split into volumes known as logical volumes, commonly referred to as **Logical Unit Numbers (LUNs)**. The use of LUNs improves disk utilization.

For example, without the use of LUNs, a host requiring only 200 GB could be allocated an entire 1TB physical disk. Using LUNs, only the required 200 GB would be allocated to the host, allowing the remaining 800 GB to be allocated to other hosts.

In the case of RAID protected drives, these logical units are slices of RAID sets and are spread across all the physical disks belonging to that set. The logical units can also be seen as a logical partition of a RAID set that is presented to a host as a physical disk.

For example, Figure 4-6 shows a RAID set consisting of five disks that have been sliced, or partitioned, into several LUNs. LUNs 0 and 1 are shown in the figure.

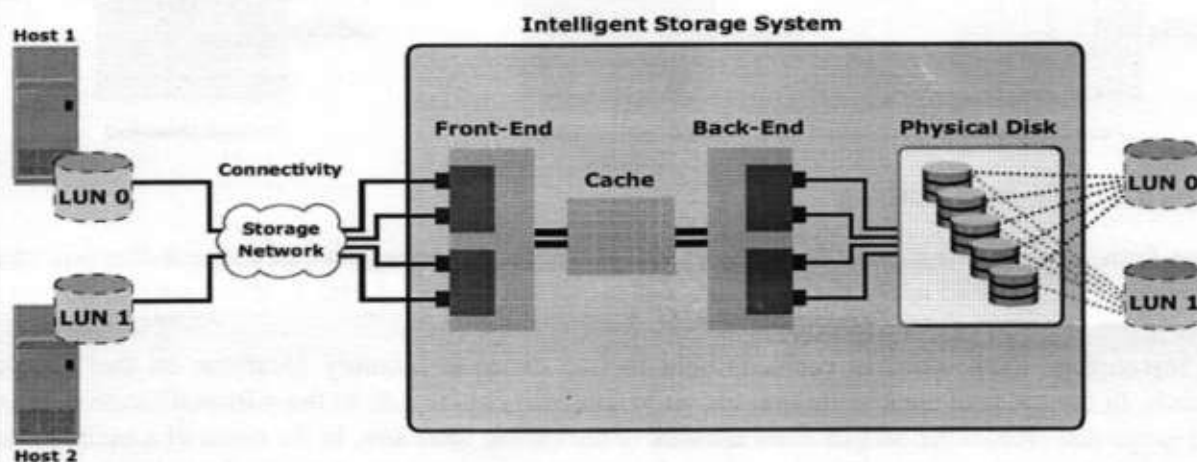


Figure 4-6: Logical unit number

LUNs 0 and 1 are presented to hosts 1 and 2, respectively, as physical volumes for storing and retrieving data. Usable capacity of the physical volumes is determined by the RAID type of the RAID set.

The capacity of a LUN can be expanded by aggregating other LUNs with it. The result of this aggregation is a larger capacity LUN, known as a *meta-LUN*.

LUN Masking: LUN masking is a process that provides data access control by defining which LUNs a host can access. LUN masking function is typically implemented at the front end controller. This ensures that volume access by servers is controlled appropriately, preventing unauthorized or accidental use in a distributed environment.

For example, consider a storage array with two LUNs that store data of the sales and finance departments. Without LUN masking, both departments can easily see and modify each other's data, posing a high risk to data integrity and security. With LUN masking, LUNs are accessible only to the designated hosts.

4.2 Intelligent Storage Array

Intelligent storage systems generally fall into one of the following two categories:

1. High-end storage systems
2. Midrange storage systems

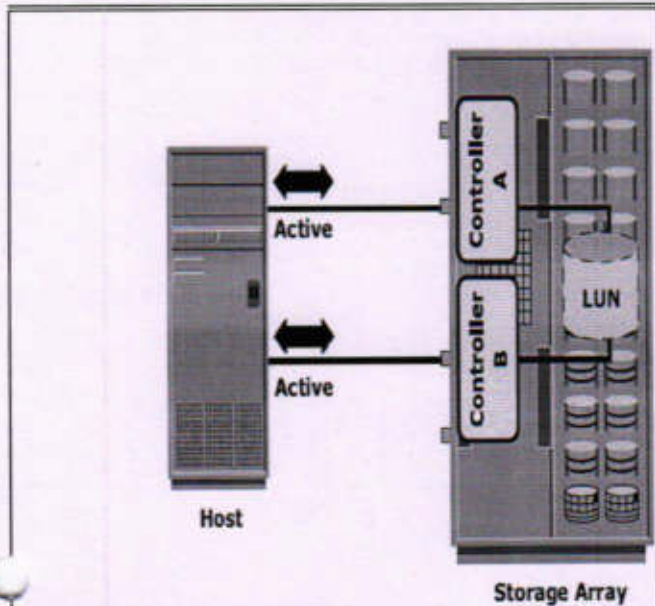


Figure 4-7: Active-active configuration

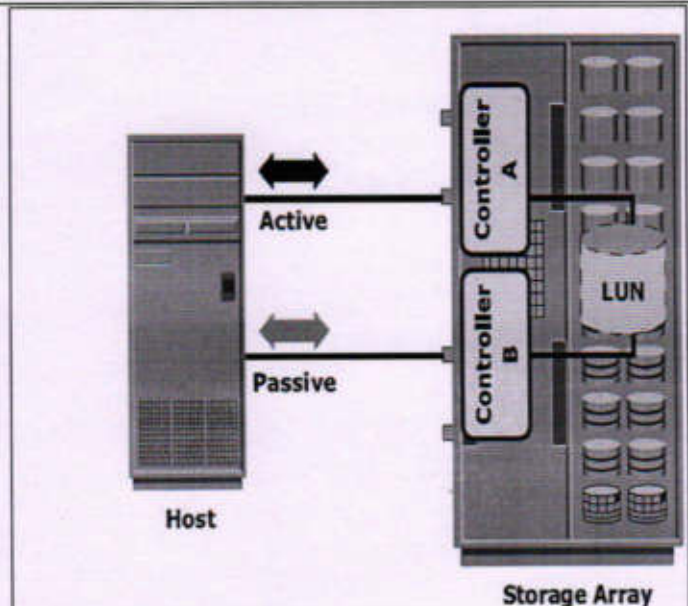


Figure 4-8: Active-passive configuration

4.2.1 High-end Storage Systems

High-end storage systems, referred to as *active-active arrays*, are aimed at large enterprises for centralizing corporate data. These arrays are designed with a large number of controllers and cache memory. An active-active array implies that the host can perform I/Os to its LUNs across any of the available paths (see Figure 4-7).

To address the enterprise storage needs, these arrays provide the following capabilities:

1. Large storage capacity
2. Large amounts of cache to service host I/Os optimally
3. Fault tolerance architecture to improve data availability
4. Connectivity to mainframe computers and open systems hosts
5. Availability of multiple front-end ports and interface protocols to serve a large number of hosts
6. Availability of multiple back-end Fibre Channel or SCSI RAID controllers to manage disk processing
7. Scalability to support increased connectivity, performance, and storage capacity requirements
8. Ability to handle large amounts of concurrent I/Os from a number of servers and applications
9. Support for array-based local and remote replication

4.2.2 Midrange Storage System

Midrange storage systems are also referred to as *active-passive arrays* and they are best suited for small- and medium-sized enterprises. In an active-passive array, a host can perform I/Os to a LUN only through the paths to the owning controller of that LUN. These paths are called *active paths*. The other paths are passive with respect to this LUN.

As shown in Figure 4-8, the host can perform reads or writes to the LUN only through the path to controller A, as controller A is the owner of that LUN. The path to controller B remains passive and no I/O activity is performed through this path.

Midrange arrays are designed to meet the requirements of small and medium enterprises; therefore, they host less storage capacity and global cache than active-active arrays. There are also fewer front-end ports for connection to servers. However, they ensure high redundancy and high performance for applications with predictable workloads. They also support array-based local and remote replication.

Module – 2: Storage Networking Technologies and Virtualization

Introduction: *Direct-Attached Storage (DAS)* is an architecture where storage connects directly to servers. Applications access data from DAS using block-level access protocols. The internal HDD of a host, tape libraries, and directly connected external HDD packs are some examples of DAS.

Types of DAS

DAS is classified as internal or external, based on the location of the storage device with respect to the host.

1 Internal DAS: In *internal DAS* architectures, the storage device is internally connected to the host by a serial or parallel bus. The physical bus has distance limitations and can only be sustained over a shorter distance for high-speed connectivity. Supports limited number of devices, and they occupy a large amount of space inside the host, making maintenance of other components difficult.

2 External DAS: In *external DAS* architectures, the server connects directly to the external storage device (see Figure 5-1). In most cases, communication between the host and the storage device takes place over SCSI or FC protocol. Compared to internal DAS, an external DAS overcomes the distance and device count limitations and provides centralized management of storage devices.

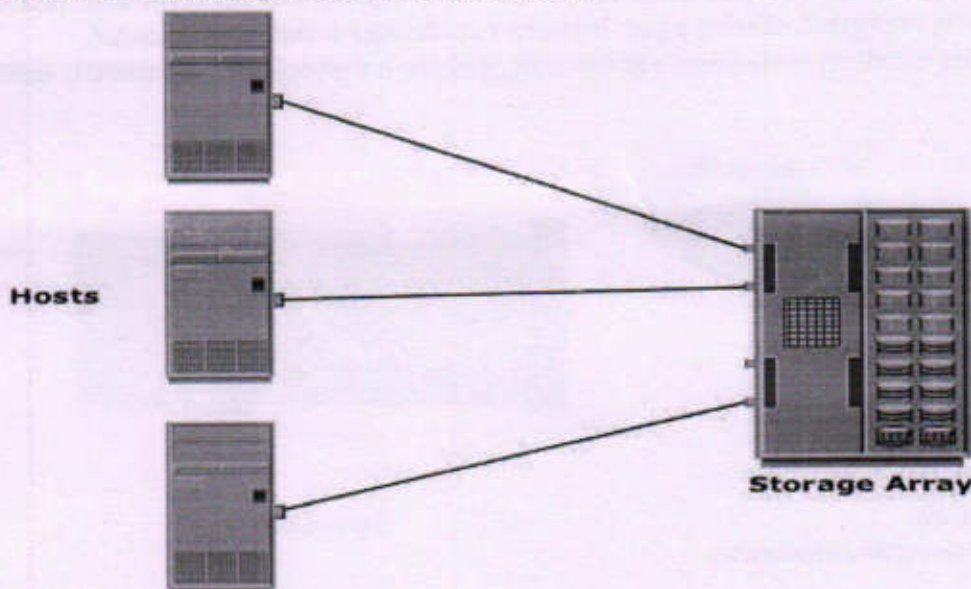


Figure 5-1: External DAS architecture

DAS Benefits and Limitations

Benefits

1. DAS requires a relatively lower initial investment than storage networking.
2. DAS configuration is simple and can be deployed easily and rapidly.
3. Setup is managed using host-based tools, such as the host OS, which makes storage management tasks easy for small and medium enterprises.
4. DAS is the simplest solution when compared to other storage networking models and requires fewer management tasks, and less hardware and software elements to set up and operate.

Limitations

1. DAS does not scale well.
2. A storage device has a limited number of ports, which restricts the number of hosts that can directly connect to the storage.
3. Limited bandwidth in DAS restricts the available I/O processing capability.

4. When capacities are being reached, the service availability may be compromised, and this has a ripple effect on the performance of all hosts attached to that specific device or array.
5. DAS does not make optimal use of resources due to its limited ability to share front end ports.
6. In DAS environments, unused resources cannot be easily re-allocated, resulting in islands of over-utilized and under-utilized storage pools.

Disk Drive Interfaces

The host and the storage device in DAS communicate with each other by using predefined protocols such as IDE/ATA, SATA, SAS, SCSI, and FC. These protocols are implemented on the HDD controller. Therefore, a storage device is also known by the name of the protocol it supports. This section describes each of these storage devices in detail.

1 IDE/ATA

An Integrated Device Electronics/Advanced Technology Attachment (IDE/ATA) disk supports the IDE protocol. The ATA component is the interface for connecting storage devices, such as CD-ROMs, floppy disk drives, and HDDs, to the motherboard. IDE/ATA has a variety of standards and names, such as ATA, ATA/ATAPI, EIDE, ATA-2, Fast ATA, ATA-3, Ultra ATA, and Ultra DMA. The latest version of ATA—Ultra DMA/133—supports a throughput of 133 MB per second.

In a master-slave configuration, an ATA interface supports two storage devices per connector. However, if the performance of the drive is important, sharing a port between two devices is not recommended.

An IDE/ATA disk offers excellent performance at low cost, making it a popular and commonly used hard disk.

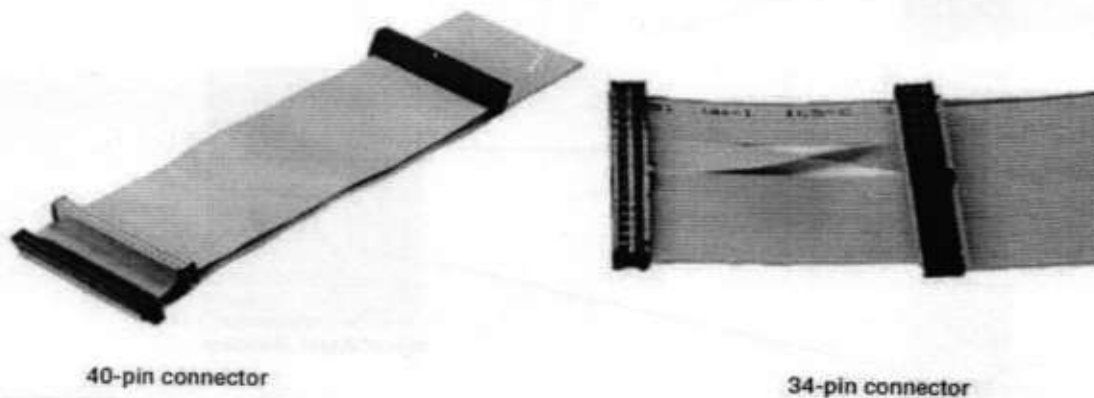


Figure 5-2: Common IDE connectors

2. SATA

A SATA (Serial ATA) is a serial version of the IDE/ATA specification. SATA is a disk-interface technology that was developed by a group of the industry's leading vendors with the aim of replacing parallel ATA.

A SATA provides point-to-point connectivity up to a distance of one meter and enables data transfer at a speed of 150 MB/s. Enhancements to the SATA have increased the data transfer speed up to 600 MB/s.

A SATA bus directly connects each storage device to the host through a dedicated link, making use of *low-voltage differential signaling (LVDS)*. LVDS is an electrical signaling system that can provide high-speed connectivity over low-cost, twisted-pair copper cables. For data transfer, a SATA bus uses LVDS with a voltage of 250 mV.

A SATA bus uses a small 7-pin connector and a thin cable for connectivity. A SATA port uses 4 signal pins, which improves its pin efficiency.

SATA devices are *hot-pluggable*, which means that they can be connected or removed while the host is up and running. A SATA port permits single-device connectivity. Connecting multiple SATA drives to a host requires multiple ports to be present on the host.

3. Parallel SCSI

SCSI is available in a variety of interfaces. Parallel SCSI (referred to as SCSI) is one of the oldest and most popular forms of storage interface used in hosts. SCSI is a set of standards used for connecting a peripheral device to a computer and transferring data between them.

SCSI has undergone rapid revisions, resulting in continuous performance improvements. The oldest SCSI variant, called SCSI-1 provided data transfer rate of 5 MB/s; SCSI Ultra 320 provides data transfer speeds of 320 MB/s. Other variants of SCSI and transfer speeds are listed in Table 5-2.

Table 5-1: Comparison of IDE/ATA with SCSI

FEATURE	IDE/ATA	SCSI
Speed	100, 133, 150 MB/s	320 MB/s
Connectivity	Internal	Internal and external
Cost	Low	Moderate to high
Hot-pluggable	No	Yes
Performance	Moderate to low	High
Ease of configuration	High	Low to moderate
Maximum number of devices supported	2	16

Introduction to Parallel SCSI

Shugart Associates and NCR developed a system interface in 1981 and named it Shugart Associates System Interface (SASI). SASI was developed to build a proprietary, high-performance standard primarily for use by these two companies.

However, to increase the acceptance of SASI in the industry, the standard was updated to a more robust interface and renamed SCSI. In 1986, the American National Standards Institution (ANSI) acknowledged the new SCSI as an industry standard.

SCSI, first developed for hard disks, is often compared to IDE/ATA. SCSI offers improved performance and expandability and compatibility options, making it suitable for high-end computers.

However, the high cost associated with SCSI limits its popularity among home or business desktop users.

Evolution of SCSI

1. SCSI-1

SCSI-1, renamed to distinguish it from other SCSI versions, is the original standard that the ANSI approved. SCSI-1 defined the basics of the first SCSI bus, including cable length, signaling characteristics, commands, and transfer modes. SCSI-1 devices supported only single-ended transmission and *passive termination*. SCSI-1 used a narrow 8-bit bus, which offered a maximum data transfer rate of 5 MB/s. SCSI-1 implementations resulted in incompatible devices and several subsets of standards. Due to these issues, work on improving the SCSI-1 standard began in 1985, a year before its formal approval.

2. SCSI-2

To control the various problems caused by the nonstandard implementation of the original SCSI, a working paper was created to define a set of standard commands for a SCSI device. This set of standards, called the common command set (CCS), formed the basis of the SCSI-2 standard. SCSI-2 was focused on improving performance, enhancing reliability, and adding additional features to the SCSI-1 interface, in addition to standardizing and formalizing the SCSI commands. The ANSI withdrew the SCSI-1 standard and, in 1994, approved SCSI-2 as one large document: X3.131-1994. The transition from SCSI-1 to SCSI-2 did not raise much concern because SCSI-2 offered backward compatibility with SCSI-1.

3. SCSI-3

In 1993, work began on developing the next version of the SCSI standard, SCSI-3. Unlike SCSI-2, the SCSI-3 standard document is comprised different but related standards, rather than one large document.

SCSI Interfaces

SCSI interface (Table 5-2 lists some of the available parallel SCSI interfaces). The SCSI design is now making a transition into Serial Attached SCSI (SAS), which is based on a serial point-to-point design, while retaining the other aspects of the SCSI technology.

Table 5-2: SCSI Interfaces

INTERFACE	STANDARD	BUS WIDTH	CLOCK SPEED	MAX THROUGHPUT	MAX DEVICES
SCSI-1	SCSI-1	8	5 MHz	5 MB/s	8
Fast SCSI	SCSI-2	8	10 MHz	10 MB/s	8
Fast Wide SCSI	SCSI-2; SCSI-3 SPI	16	10 MHz	20 MB/s	16
Ultra SCSI	SCSI-3 SPI	8	20 MHz	20 MB/s	8
Ultra Wide SCSI	SCSI-3 SPI	16	20 MHz	40 MB/s	16
Ultra2 SCSI	SCSI-3 SPI-2	8	40 MHz	40 MB/s	8
Ultra2 Wide SCSI	SCSI-3 SPI-2	16	40 MHz	80 MB/s	16
Ultra3 SCSI	SCSI-3 SPI-3	16	40 MHz DDR	160 MB/s	16
Ultra320 SCSI	SCSI-3 SPI-4	16	80 MHz DDR	320 MB/s	16
Ultra640 SCSI	SCSI-3 SPI-5	16	160 MHz DDR	640 MB/s	16

SCSI-3 Architecture

The SCSI-3 architecture defines and categorizes various SCSI-3 standards and requirements for SCSI-3 implementations. The SCSI-3 architecture was approved and published as the standard X.3.270-1996 by the ANSI. This architecture helps developers, hardware designers, and users to understand and effectively utilize SCSI.

The three major components of a SCSI architectural model are as follows:

- 1. SCSI-3 command protocol:** This consists of primary commands that are common to all devices as well as device-specific commands that are unique to a given class of devices.
- 2. Transport layer protocols:** These are a standard set of rules by which devices communicate and share information.
- 3. Physical layer interconnects:** These are interface details such as electrical signaling methods and data transfer modes.

Common access methods are the ANSI software interfaces for SCSI devices.

Figure 5-3 shows the SCSI-3 standards architecture with interrelated groups of other standards within SCSI-3.

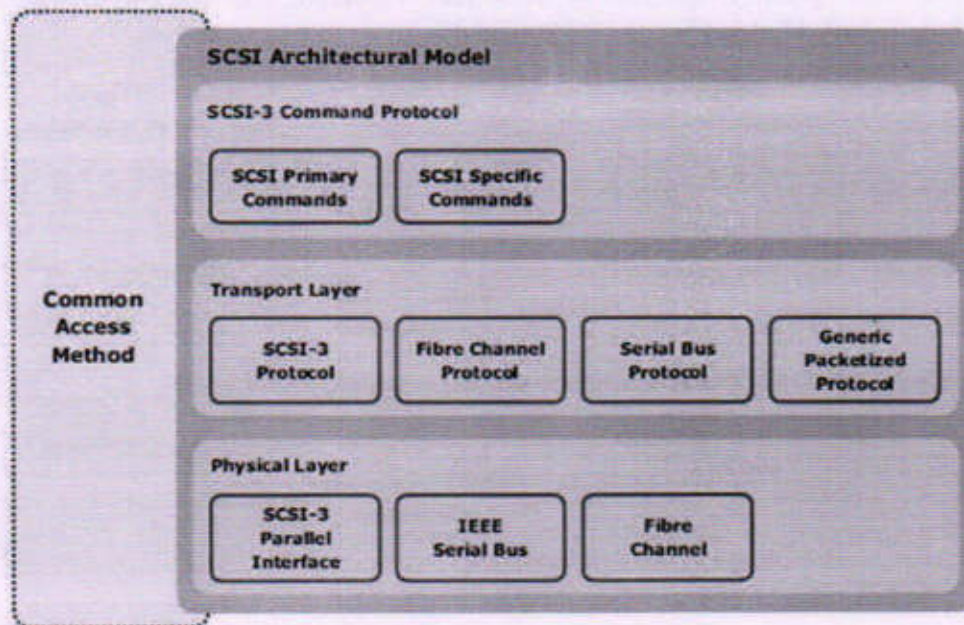


Figure 5-3: SCSI-3 standards architecture

SCSI-3 Client-Server Model

SCSI-3 architecture derives its base from the client-server relationship, in which a client directs a service request to a server, which then fulfills the client's request. In a SCSI environment, an initiator-target concept represents the client server model. In a SCSI-3 client-server model, a particular SCSI device acts as a SCSI target device, a SCSI initiator device, or a SCSI target/initiator device.

Each device performs the following functions:

- 1. SCSI initiator device:** Issues a command to the SCSI target device, to perform a task. A SCSI host adaptor is an example of an initiator.
- 2. SCSI target device:** Executes commands to perform the task received from a SCSI initiator. Typically a SCSI peripheral device acts as a target device. However, in certain implementations, the host adaptor can also be a target device.

Figure 5-4 displays the SCSI-3 client-server model, in which a SCSI initiator, or a client, sends a request to a SCSI target, or a server. The target performs the tasks requested and sends the output to the initiator, using the protocol service interface.

A SCSI target device contains one or more logical units. A logical unit is an object that implements one of the device functional models as described in the SCSI command standards. A logical unit has two components, a *device server* and a *task manager*, as shown in Figure 5-4.

The logical unit processes the commands sent by a SCSI initiator.

The device server addresses client requests, and the task manager performs management functions.

The SCSI initiator device is comprised of an application client and task management function, which initiates device service and task management requests.

Each device service request contains a *Command Descriptor Block (CDB)*. The CDB defines the command to be executed and lists command-specific inputs and other parameters specifying how to process the command.

The SCSI devices are identified by a specific number called a SCSI ID. In narrow SCSI (bus width=8), the devices are numbered 0 through 7; in wide (bus width=16) SCSI, the devices are numbered 0 through 15. These ID numbers set the device priorities on the SCSI bus. In narrow SCSI, 7 has the highest priority and 0 has the lowest priority. In wide SCSI, the device IDs from 8 to 15 have the highest priority, but the entire sequence of

wide SCSI IDs has lower priority than narrow SCSI IDs. Therefore, the overall priority sequence for a wide SCSI is 7, 6, 5, 4, 3, 2, 1, 0, 15, 14, 13, 12, 11, 10, 9, and 8.

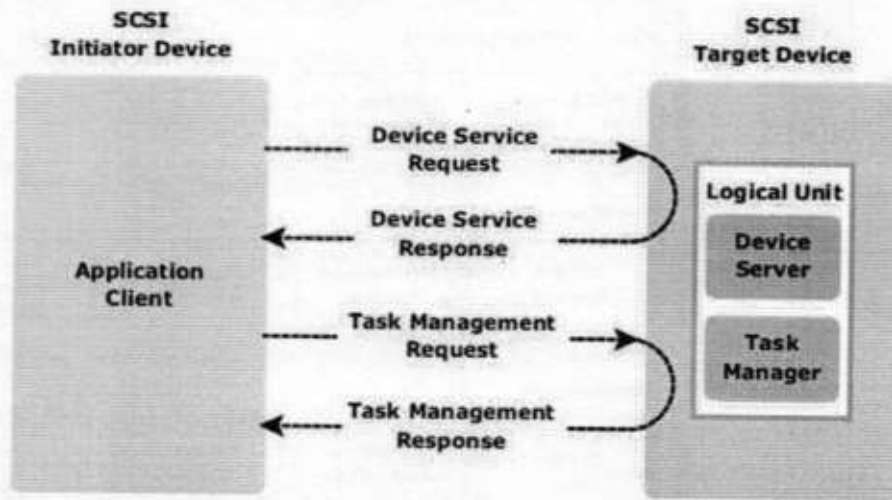


Figure 5-4: SCSI-3 client-server model

SCSI Ports

SCSI ports are the physical connectors that the SCSI cable plugs into for communication with a SCSI device. A SCSI device may contain target ports, initiator ports, target/initiator ports, or a target with multiple ports. Based on the port combinations, a SCSI device can be classified as an initiator model, a target model, a combined model, or a target model with multiple ports (see Figure 5-5).

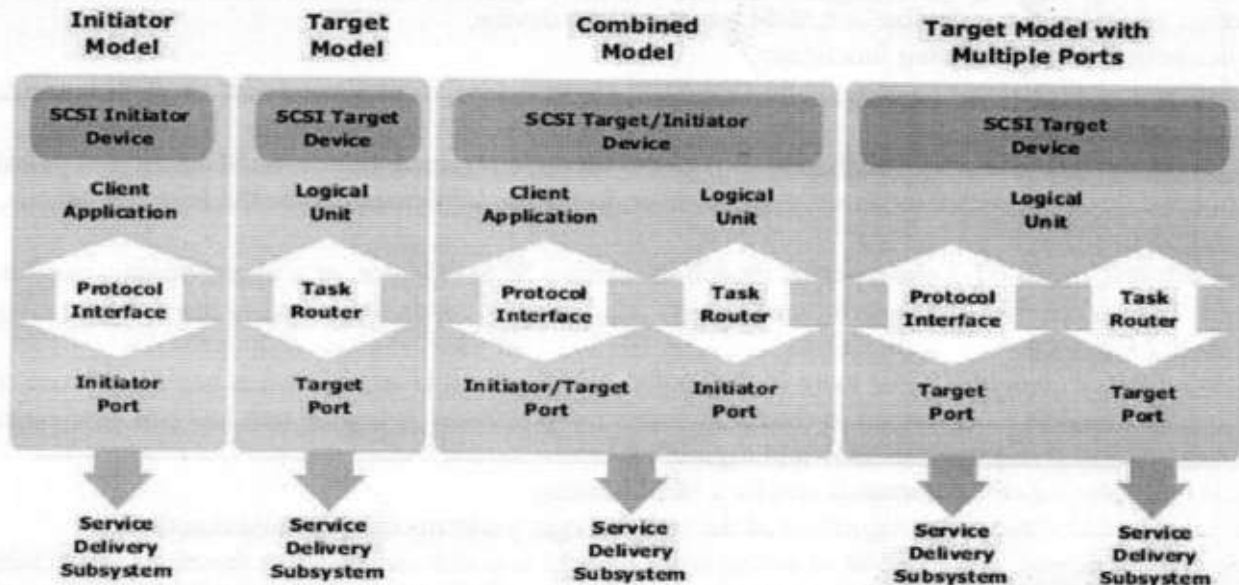


Figure 5-5: SCSI device models with different port configurations

In an initiator model, the SCSI initiator device has only initiator ports. Therefore, the application client can only initiate requests to the service delivery subsystem and receive confirmation. This device cannot serve any requests, and therefore does not contain a logical unit.

Similarly, a SCSI target device with only a target port can serve requests but cannot initiate them. The SCSI target/initiator device has a target/initiator port that can switch orientations depending on the role it plays while

participating in an I/O operation. To cater to service requests from multiple devices, a SCSI device may also have multiple ports of the same orientation (target).

SCSI Communication Model

A SCSI communication model (see Figure 5-6) is comprised of three interconnecting layers as defined in the SAM-3 and is similar to the OSI seven-layer model. Lower-level layers render their services to the upper-level layers. A high level layer communicates with a low-level layer by invoking the services that the low-level layer provides. The protocol at each layer defines the communication between peer layer entities.

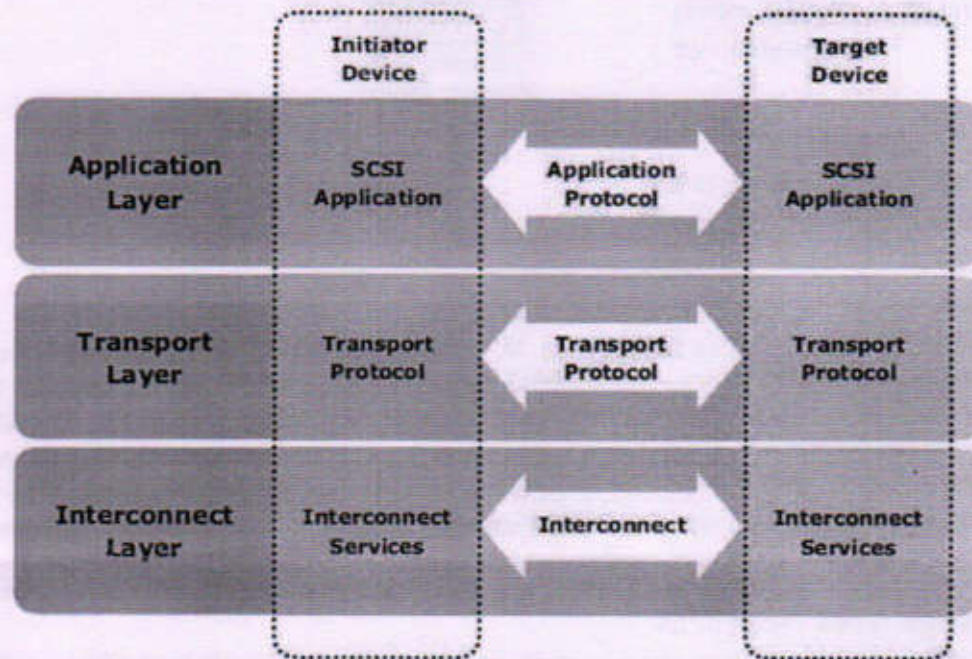


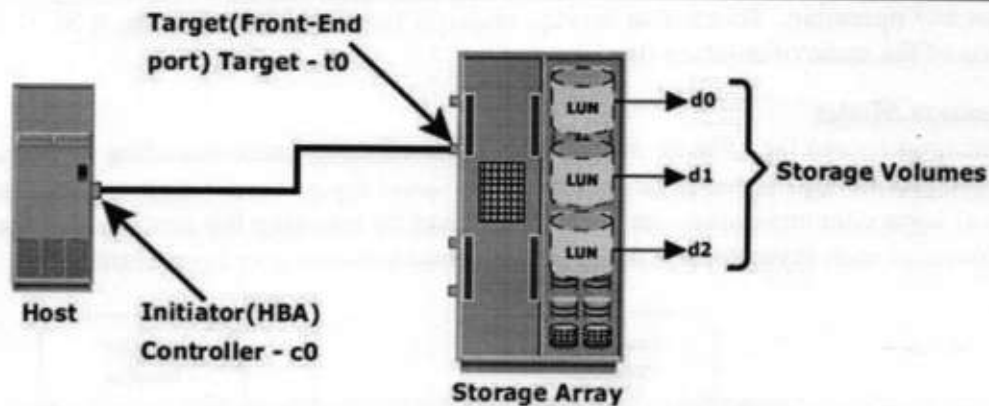
Figure 5-6: SCSI communication model

There are three layers in the SCSI communication model:

1. **SCSI application layer (SAL):** This layer contains both client and server applications that initiate and process SCSI I/O operations using a SCSI application protocol.
2. **SCSI transport protocol layer (STPL):** This layer contains the services and protocols that allow communication between an initiator and targets.
3. **Interconnect layer:** This layer facilitates data transfer between the initiator and targets. The interconnect layer is also known as the *service delivery subsystem* and comprises the services, signalling mechanisms, and interconnects for data transfer.

Parallel SCSI Addressing

In the Parallel SCSI Initiator-Target communication (see Figure 5-7), an initiator ID uniquely identifies the initiator and is used as an originating address. This ID is in the range of 0 to 15, with the range 0 to 7 being the most common. A target ID uniquely identifies a target and is used as the address for exchanging commands and status information with initiators. The target ID is in the range of 0 to 15.



Host Addressing :
Storage Volume 1 - c0 t0 d0
Storage Volume 2 - c0 t0 d1
Storage Volume 3 - c0 t0 d2

Figure 5-7: SCSI Initiator-Target communication

SCSI addressing is used to identify hosts and devices. In this addressing, the UNIX naming convention is used to identify a disk and the three identifiers—initiator ID, target ID, and a LUN—in the $cn|tn|dn$ format, which is also referred as *ctd addressing*. Here, cn is the initiator ID, commonly referred to as the controller ID; tn is the target ID of the device, such as $t0$, $t1$, $t2$, and so on; and dn is the device number reflecting the actual address of the device unit, such as $d0$, $d1$, and $d2$. A LUN identifies a specific logical unit in a target. The implementation of SCSI addressing may differ from one vendor to another. Figure 5-7 shows *ctd* addressing in the SCSI architecture.

Chapter 6 - Storage Area Networks

Organizations are experiencing an explosive growth in information. This information needs to be stored, protected, optimized, and managed efficiently. Data center managers are burdened with the challenging task of providing low-cost, high-performance information management solutions. An effective information management solution must provide the following:

1. **Just-in-time information to business users:** Information must be available to business users when they need it. The explosive growth in online storage, proliferation of new servers and applications, spread of mission-critical data throughout enterprises, and demand for 24×7 data availability are some of the challenges that need to be addressed.
2. **Integration of information infrastructure with business processes:** The storage infrastructure should be integrated with various business processes without compromising its security and integrity.
3. **Flexible and resilient storage architecture:** The storage infrastructure must provide flexibility and resilience that aligns with changing business requirements. Storage should scale without compromising performance requirements of the applications and, at the same time, the total cost of managing information must be low.

Chapter Objective: SAN is a high speed, dedicated network of servers and shared storage devices. Traditionally connected over Fibre Channel (FC) networks, a SAN forms a single-storage pool and facilitates data centralization and consolidation. SAN meets the storage demands efficiently with better economies of scale. A SAN also provides effective maintenance and protection of data.

This chapter provides detailed insight into the FC technology on which a SAN is deployed and also reviews SAN design and management fundamentals.

Fibre Channel: Overview

The FC architecture forms the fundamental construct of the SAN infrastructure. *Fibre Channel* is a high-speed network technology that runs on high-speed optical fiber cables (preferred for front-end SAN connectivity) and serial copper cables (preferred for back-end disk connectivity). The FC technology was created to meet the demand for increased speeds of data transfer among computers, servers, and mass storage subsystems.

FC networking was introduced in 1988, the FC standardization process began when the American National Standards Institute (ANSI) chartered the Fibre Channel Working Group (FCWG).

By 1994, the new high-speed computer interconnection standard was developed and the Fibre Channel Association (FCA) was founded with 70 charter member companies.

Technical Committee T11, which is the committee within INCITS (International Committee for Information Technology Standards), is responsible for Fibre Channel interfaces. T11 (previously known as X3T9.3) has been producing interface standards for high performance and mass storage applications since the 1970s.

Higher data transmission speeds are an important feature of the FC networking technology. The initial implementation offered throughput of 100 MB/s (equivalent to raw bit rate of 1Gb/s i.e. 1062.5 Mb/s in Fibre Channel), which was greater than the speeds of Ultra SCSI (20 MB/s) commonly used in DAS environments.

FC in full-duplex mode could sustain throughput of 200 MB/s. In comparison with Ultra-SCSI, FC is a significant leap in storage networking technology.

Latest FC implementations of 8 GFC (Fibre Channel) offers throughput of 1600 MB/s (raw bit rates of 8.5 Gb/s), whereas Ultra320 SCSI is available with a throughput of 320 MB/s. The FC architecture is highly scalable and theoretically a single FC network can accommodate approximately 15 million nodes.

The SAN and Its Evolution

A *storage area network (SAN)* carries data between servers (also known as *hosts*) and storage devices through fibre channel switches (see Figure 6-1). A SAN enables storage consolidation and allows storage to be shared across multiple servers. It enables organizations to connect geographically dispersed servers and storage.

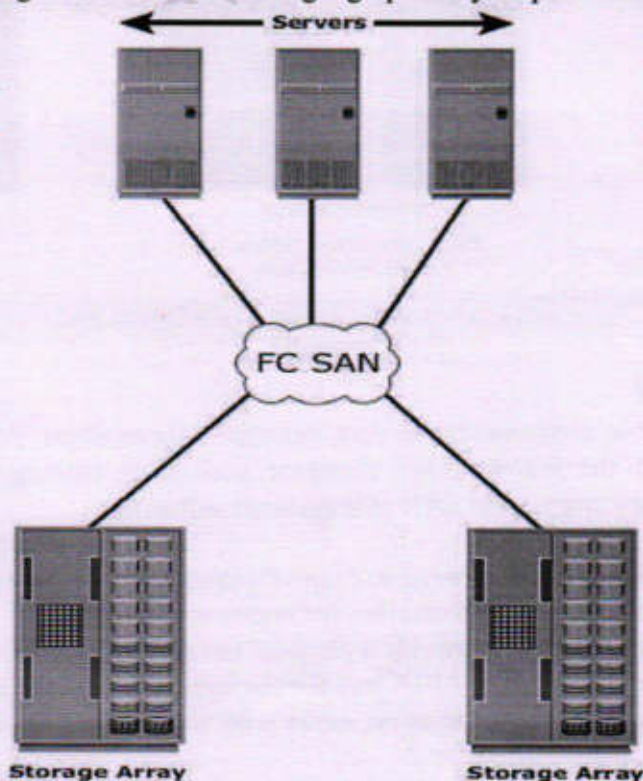
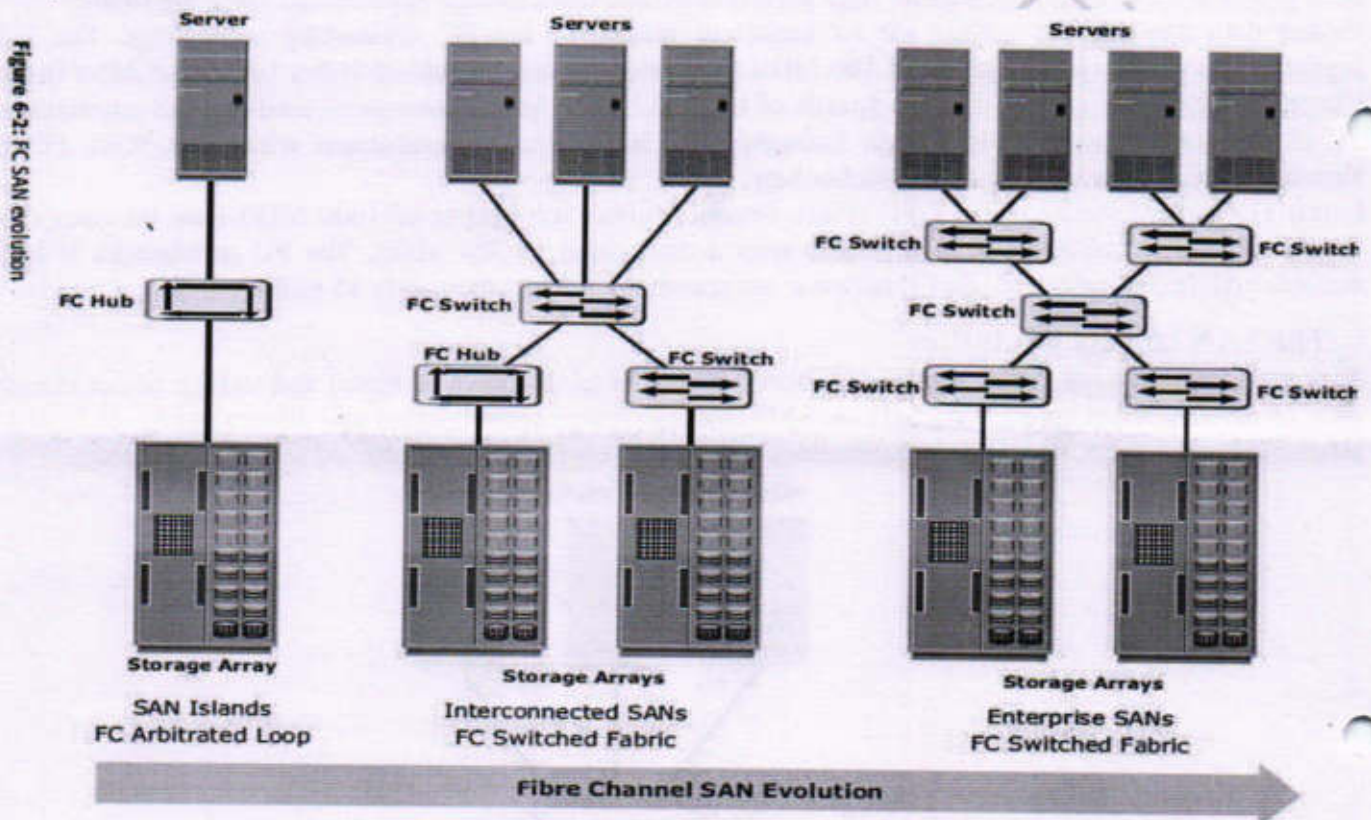


Figure 6-1: SAN implementation

A SAN provides the physical communication infrastructure and enables secure and robust communication between host and storage devices. The SAN management interface organizes connections and manages storage elements and hosts.

In its earliest implementation, the SAN was a simple grouping of hosts and the associated storage that was connected to a network using a hub as a connectivity device. This configuration of a SAN is known as a *Fibre Channel Arbitrated Loop (FC-AL)*, which is detailed later in the chapter. Use of hubs resulted in isolated FC-AL SAN islands because hubs provide limited connectivity and bandwidth.

The inherent limitations associated with hubs gave way to high-performance FC *switches*. The switched fabric topologies improved connectivity and performance, which enabled SANs to be highly scalable. This enhanced data accessibility to applications across the enterprise. FC-AL has been abandoned for SANs due to its limitations, but still survives as a disk-drive interface. Figure 6-2 illustrates the FC SAN evolution from FC-AL to enterprise SANs.



Components of SAN

A SAN consists of three basic components: servers, network infrastructure, and storage. These components can be further broken down into the following key elements: node ports, cabling, interconnecting devices (such as FC switches or hubs), storage arrays, and SAN management software.

Node Ports

In fibre channel, devices such as hosts, storage and tape libraries are all referred to as *nodes*.

Each node is a source or destination of information for one or more nodes.

Each node requires one or more ports to provide a physical interface for communicating with other nodes.

These ports are integral components of an HBA and the storage front-end adapters.

A port operates in full-duplex data transmission mode with a *transmit (Tx)* link and a *receive (Rx)* link (see Figure 6-3).

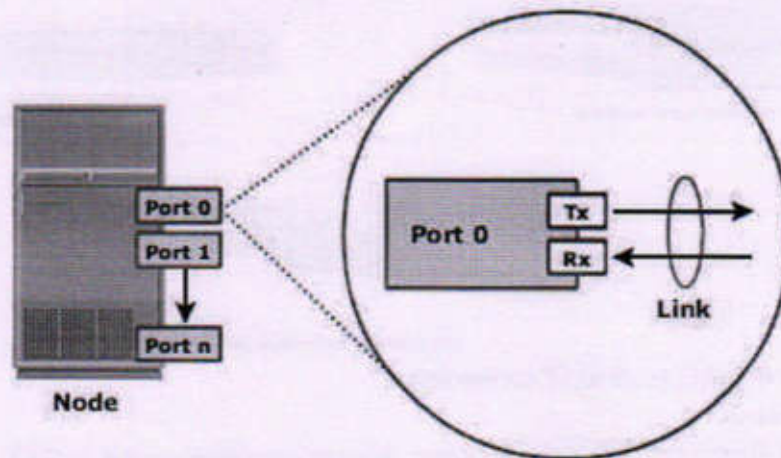


Figure 6-3: Nodes, ports, and links

Cabling

SAN implementations use optical fiber cabling. Copper can be used for shorter distances for back-end connectivity, as it provides a better signal-to-noise ratio for distances up to 30 meters. Optical fiber cables carry data in the form of light. There are two types of optical cables, multi-mode and single-mode. Multi-mode fiber (MMF) cable carries multiple beams of light projected at different angles simultaneously onto the core of the cable (see Figure 6-4 (a)).

Based on the bandwidth, multi-mode fibers are classified as OM1 (62.5µm), OM2 (50µm) and laser optimized OM3 (50µm). In an MMF transmission, multiple light beams traveling inside the cable tend to disperse and collide. This collision weakens the signal strength after it travels a certain distance — a process known as *modal dispersion*. An MMF cable is usually used for distances of up to 500 meters because of signal degradation (attenuation) due to modal dispersion. Single-mode fiber (SMF) carries a single ray of light projected at the center of the core (see Figure 6-4 (b)). These cables are available in diameters of 7–11 microns; the most common size is 9 microns. In an SMF transmission, a single light beam travels in a straight line through the core of the fiber. The small core and the single light wave limits modal dispersion. Among all types of fibre cables, single-mode provides minimum signal attenuation over maximum distance (up to 10 km).

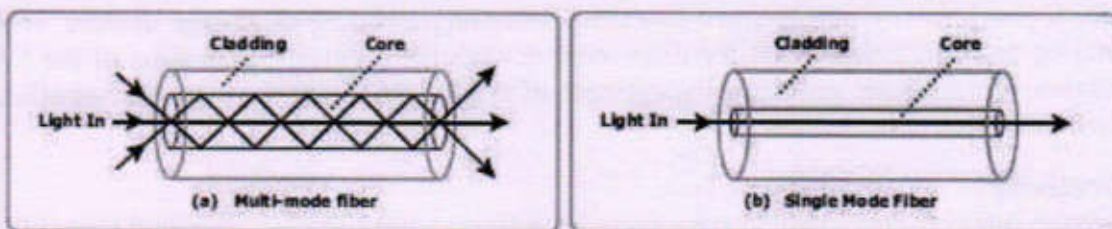


Figure 6-4: Multi-mode fiber and single-mode fiber

A Standard connector (SC) (see Figure 6-5 (a)) and a Lucent connector (LC) (see Figure 6-5 (b)) are two commonly used connectors for fiber optic cables. An SC is used for data transmission speeds up to 1 Gb/s, whereas an LC is used for speeds up to 4 Gb/s. Figure 6-6 depicts a Lucent connector and a Standard connector. A *Straight Tip (ST)* is a fiber optic connector with a plug and a socket that is locked with a half-twisted bayonet lock (see Figure 6-5 (c)). In the early days of FC deployment, fiber optic cabling predominantly used ST connectors. This connector is often used with Fibre Channel patch panels. The Small Form-factor Pluggable (SFP) is an optical transceiver used in optical communication. The standard SFP+ transceivers support data rates up to 10 Gb/s.

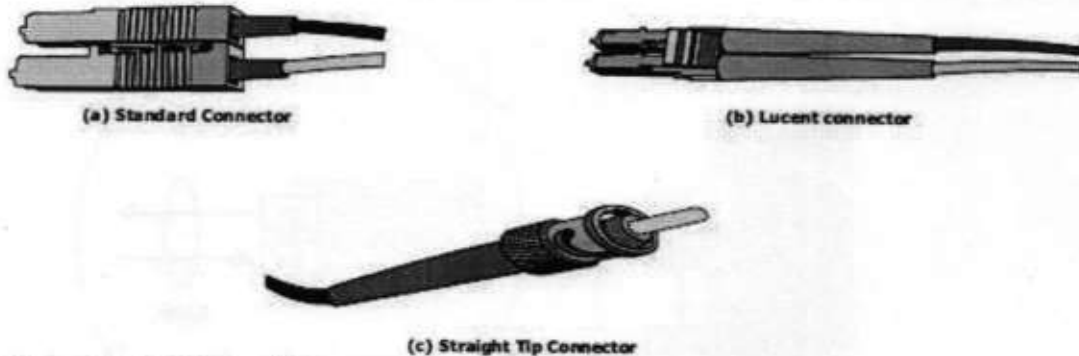


Figure 6-5: SC, LC, and ST connectors

Interconnect Devices

Hubs, switches, and directors are the interconnect devices commonly used in SAN.

Hubs are used as communication devices in FC-AL implementations. Hubs physically connect nodes in a logical loop or a physical star topology.

All the nodes must share the bandwidth because data travels through all the connection points. Because of availability of low cost and high performance switches, hubs are no longer used in SANs.

Switches are more intelligent than hubs and directly route data from one physical port to another. Therefore, nodes do not share the bandwidth. Instead, each node has a dedicated communication path, resulting in bandwidth aggregation.

Directors are larger than switches and are deployed for data center implementations. The function of directors is similar to that of FC switches, but directors have higher port count and fault tolerance capabilities.

Storage Arrays

The fundamental purpose of a SAN is to provide host access to storage resources. The large storage capacities offered by modern storage arrays have been exploited in SAN environments for storage consolidation and centralization. SAN implementations complement the standard features of storage arrays by providing high availability and redundancy, improved performance, business continuity, and multiple host connectivity.

SAN Management Software

SAN management software manages the interfaces between hosts, interconnect devices, and storage arrays. The software provides a view of the SAN environment and enables management of various resources from one central console. It provides key management functions, including mapping of storage devices, switches, and servers, monitoring and generating alerts for discovered devices, and logical partitioning of the SAN, called *zoning*. In addition, the software provides management of typical SAN components such as HBAs, storage components, and interconnecting devices.

FC Connectivity

The FC architecture supports three basic interconnectivity options: point-to-point, arbitrated loop (FC-AL), and fabric connect.

Point-to-Point

Point-to-point is the simplest FC configuration — two devices are connected directly to each other, as shown in Figure 6-6.

This configuration provides a dedicated connection for data transmission between nodes. However, the point-to-point configuration offers limited connectivity, as only two devices can communicate with each other at a given time.

Moreover, it cannot be scaled to accommodate a large number of network devices.

Standard DAS uses point-to-point connectivity.

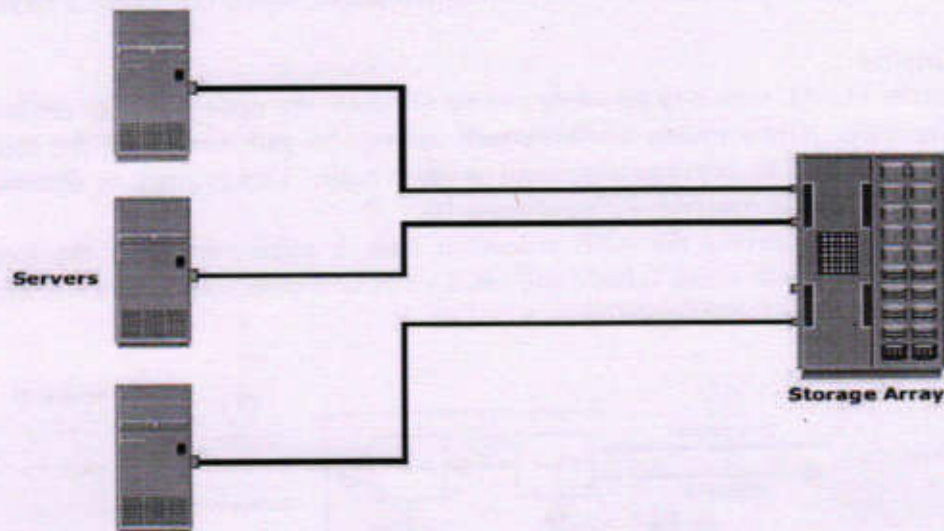


Figure 6-6: Point-to-point topology

Fibre Channel Arbitrated Loop

In the FC-AL configuration, devices are attached to a shared loop, as shown in Figure 6-7.

FC-AL has the characteristics of a token ring topology and a physical star topology.

In FC-AL, each device contends with other devices to perform I/O operations.

Devices on the loop must “arbitrate” to gain control of the loop.

At any given time, only one device can perform I/O operations on the loop.

As a loop configuration, FC-AL can be implemented without any interconnecting devices by directly connecting one device to another in a ring through cables.

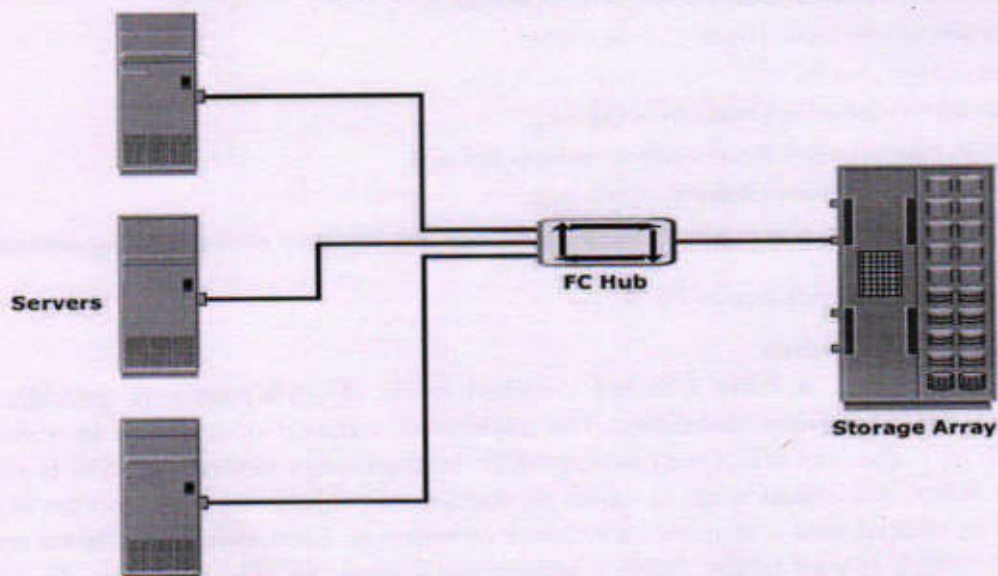


Figure 6-7: Fibre Channel arbitrated loop

The FC-AL configuration has the following limitations in terms of scalability:

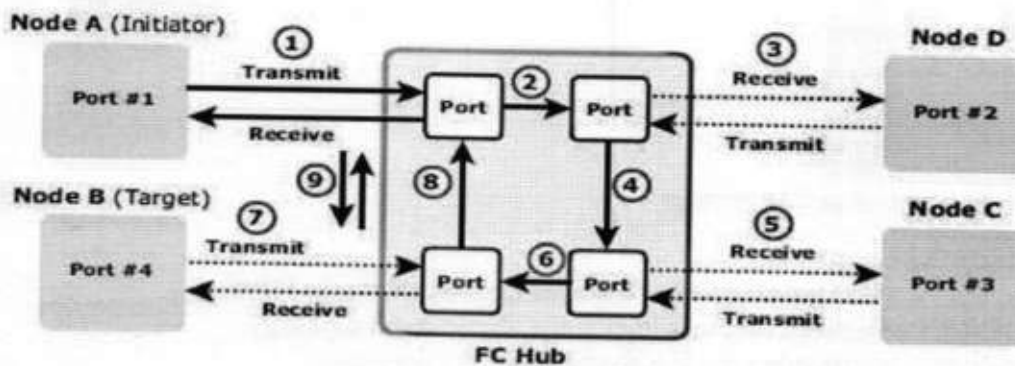
1. FC-AL shares the bandwidth in the loop. Only one device can perform I/O operations at a time. Because each device in a loop has to wait for its turn to process I/O request, the speed of data transmission is low in an FC-AL topology.
2. FC-AL uses 8-bit addressing. It can support up to 127 devices on a loop.

3. Adding or removing a device results in loop re-initialization, which can cause a momentary pause in loop traffic.

FC-AL Transmission

When a node in the FC-AL topology attempts to transmit data, the node sends an *arbitration (ARB)* frame to each node on the loop. If two nodes simultaneously attempt to gain control of the loop, the node with the highest priority is allowed to communicate with another node. This priority is determined on the basis of Arbitrated Loop Physical Address (AL-PA) and Loop ID.

When the initiator node receives the ARB request it sent, it gains control of the loop. The initiator then transmits data to the node with which it has established a virtual connection. Figure 6-8 illustrates the process of data transmission in an FC-AL configuration.



Node A want to communicate with Node B

- ① High priority initiator, Node A inserts the ARB frame in the loop.
- ② ARB frame is passed to the next node (Node D) in the loop.
- ③ Node D receives high priority ARB, therefore remains idle.
- ④ ARB is forwarded to next node (Node C) in the loop.
- ⑤ Node C receives high priority ARB, therefore remains idle.
- ⑥ ARB is forwarded to next node (Node B) in the loop.
- ⑦ Node B receives high priority ARB, therefore remains idle and
- ⑧ ARB is forwarded to next node (Node A) in the loop.
- ⑨ Node A receives ARB back; now it gains control of the loop and can start communicating with target Node B.

Figure 6-8: Data transmission in FC-AL

Fibre Channel Switched Fabric

Unlike a loop configuration, a Fibre Channel switched fabric (FC-SW) network provides interconnected devices, dedicated bandwidth, and scalability. The addition or removal of a device in a switched fabric is minimally disruptive; it does not affect the ongoing traffic between other devices. FC-SW is also referred to as *fabric connect*. A fabric is a logical space in which all nodes communicate with one another in a network. This virtual space can be created with a switch or a network of switches. Each switch in a fabric contains a unique domain identifier, which is part of the fabric's addressing scheme. In FC-SW, nodes do not share a loop; instead, data is transferred through a dedicated path between the nodes. Each port in a fabric has a unique 24-bit fibre channel address for communication. Figure 6-9 shows an example of FC-SW.

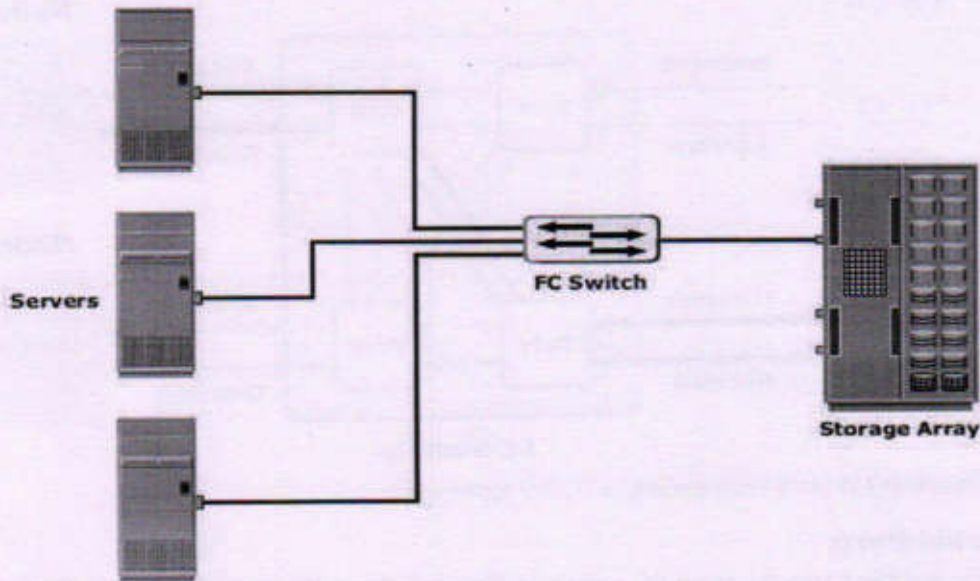


Figure 6-9: Fibre Channel switched fabric

When the number of tiers in a fabric increases, the distance that a fabric management message must travel to reach each switch in the fabric also increases. The increase in the distance also increases the time taken to propagate and complete a fabric reconfiguration event, such as the addition of a new switch, or a zone set propagation event (detailed later in this chapter). Figure 6-10 illustrates two-tier and three-tier fabric architecture.

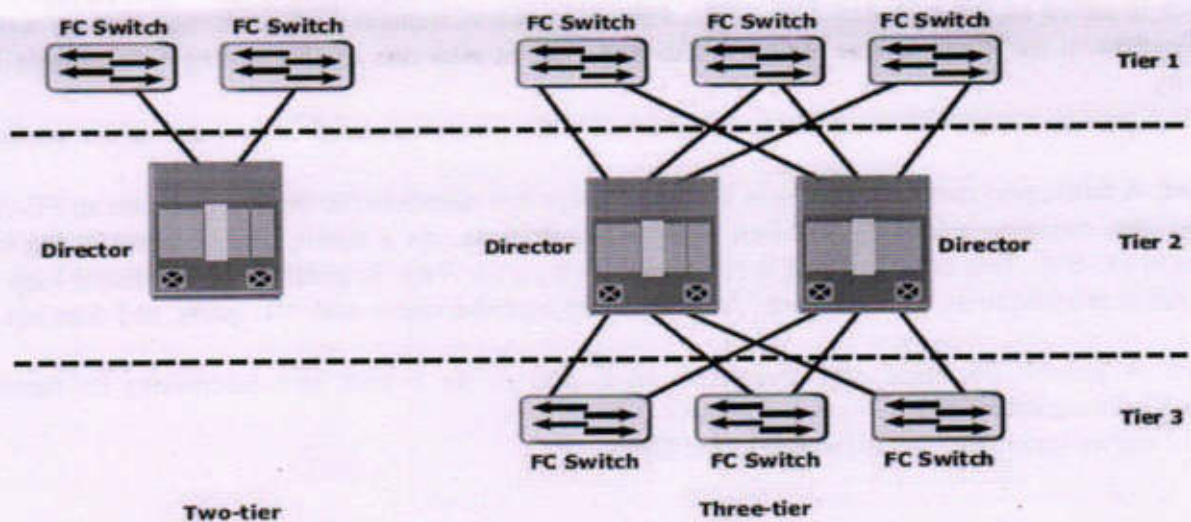


Figure 6-10: Tiered structure of FC-SW topology

FC-SW Transmission

FC-SW uses switches that are intelligent devices. They can switch data traffic from an initiator node to a target node directly through switch ports. Frames are routed between source and destination by the fabric. As shown in Figure 6-11, if node B wants to communicate with node D, Nodes should individually login first and then transmit data via the FC-SW. This link is considered a dedicated connection between the initiator and the target.

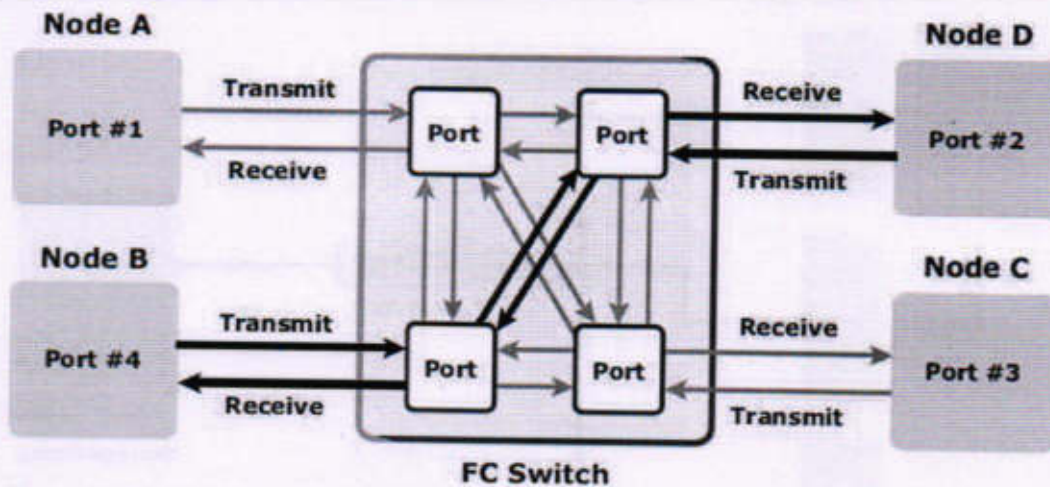


Figure 6-11: Data transmission in FC-SW topology

Fibre Channel Ports

Ports are the basic building blocks of an FC network. Ports on the switch can be one of the following types:

1. **N_port:** An end point in the fabric. This port is also known as the *node port*. Typically, it is a host port (HBA) or a storage array port that is connected to a switch in a switched fabric.
2. **NL_port:** A node port that supports the arbitrated loop topology. This port is also known as the *node loop port*.
3. **E_port:** An FC port that forms the connection between two FC switches. This port is also known as the *expansion port*. The E_port on an FC switch connects to the E_port of another FC switch in the fabric through a link, which is called an *Inter-Switch Link (ISL)*. ISLs are used to transfer host-to-storage data as well as the fabric management traffic from one switch to another. ISL is also one of the scaling mechanisms in SAN connectivity.
4. **F_port:** A port on a switch that connects an N_port. It is also known as a *fabric port* and cannot participate in FC-AL.
5. **FL_port:** A fabric port that participates in FC-AL. This port is connected to the NL_ports on an FC-AL loop. A FL_port also connects a loop to a switch in a switched fabric. As a result, all NL_ports in the loop can participate in FC-SW. This configuration is referred to as a *public loop*. In contrast, an arbitrated loop without any switches is referred to as a *private loop*. A private loop contains nodes with NL_ports, and does not contain FL_port.
6. **G_port:** A generic port that can operate as an E_port or an F_port and determines its functionality automatically during initialization.

Figure 6-12 shows various FC ports located in the fabric.

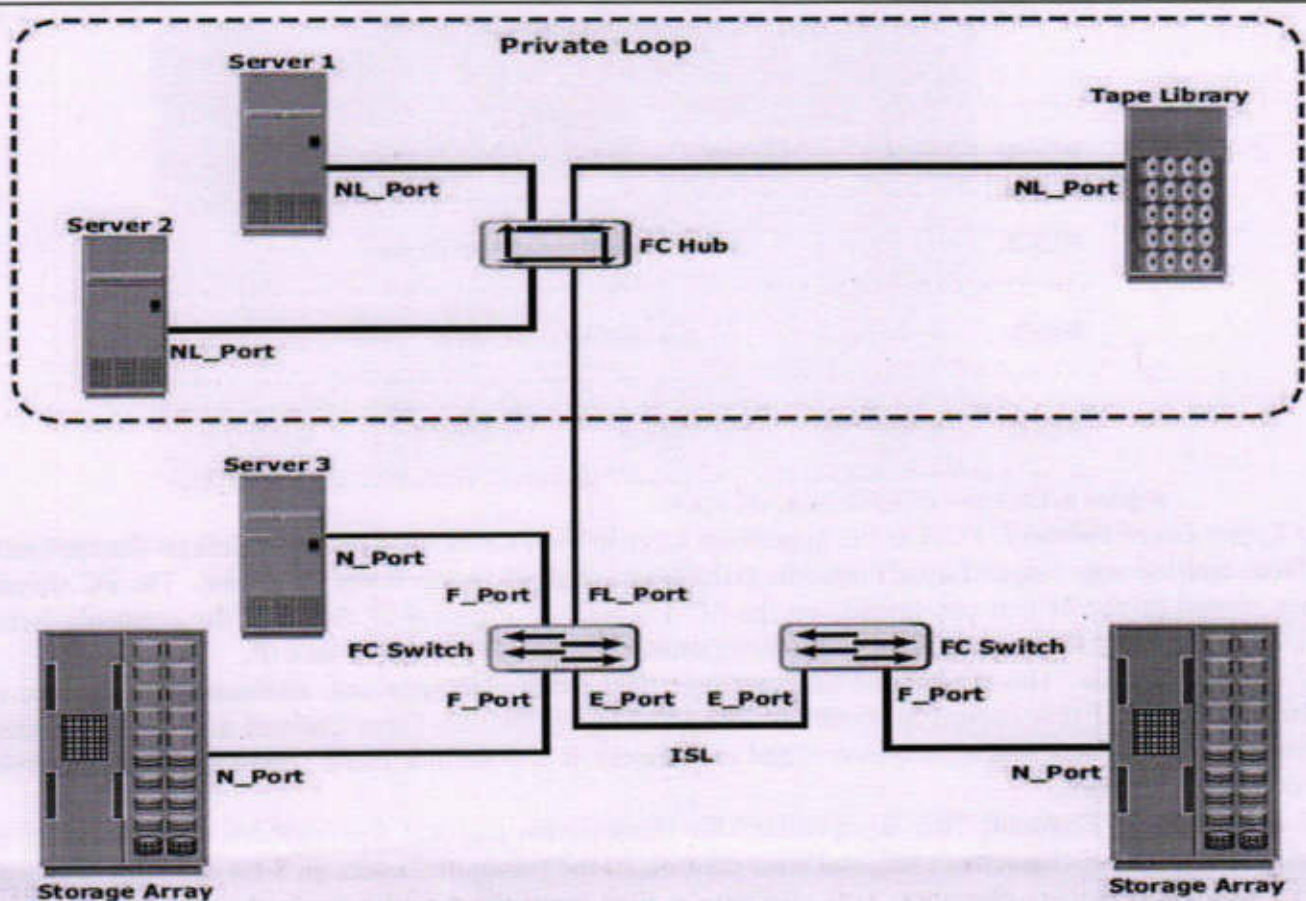


Figure 6-12: Fibre channel ports

Fibre Channel Architecture

The FC architecture represents true channel/network integration with standard interconnecting devices. Connections in a SAN are accomplished using FC. Channel technologies provide high levels of performance with low protocol overheads. Such performance is due to the static nature of channels and the high level of hardware and software integration provided by the channel technologies.

Fibre Channel Protocol (FCP) is the implementation of serial SCSI-3 over an FC network. In the FCP architecture, all external and remote storage devices attached to the SAN appear as local devices to the host operating system.

The key advantages of FCP are as follows:

1. Sustained transmission bandwidth over long distances.
2. Support for a larger number of addressable devices over a network. Theoretically, FC can support over 15 million device addresses on a network.
3. Exhibits the characteristics of channel transport and provides speeds up to 8.5 Gb/s (8 GFC).

Fibre Channel Protocol Stack

It is easier to understand a communication protocol by viewing it as a structure of independent layers. FCP defines the communication protocol in five layers: FC-0 through FC-4 (except FC-3 layer, which is not implemented). In a layered communication model, the peer layers on each node talk to each other through defined protocols. Figure 6-13 illustrates the fibre channel protocol stack.

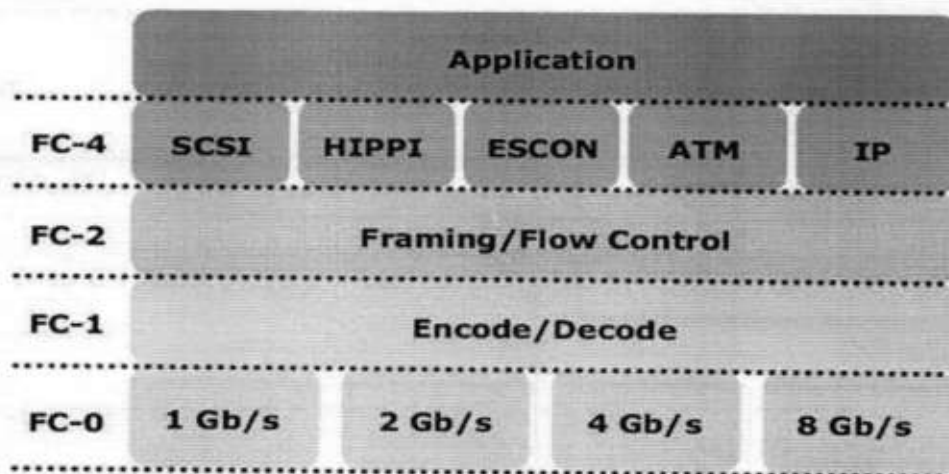


Figure 6-13: Fibre channel protocol stack

FC-4 Upper Layer Protocol: FC-4 is the uppermost layer in the FCP stack. This layer defines the application interfaces and the way Upper Layer Protocols (ULPs) are mapped to the lower FC layers. The FC standard defines several protocols that can operate on the FC-4 layer (see Figure 6-7). Some of the protocols include SCSI, HIPPI Framing Protocol, Enterprise Storage Connectivity (ESCON), ATM, and IP.

FC-2 Transport Layer: The FC-2 is the transport layer that contains the payload, addresses of the source and destination ports, and link control information. The FC-2 layer provides Fibre Channel addressing, structure, and organization of data (frames, sequences, and exchanges). It also defines fabric services, classes of service, flow control, and routing.

FC-1 Transmission Protocol: This layer defines the transmission protocol that includes serial encoding and decoding rules, special characters used, and error control. At the transmitter node, an 8-bit character is encoded into a 10-bit transmissions character. This character is then transmitted to the receiver node. At the receiver node, the 10-bit character is passed to the FC-1 layer, which decodes the 10-bit character into the original 8-bit character.

FC-0 Physical Interface: FC-0 is the lowest layer in the FCP stack. This layer defines the physical interface, media, and transmission of raw bits. The FC-0 specification includes cables, connectors, and optical and electrical parameters for a variety of data rates. The FC transmission can use both electrical and optical media.

Fibre Channel Addressing

An FC address is dynamically assigned when a port logs on to the fabric. The FC address has a distinct format that varies according to the type of node port in the fabric. These ports can be an N_port and an NL_port in a public loop, or an NL_port in a private loop.

The first field of the FC address of an N_port contains the domain ID of the switch (see Figure 6-14). This is an 8-bit field. Out of the possible 256 domain IDs, 239 are available for use; the remaining 17 addresses are reserved for specific services. For example, FFFFFC is reserved for the name server, and FFFFFE is reserved for the fabric login service. The maximum possible number of N_ports in a switched fabric is calculated as $239 \text{ domains} \times 256 \text{ areas} \times 256 \text{ ports} = 15,663,104$ Fibre Channel addresses.

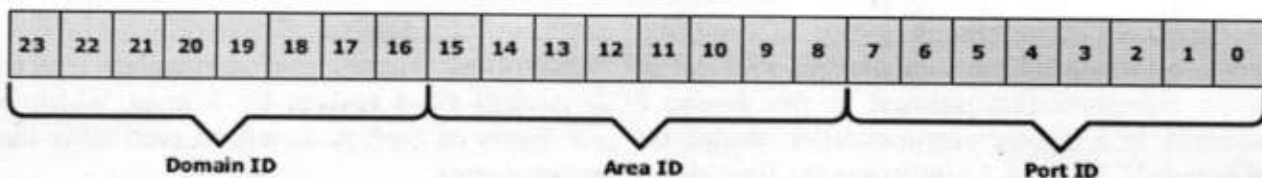


Figure 6-14: 24-bit FC address of N_port

FC Address of an NL_port

The FC addressing scheme for an NL_port differs from other ports. The two upper bytes in the FC addresses of the NL_ports in a private loop are assigned zero values. However, when an arbitrated loop is connected to a fabric through an FL_port, it becomes a public loop. In this case, an NL_port supports a fabric login. The two upper bytes of this NL_port are then assigned a positive value, called a *loop identifier*, by the switch. The loop identifier is the same for all NL_ports on a given loop.

Figure 6-15 illustrates the FC address of an NL_port in both a public loop and a private loop. The last field in the FC addresses of the NL_ports, in both public and private loops, identifies the AL-PA. There are 127 allowable AL-PA addresses; one address is reserved for the FL_port on the switch.

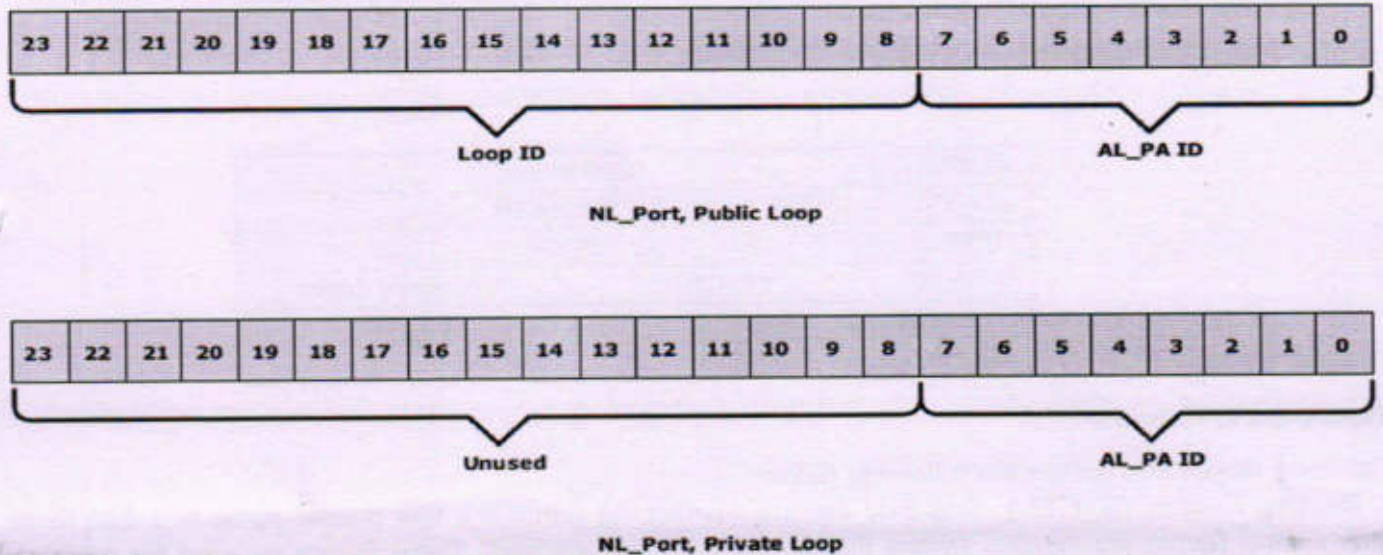


Figure 6-15: 24-bit FC address of NL_port

World Wide Names

Each device in the FC environment is assigned a 64-bit unique identifier called the *World Wide Name* (WWN). The Fibre Channel environment uses two types of WWNs: World Wide Node Name (WWNN) and World Wide Port Name (WWPN). Unlike an FC address, which is assigned dynamically, a WWN is a static name for each device on an FC network. WWNs are similar to the Media Access Control (MAC) addresses used in IP networking. WWNs are *burned* into the hardware or assigned through software. Several configuration definitions in a SAN use WWN for identifying storage devices and HBAs. The name server in an FC environment keeps the association of WWNs to the dynamically created FC addresses for nodes. Figure 6-16 illustrates the WWN structure for an array and the HBA.

World Wide Name - Array															
5	0	0	6	0	1	6	0	0	0	6	0	0	1	B	2
0101	0000	0000	0110	0000	0001	0110	0000	0000	0000	0110	0000	0000	0001	1011	0010
Company ID 24 bits						Port		Model Seed 32 bits							

World Wide Name - HBA															
1	0	0	0	0	0	0	0	c	9	2	0	d	c	4	0
Reserved 12 bits				Company ID 24 bits						Company Specific 24 bits					

Figure 6-16: World Wide Names

FC Frame

An FC frame (Figure 6-17) consists of five parts: *start of frame (SOF)*, *frame header*, *data field*, *cyclic redundancy check (CRC)*, and *end of frame (EOF)*. The SOF and EOF act as delimiters. In addition to this role, the SOF is a flag that indicates whether the frame is the first frame in a sequence of frames. The frame header is 24 bytes long and contains addressing information for the frame. It includes the following information: Source ID (S_ID), Destination ID (D_ID), Sequence ID (SEQ_ID), Sequence Count (SEQ_CNT), Originating Exchange ID (OX_ID), and Responder Exchange ID (RX_ID), in addition to some control fields.

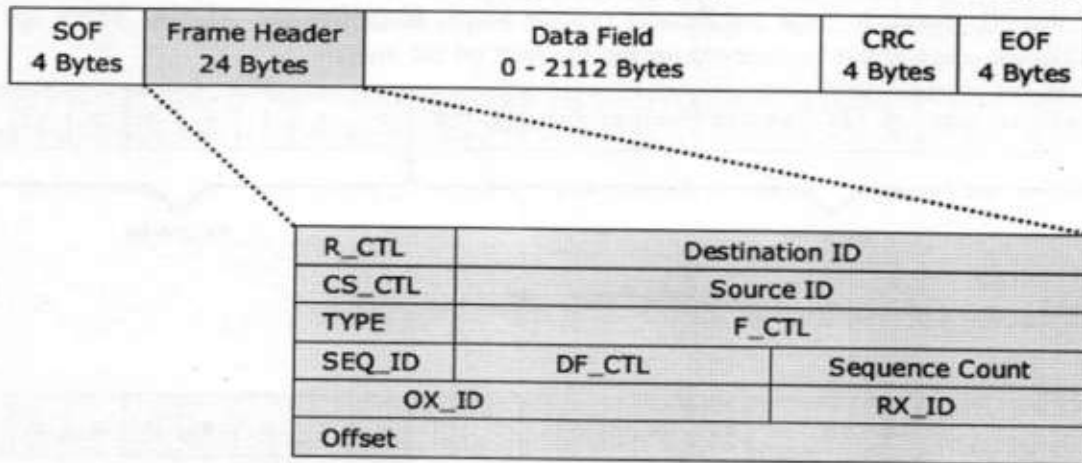


Figure 6-17: FC frame

The frame header also defines the following fields:

1. **Routing Control (R_CTL):** This field denotes whether the frame is a link control frame or a data frame. Link control frames are nondata frames that do not carry any payload. These frames are used for setup and messaging. In contrast, data frames carry the payload and are used for data transmission.
2. **Class Specific Control (CS_CTL):** This field specifies link speeds for class 1 and class 4 data transmission.
3. **TYPE:** This field describes the upper layer protocol (ULP) to be carried on the frame if it is a data frame. However, if it is a link control frame, this field is used to signal an event such as “fabric busy.” For example, if the TYPE is 08, and the frame is a data frame, it means that the SCSI will be carried on an FC.
4. **Data Field Control (DF_CTL):** A 1-byte field that indicates the existence of any optional headers at the beginning of the data payload. It is a mechanism to extend header information into the payload.
5. **Frame Control (F_CTL):** A 3-byte field that contains control information related to frame content. For example, one of the bits in this field indicates whether this is the first sequence of the exchange.

6.6.4. Structure and Organization of FC Data

In an FC network, data transport is analogous to a conversation between two people, whereby a frame represents a word, a sequence represents a sentence, and an exchange represents a conversation.

1. **Exchange operation:** An exchange operation enables two N_ports to identify and manage a set of information units. This unit maps to a sequence. Sequences can be both unidirectional and bidirectional depending upon the type of data sequence exchanged between the initiator and the target.
2. **Sequence:** A sequence refers to a contiguous set of frames that are sent from one port to another. A sequence corresponds to an information unit, as defined by the ULP.
3. **Frame:** A frame is the fundamental unit of data transfer at Layer 2. Each frame can contain up to 2,112 bytes of payload.

Flow Control

Flow control defines the pace of the flow of data frames during data transmission. FC technology uses two flow-control mechanisms: buffer-to-buffer credit (BB_Credit) and end-to-end credit (EE_Credit).

1. **BB_Credit:** FC uses the *BB_Credit* mechanism for hardware-based flow control. *BB_Credit* controls the maximum number of frames that can be present over the link at any given point in time. In a switched fabric, *BB_Credit* management may take place between any two FC ports. The transmitting port maintains a count of free receiver buffers and continues to send frames if the count is greater than 0. The *BB_Credit* mechanism provides frame acknowledgment through the *Receiver Ready (R_RDY)* primitive.
2. **EE_Credit:** The function of end-to-end credit, known as *EE_Credit*, is similar to that of *BB_Credit*. When an initiator and a target establish themselves as nodes communicating with each other, they exchange the *EE_Credit* parameters (part of Port Login). The *EE_Credit* mechanism affects the flow control for class 1 and class 2 traffic only.

Classes of Service

The FC standards define different classes of service to meet the requirements of a wide range of applications. The table below shows three classes of services and their features (Table 6-1).

Table 6-1: FC Class of Services

	CLASS 1	CLASS 2	CLASS 3
Communication type	Dedicated connection	Nondedicated connection	Nondedicated connection
Flow control	End-to-end credit	End-to-end credit B-to-B credit	B-to-B credit
Frame delivery	In order delivery	Order not guaranteed	Order not guaranteed
Frame acknowledgement	Acknowledged	Acknowledged	Not acknowledged
Multiplexing	No	Yes	Yes
Bandwidth utilization	Poor	Moderate	High

Another class of services is *class F*, which is intended for use by the switches communicating through ISLs. Class F is similar to Class 2, and it provides notification of nondelivery of frames. Other defined Classes 4, 5, and 6 are used for specific applications.

Zoning

Zoning is an FC switch function that enables nodes within the fabric to be logically segmented into groups that can communicate with each other (see Figure 6-18). When a device (host or storage array) logs onto a fabric, it is registered with the name server. When a port logs onto the fabric, it goes through a device discovery process with other devices registered in the name server. The zoning function controls this process by allowing only the members in the same zone to establish these link-level services.

Multiple zone sets may be defined in a fabric, but only one zone set can be active at a time. A zone set is a set of zones and a zone is a set of members. A member may be in multiple zones. Members, zones, and zone sets form the hierarchy defined in the zoning process (see Figure 6-19). *Members* are nodes within the SAN that can be included in a zone. *Zones* comprise a set of members that have access to one another. A port or a node can be a member of multiple zones. *Zone sets* comprise a group of zones that can be activated or deactivated as a single entity in a fabric. Only one zone set per fabric can be active at a time. Zone sets are also referred to as *zone configurations*.

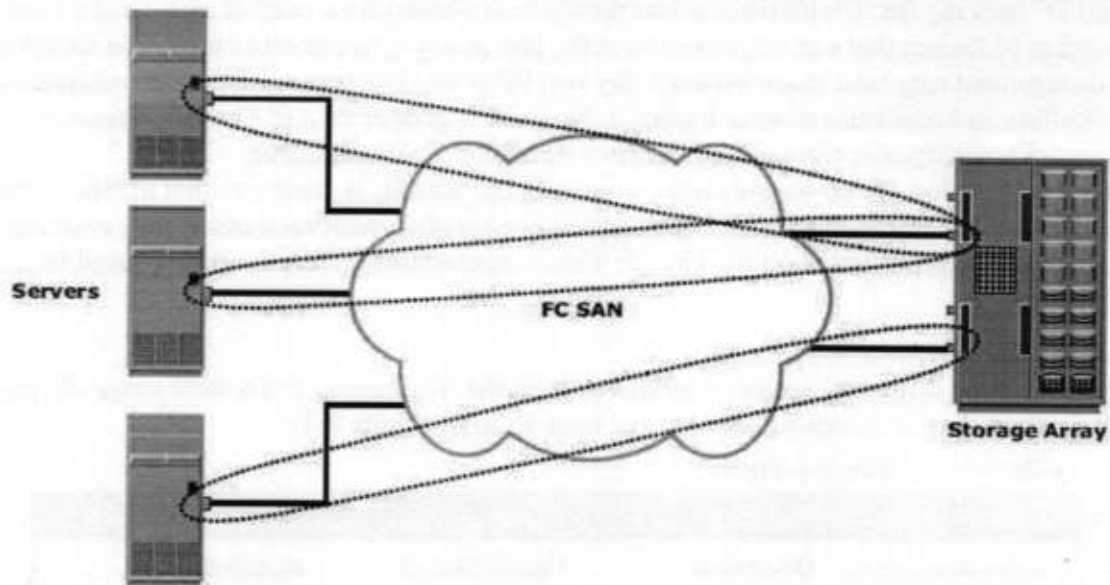


Figure 6-18: Zoning

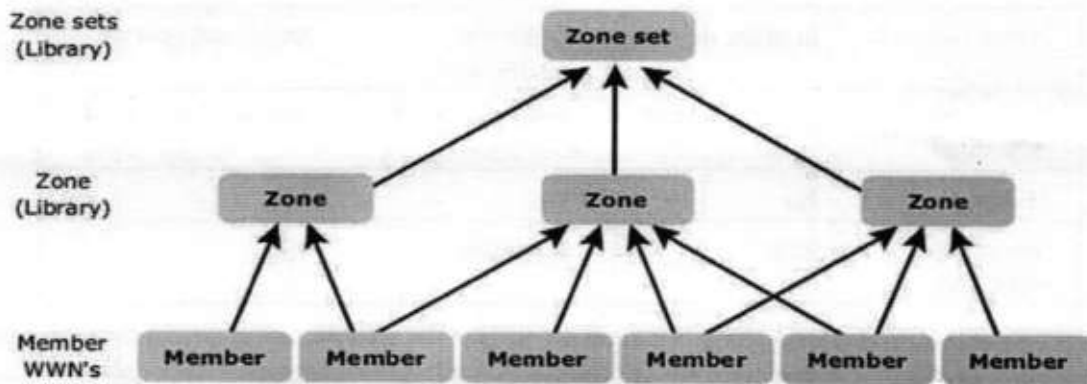


Figure 6-19: Members, zones, and zone sets

Types of Zoning

Zoning can be categorized into three types:

1. Port zoning: It uses the FC addresses of the physical ports to define zones. In port zoning, access to data is determined by the physical switch port to which a node is connected. The FC address is dynamically assigned when the port logs on to the fabric. Therefore, any change in the fabric configuration affects zoning. Port zoning is also called *hard zoning*. Although this method is secure, it requires updating of zoning configuration information in the event of fabric reconfiguration.

2. WWN zoning: It uses World Wide Names to define zones. WWN zoning is also referred to as *soft zoning*. A major advantage of WWN zoning is its flexibility. It allows the SAN to be recabled without reconfiguring the zone information. This is possible because the WWN is static to the node port.

3. Mixed zoning: It combines the qualities of both WWN zoning and port zoning. Using mixed zoning enables a specific port to be tied to the WWN of a node.

Figure 6-20 shows the three types of zoning on an FC network.

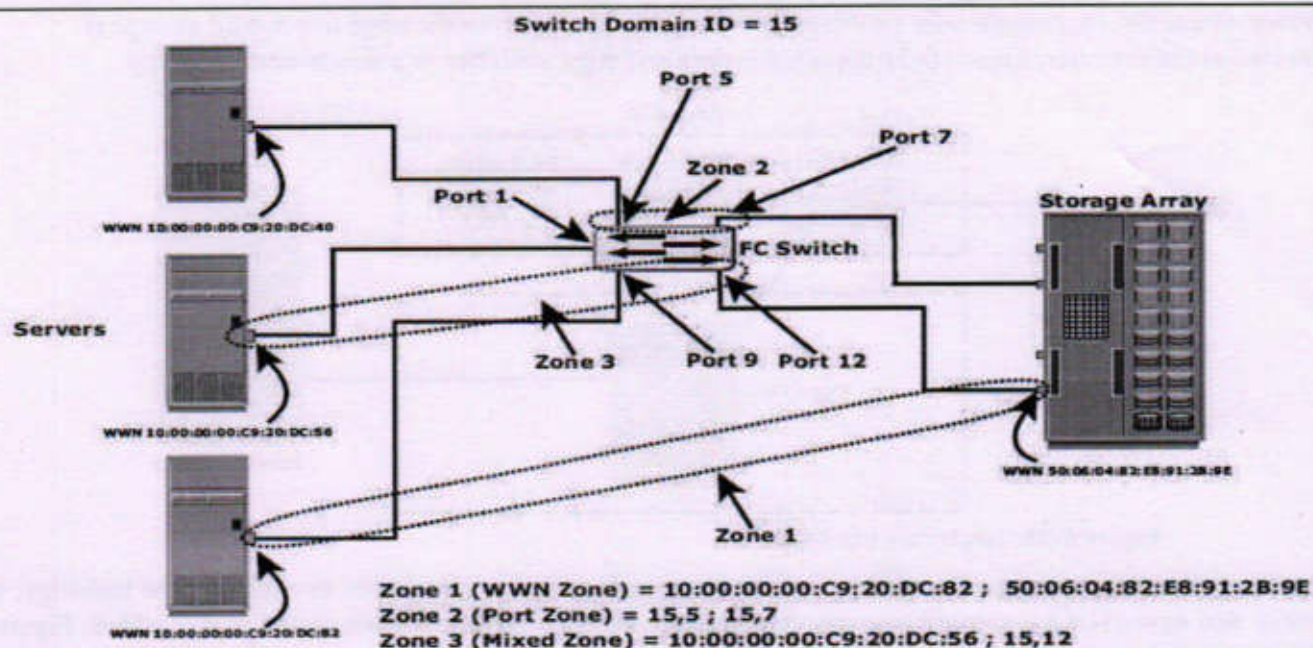


Figure 6-20: Types of zoning

Fibre Channel Login Types

Fabric services define three login types:

1. **Fabric login (FLOGI)** is performed between an N_port and an F_port. To log on to the fabric, a device sends a FLOGI frame with the World Wide Node Name (WWNN) and World Wide Port Name (WWPN) parameters to the login service at the well-known FC address FFFFFE. In turn, the switch accepts the login and returns an Accept (ACC) frame with the assigned FC address for the device. Immediately after the FLOGI, the N_port registers itself with the local name server on the switch, indicating its WWNN, WWPN, and assigned FC address.
2. **Port login (PLOGI)** is performed between an N_port and another N_port to establish a session. The initiator N_port sends a PLOGI request frame to the target N_port, which accepts it. The target N_port returns an ACC to the initiator N_port. Next, the N_ports exchange service parameters relevant to the session.
3. **Process login (PRLI)** is also performed between an N_port and another N_port. This login relates to the FC-4 ULPs such as SCSI. N_ports exchange SCSI-3-related service parameters. N_ports share information about the FC-4 type in use, the SCSI initiator, or the target.

FC Topologies

Fabric design follows standard topologies to connect devices. Core-edge fabric is one of the popular topology designs.

Core-Edge Fabric

In the *core-edge fabric* topology, there are two types of switch tiers in this fabric. The *edge tier* usually comprises switches and offers an inexpensive approach to adding more hosts in a fabric. The tier at the edge fans out from the tier at the core. The nodes on the edge can communicate with each other.

The *core tier* usually comprises enterprise directors that ensure high fabric availability. Additionally all traffic has to either traverse through or terminate at this tier. In a two-tier configuration, all storage devices are connected to the core tier, facilitating fan-out.

The core-edge fabric topology increases connectivity within the SAN while conserving overall port utilization. If expansion is required, an additional edge switch can be connected to the core. This topology can have

different variations. In a *single-core topology*, all hosts are connected to the edge tier and all storage is connected to the core tier. Figure 6-21 depicts the core and edge switches in a single-core topology.

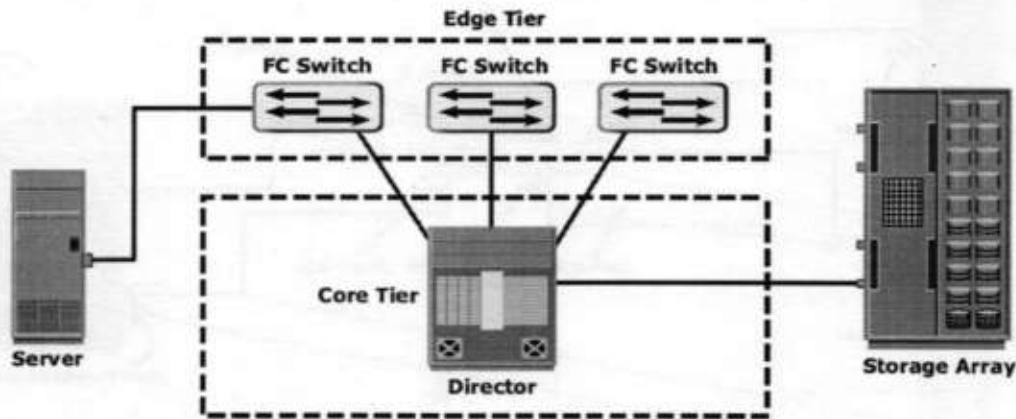


Figure 6-21: Single core topology

A *dual-core topology* can be expanded to include more core switches. However, to maintain the topology, it is essential that new ISLs are created to connect each edge switch to the new core switch that is added. Figure 6-22 illustrates the core and edge switches in a dual-core topology.

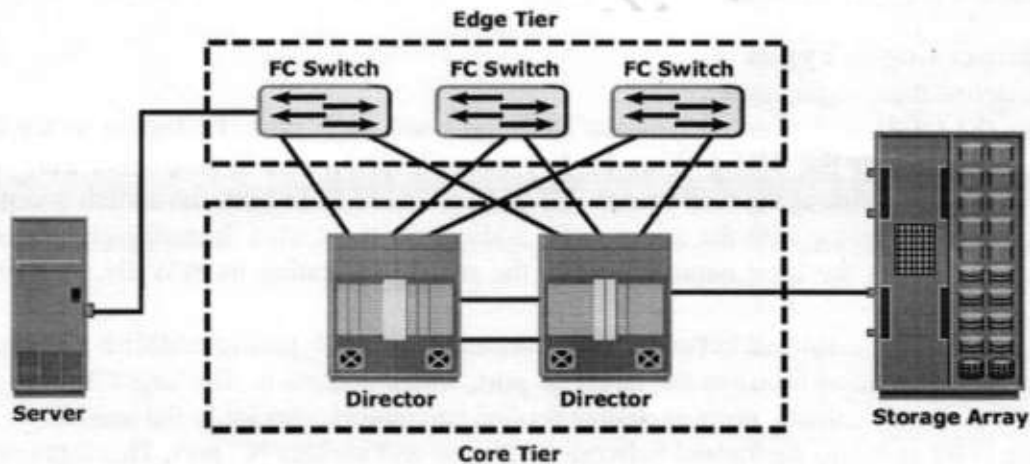


Figure 6-22: Dual-core topology

Benefits and Limitations of Core-Edge Fabric

Benefits

1. The core-edge fabric provides one-hop storage access to all storage in the system. Because each tier's switch is used for either storage or hosts, one can easily identify which resources are approaching their capacity, making it easier to develop a set of rules for scaling and apportioning.
2. A well-defined, easily reproducible building-block approach makes rolling out new fabrics easier. Core-edge fabrics can be scaled to larger environments by linking core switches, adding more core switches, or adding more edge switches.
3. This method can be used to extend the existing simple core-edge model or to expand the fabric into a compound or complex core-edge model.

Limitations

1. The core-edge fabric may lead to some performance-related problems because scaling a core-edge topology involves increasing the number of ISLs in the fabric. As more edge switches are added, the domain count in the fabric increases.

2. A common best practice is to keep the number of host-to-storage hops unchanged, at one hop, in a core-edge. Hop count represents the total number of devices a given piece of data (packet) passes through. Generally a large hop count means greater the transmission delay between data traverse from its source to destination.
3. As the number of cores increases, it may be prohibitive to continue to maintain ISLs from each core to each edge switch. When this happens, the fabric design can be changed to a compound or complex core-edge design.

Mesh Topology

In a *mesh topology*, each switch is directly connected to other switches by using ISLs. This topology promotes enhanced connectivity within the SAN. When the number of ports on a network increases, the number of nodes that can participate and communicate also increases.

A mesh topology may be one of the two types: full mesh or partial mesh. In a *full mesh*, every switch is connected to every other switch in the topology. Full mesh topology may be appropriate when the number of switches involved is small.

In a *partial mesh* topology, several hops or ISLs may be required for the traffic to reach its destination. Hosts and storage can be located anywhere in the fabric, and storage can be localized to a director or a switch in both mesh topologies.

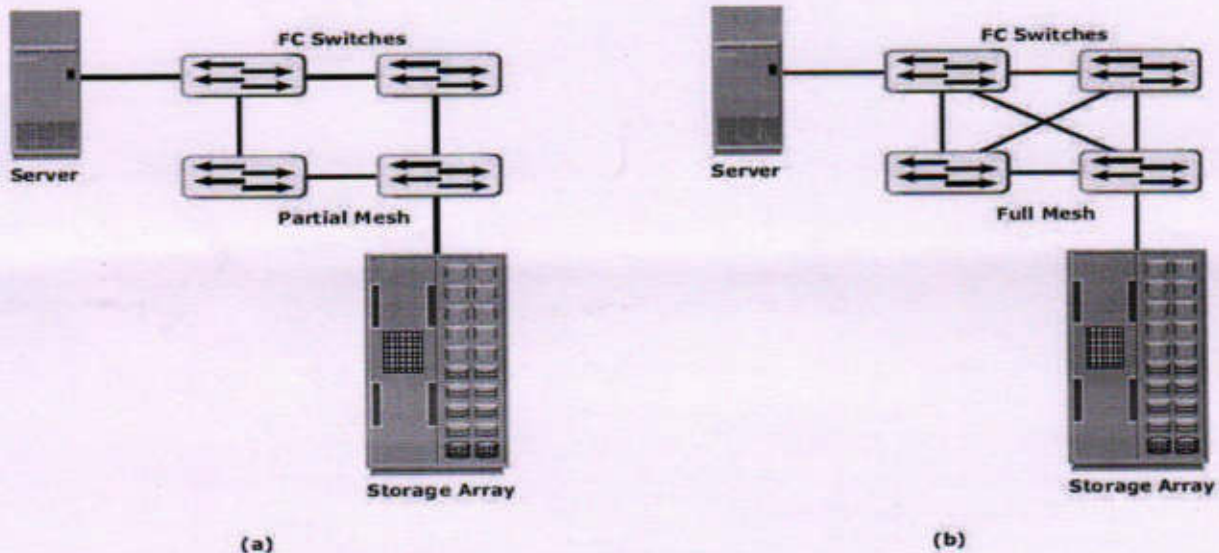
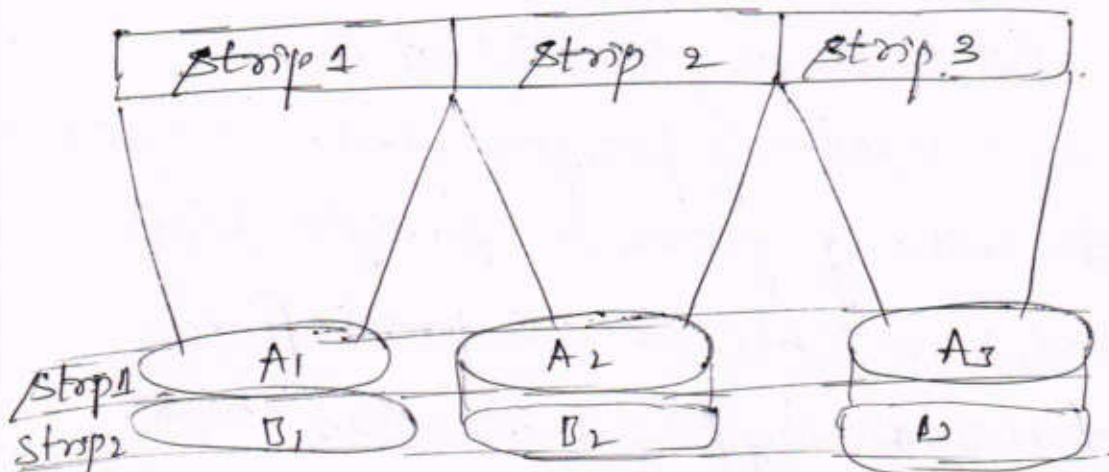
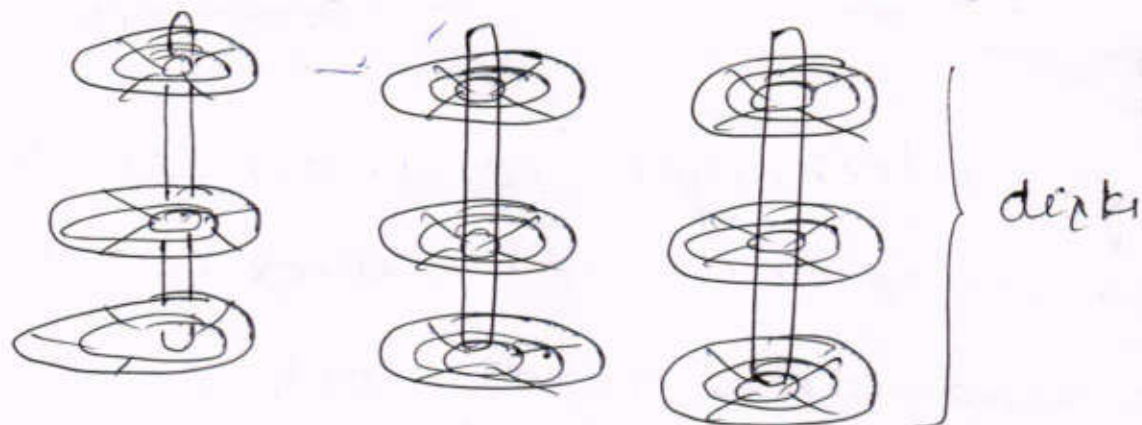


Figure 6-23: Partial mesh and full mesh topologies

Striping:-

- * Striping is a technique to spread data across multiple drives to use the drives in parallel
- * All the Read-write heads work simultaneously allowing more data to be processed in a shorter time & increasing performance compared to reading and writing from a single disk.
- * Within each disk in RAID sets, a predefined number of contiguously addressable disk blocks are defined as a strip.
- * Strip size describes the number of blocks in a strip and is the maximum amount of data that can be written to or read from a single disk in the set assuming that the accessed data starts at the beginning of the strip
- * All strips in a stripe have the same number of blocks
- * Having a smaller strip size means that data is broken into smaller pieces while spread across the disks.



Mirroring:-

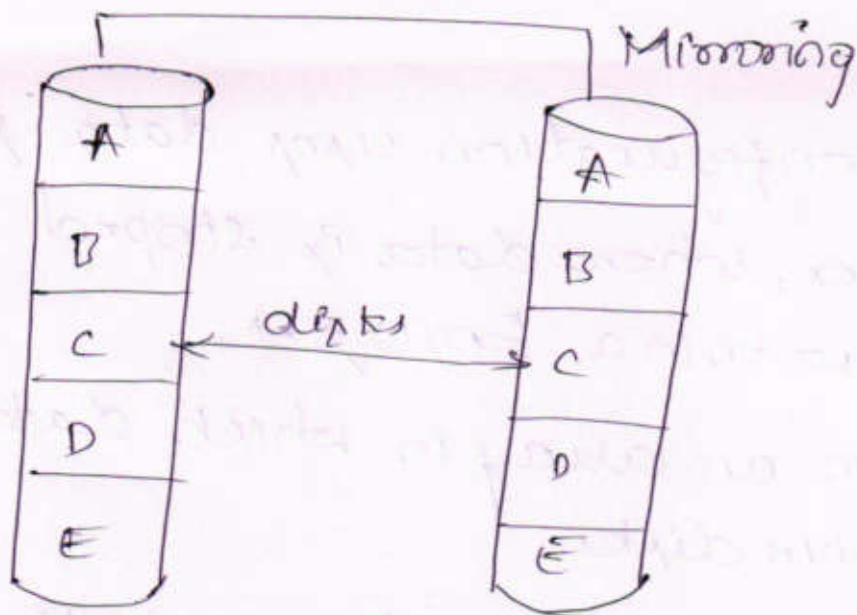
* Mirroring is a technique whereby the same data is stored on two different disk drives yielding two copies of data. If one disk drive fails, the data is intact on the surviving disk drives.

* When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair.

* In addition to providing complete data redundancy, mirroring enables fast recovery from disk failure.

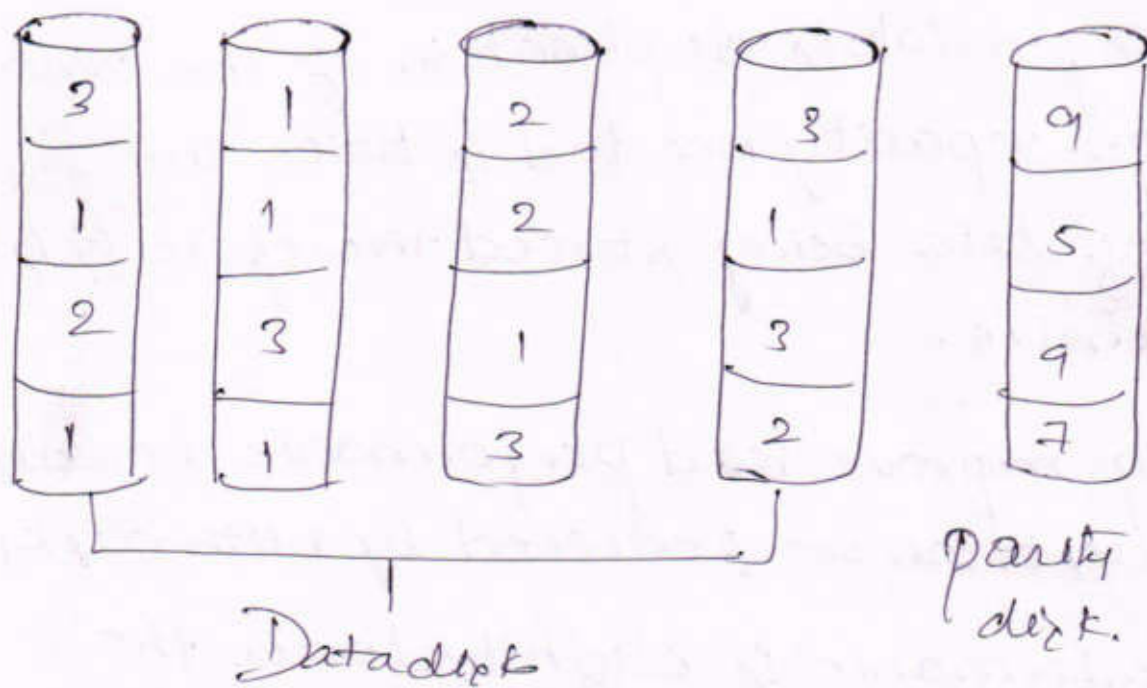
Limitation

- * Mirroring involves duplication of the amount of storage capacity needed is twice the amount of data being stored therefore it is expensive.
- * Mirroring improve read performance because read requests can be serviced by both disks
- * Write performance is slightly lower than that in single disk because write request manifests as two writes on the disk drives



parity:

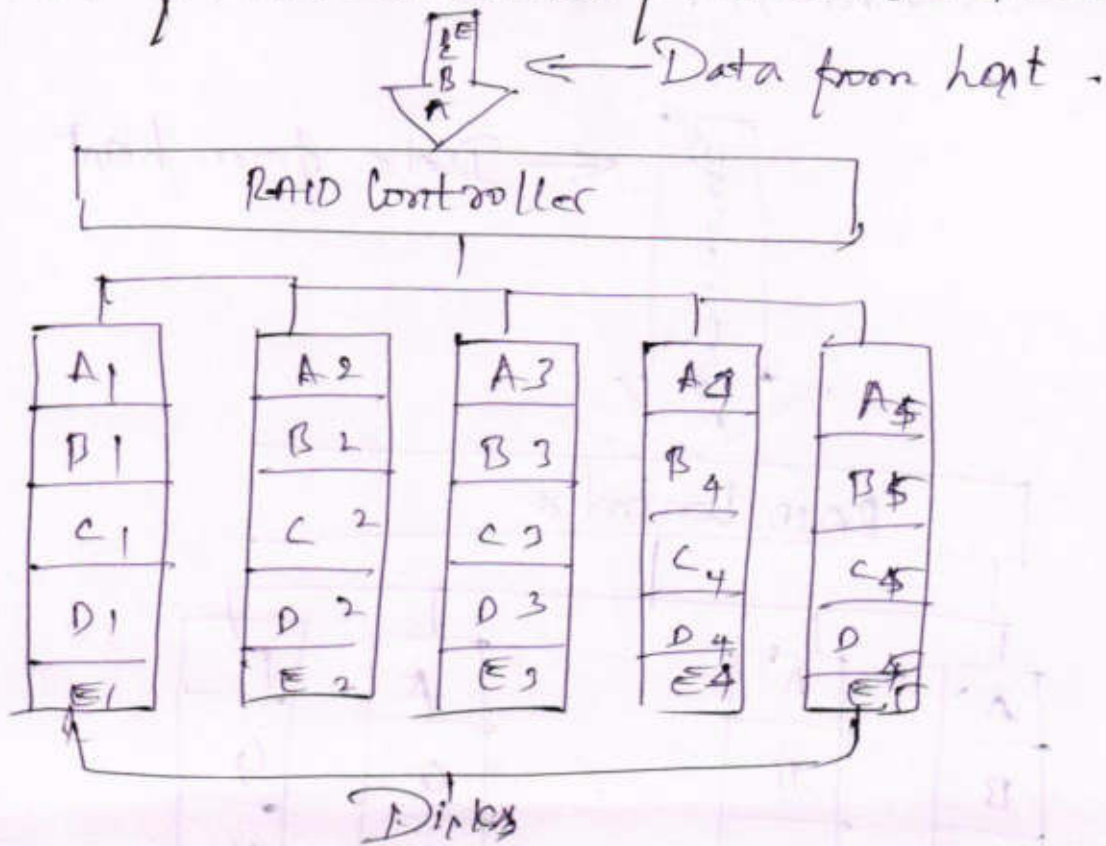
- * parity is a redundancy technique that ensures protection of data without maintaining a full set of duplicate data
- * Calculation of parity is a function of RAID controller



RAID 0

- * RAID 0 configuration uses data striping technique, where data is striped across all the disks within a RAID set.
- * RAID 0 is an array in which data is striped across five disks.
- * When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously.
- * RAID 0 is a good option for applications that need high I/O throughput.

* If these applications require high availability during drive failures, RAID 0 does not provide data protection & availability.



RAID 1

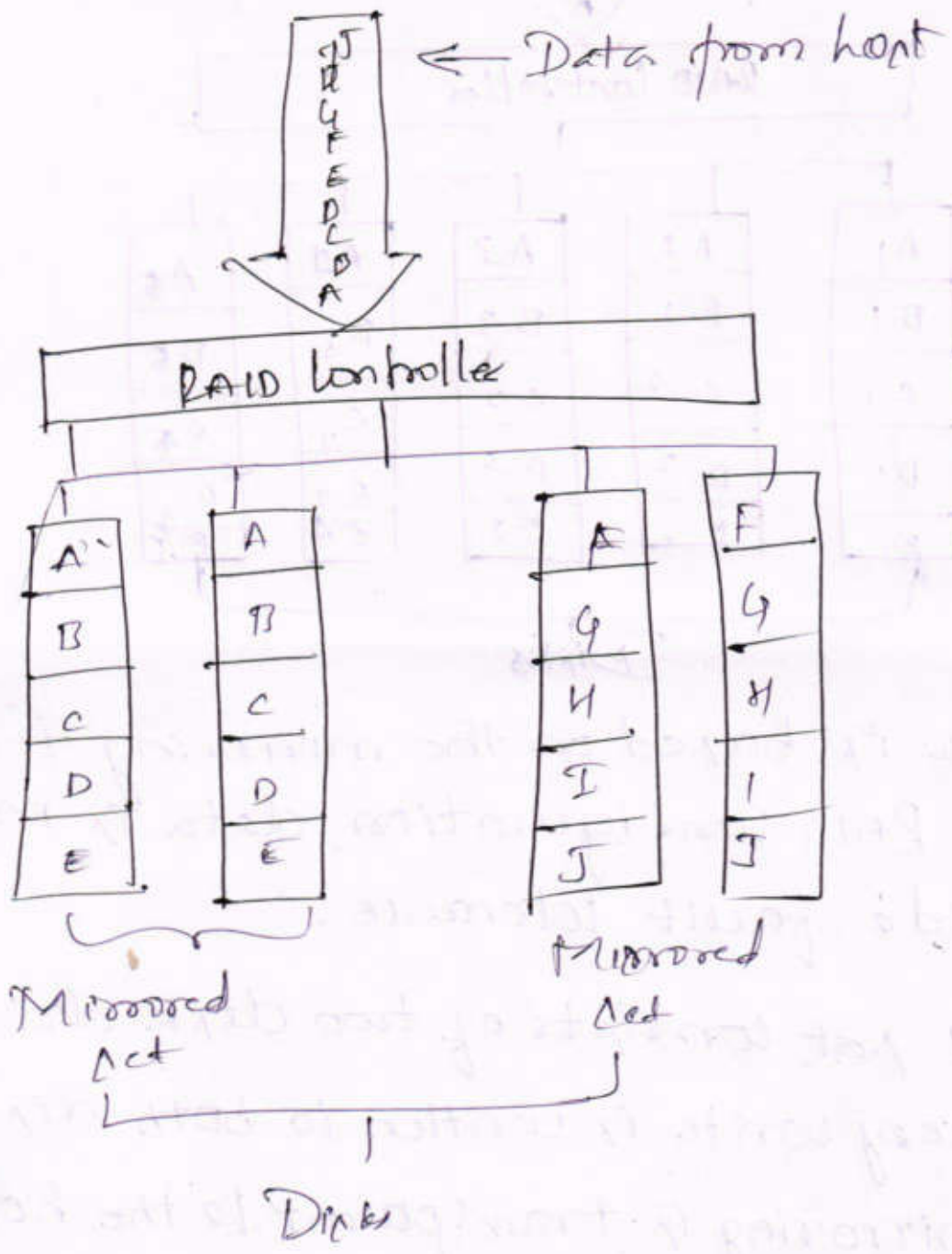
* RAID 1 is based on the mirroring technique. In this, RAID configuration, data is mirrored to provide fault tolerance.

* RAID 1 set consists of two disk drives and every write is written to both disks.

* The mirroring is transparent to the host.

* During disk failure, the impact on data recovery in RAID 1 is the least among all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery.

* RAID 1 is suitable for applications that require high availability and cost is no constraint.



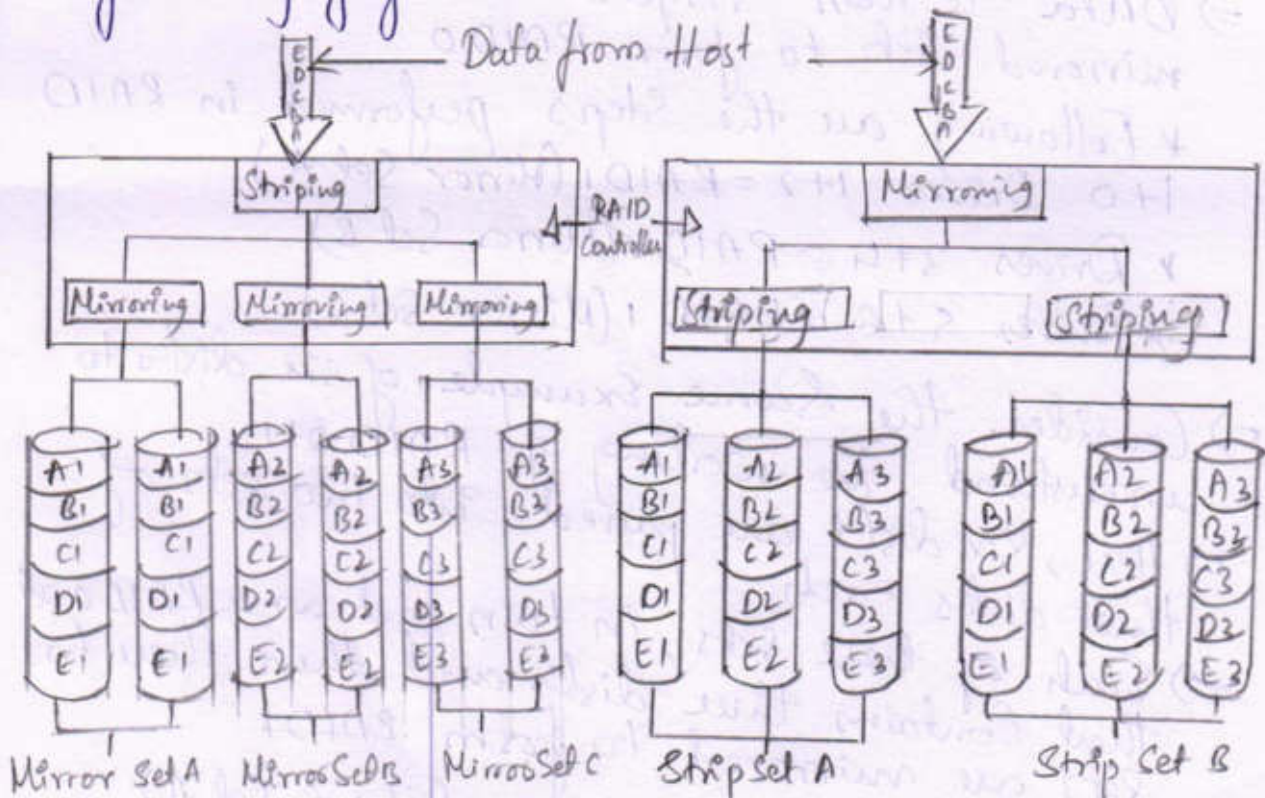
Nested RAID:

* Most data centers require data redundancy and performance from their RAID arrays.

* RAID 1+0 and RAID 0+1 combine the performance benefits of RAID 0 with the redundancy benefits of RAID 1.

* They use striping and mirroring techniques and combine their benefits.

* These types of RAID require an even number of disks, the minimum being four as show in the following figure.



(a) RAID 1+0

(b) RAID 0+1

Nested RAID

* RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. Similarly, RAID 0+1 is also known as RAID 0/1 or RAID 0/1.

* Some applications that benefit from RAID 1+0 include the following:

* High transaction rate online transaction processing (OLTP)

* Large messaging installations

* Database applications with random access workloads

⇒ When replacing a failed drive, only the mirror is rebuilt.

⇒ RAID 1+0, Consider an Example of six disks forming a RAID 1+0 (RAID 1 first and then RAID 0) set. There six disks are paired into three sets of two disks, where each set acts as RAID 1 set (mirrored pair of disks).

⇒ Data is then striped across all the three mirrored sets to form RAID 0

* Following are the steps performed in RAID 1+0
Drives 1+2 = RAID 1 (Mirror Set A)

* Drives 3+4 = RAID 1 (Mirror Set B)

* Drives 5+6 = RAID 1 (Mirror Set C)

⇒ Consider the same Example of six disks to understand the working of RAID 0+1.

⇒ Here, six disks are paired into two sets of three disks each.

⇒ Each of these sets, in turn, act as a RAID 0 set that contains three disks and then these two sets are mirrored to form RAID 1

* Drives 1+2+3 = RAID 0 (Stripe Set A)

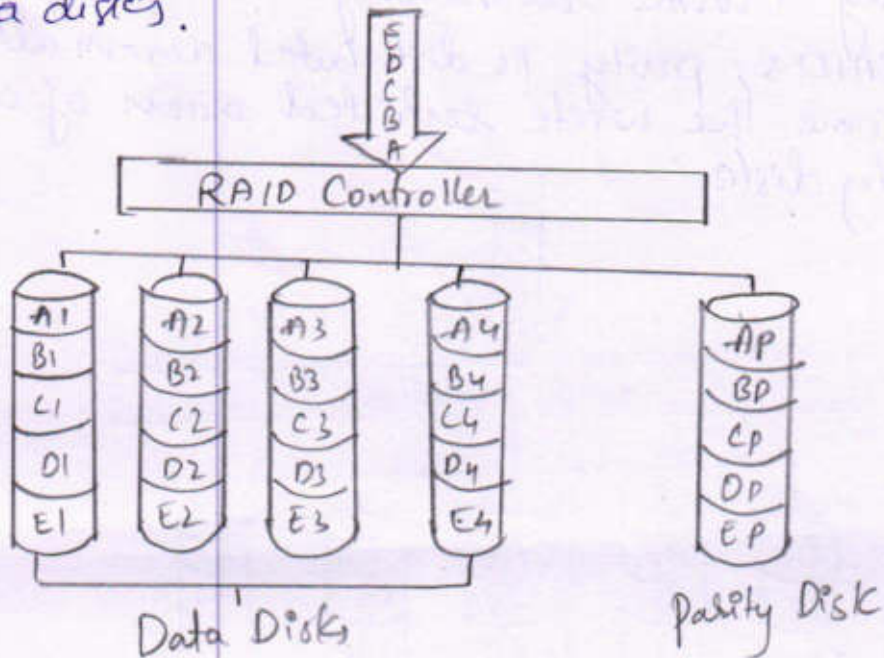
* Drives 4+5+6 = RAID 0 (Stripe Set B)

RAID 3:

⇒ RAID 3 stripes data for performance and uses parity for fault tolerance.

⇒ parity information is stored on a dedicated drive so that the data can be reconstructed if a drive fails in a RAID set.

⇒ For example, in a set of five disks, four are used for data and one for parity. Therefore, the total disk space required is 1.25 times the size of the data disks.



RAID 3

Advantage: RAID 3 provides good performance for applications that involve large sequential data access, such as data backup or video streaming.

RAID 4

⇒ Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance.

⇒ Data is striped across all disks except the parity disk in the array.

⇒ parity information is stored on a dedicated disk so that the data can be rebuilt if a drive fails.

⇒ Data disks in RAID 4 can be accessed independently so that specific data elements can be read or written on a single disk without reading or writing an entire stripe.

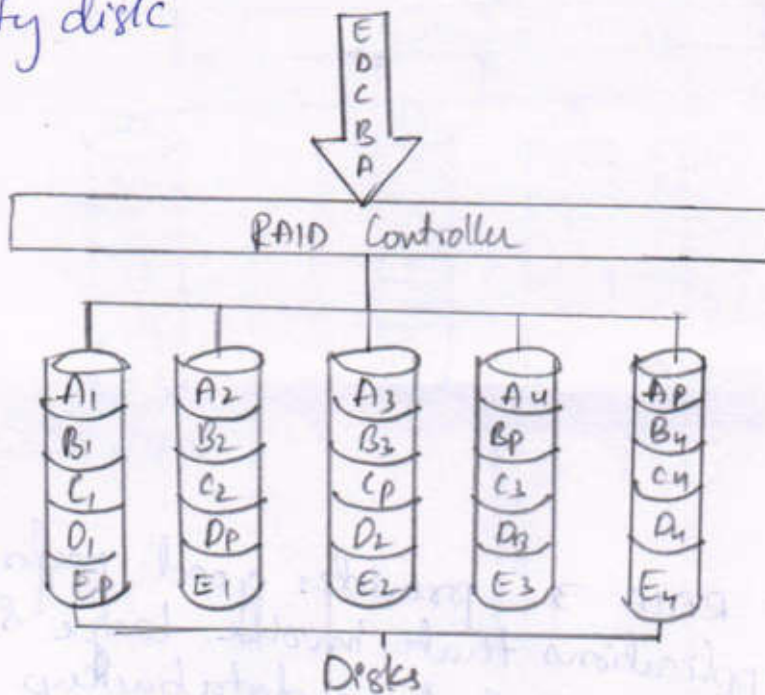
• RAID 5:

⇒ RAID 5 is a Versatile RAID implementation. It is similar to RAID 4 because it uses striping. The drives are also independently accessible.

⇒ The difference between RAID 4 and RAID 5 is the parity location.

⇒ In RAID 4, parity is written to a dedicated drive, creating a write restricted for the parity disc.

⇒ In RAID 5, parity is distributed across all disks to overcome the write restricted access of a dedicated parity disc.

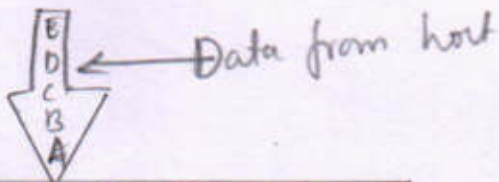


• RAID 6:

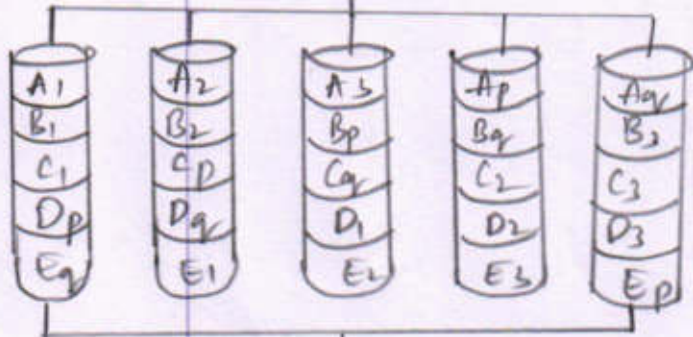
⇒ RAID 6 works the same way as RAID 5, except that RAID 6 includes a second parity element to enable survival of two disk failures occur in a RAID set.

⇒ Dis Advantage:

The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.



RAID Controller



Disks

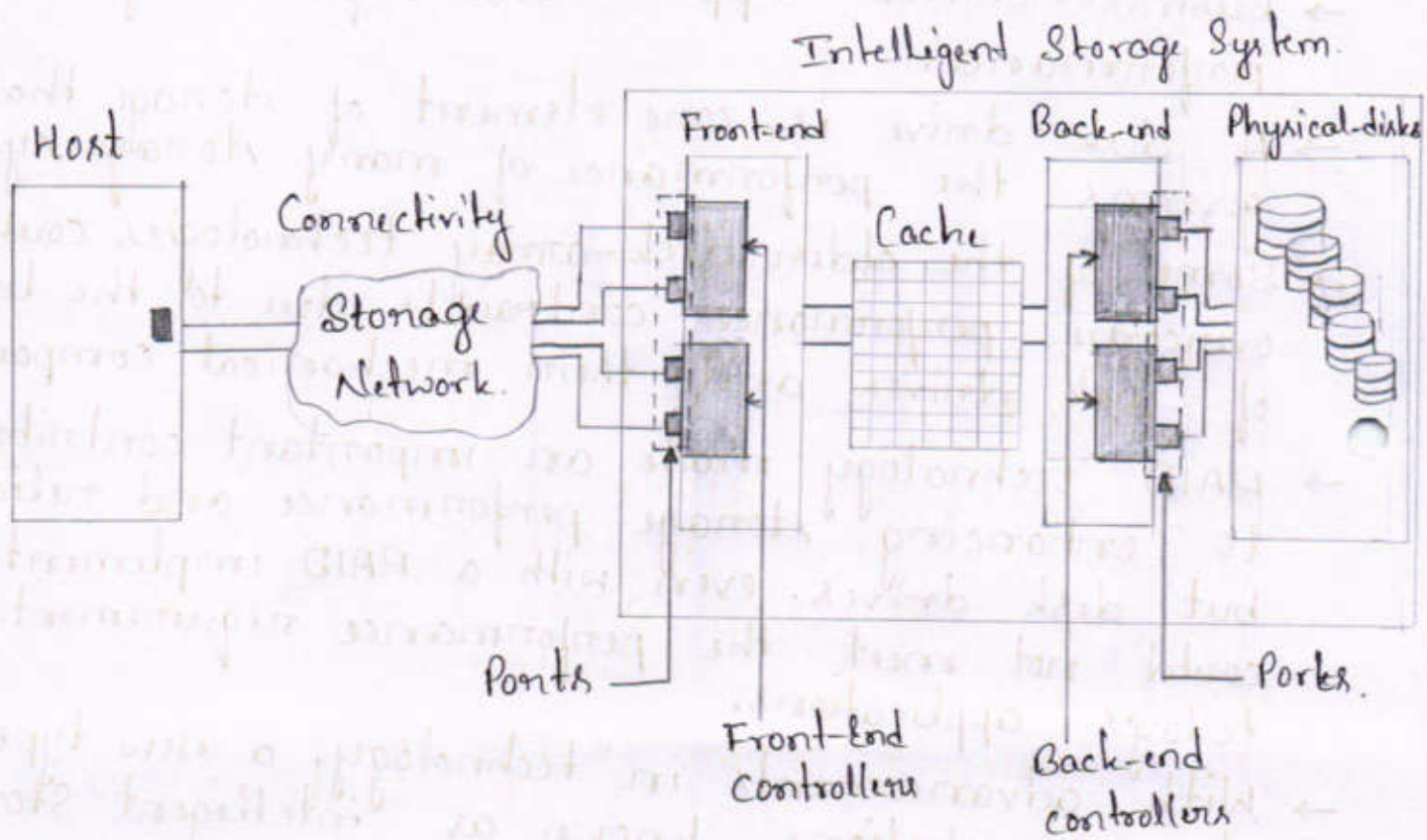
Intelligent Storage Systems.

- Business-critical applications require high levels of performance.
- A disk drive is core element of storage that governs the performance of many storage system.
- Some of the older disk-array technologies could not overcome performance constraints due to the limitation of disk drives and their mechanical components.
- RAID technology made an important contribution to enhancing storage performance and reliability, but disk drives, even with a RAID implementation, could not meet the performance requirements of today's applications.
- With advancements in technology, a new type of storage solutions, known as intelligent storage systems, has evolved.
- These intelligent storage systems are feature-rich RAID arrays that provide highly optimized I/O processing capabilities.

Components of an Intelligent Storage System

- An intelligent storage system consists of four key components.
 1. Front end
 2. cache
 3. Back end
 4. Physical disks.

→ Following fig illustrates these components and their interconnections.



1. Front End

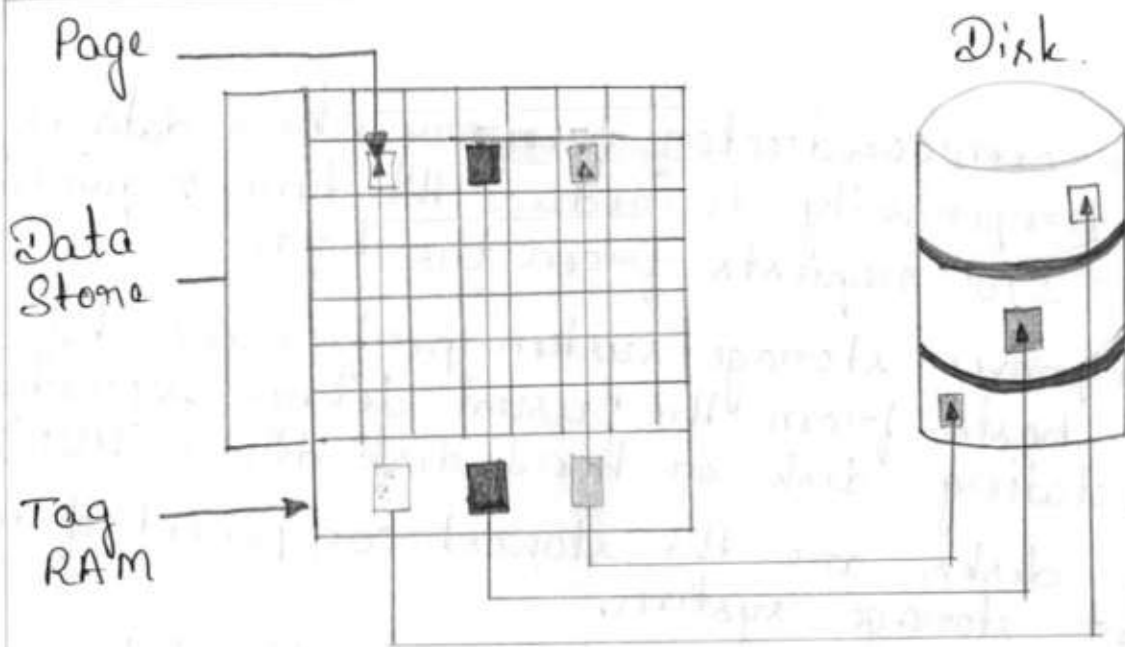
- The front end provides the interface between the storage system and the host.
- It consists of two components: Front-end ports and front-end controllers.
- Typically, a front end has controllers for high availability, and each controller contains multiple ports that enable large numbers of hosts to connect to the intelligent storage system.
- Front-end controllers route data to and from cache via the internal data bus.
- When the cache receives the write data, the controller sends an acknowledgment message back to the host.

2. Cache

- Cache is semiconductor memory where data is placed temporarily to reduce the time required to service I/O requests from the host.
- Cache improves storage system performance by isolating hosts from the usual delays associated with rotating disk on hard disk drives (HDD).
- Rotating disks are the slowest component of an intelligent storage system.
- Data access on rotating disks usually takes several milliseconds because of seek time.
- Accessing data from cache is fast and typically takes less than a millisecond.
- On intelligent arrays, write data is first placed in cache and then written to disk.

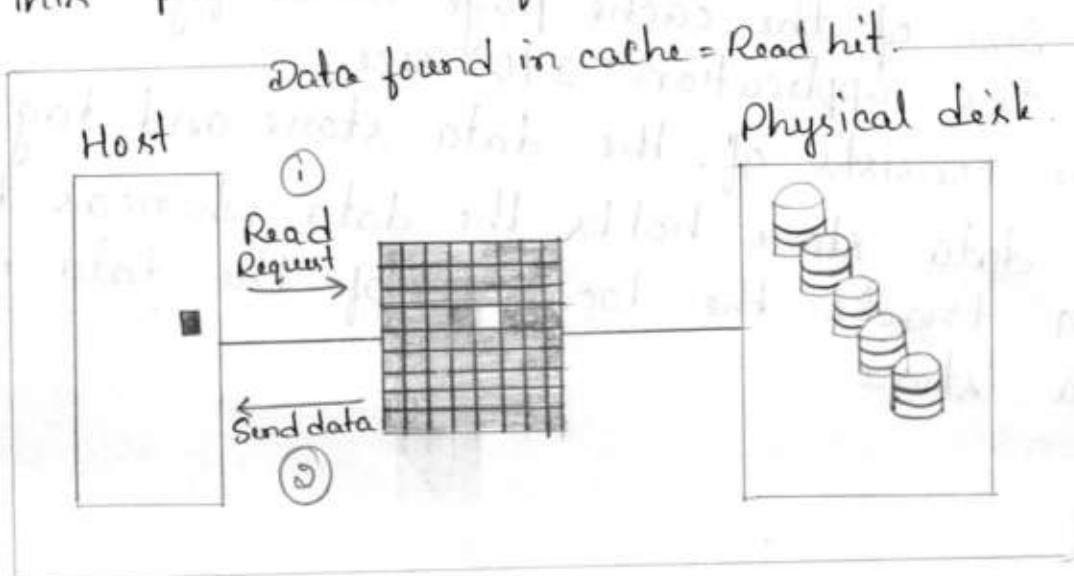
Structure of Cache

- Cache is organized into pages, which is the smallest unit of cache allocation.
- The size of the cache page is configured according to the application I/O size.
- Cache consists of the data store and tag RAM.
- The data store holds the data whereas the tag RAM tracks the location of the data in the data store.

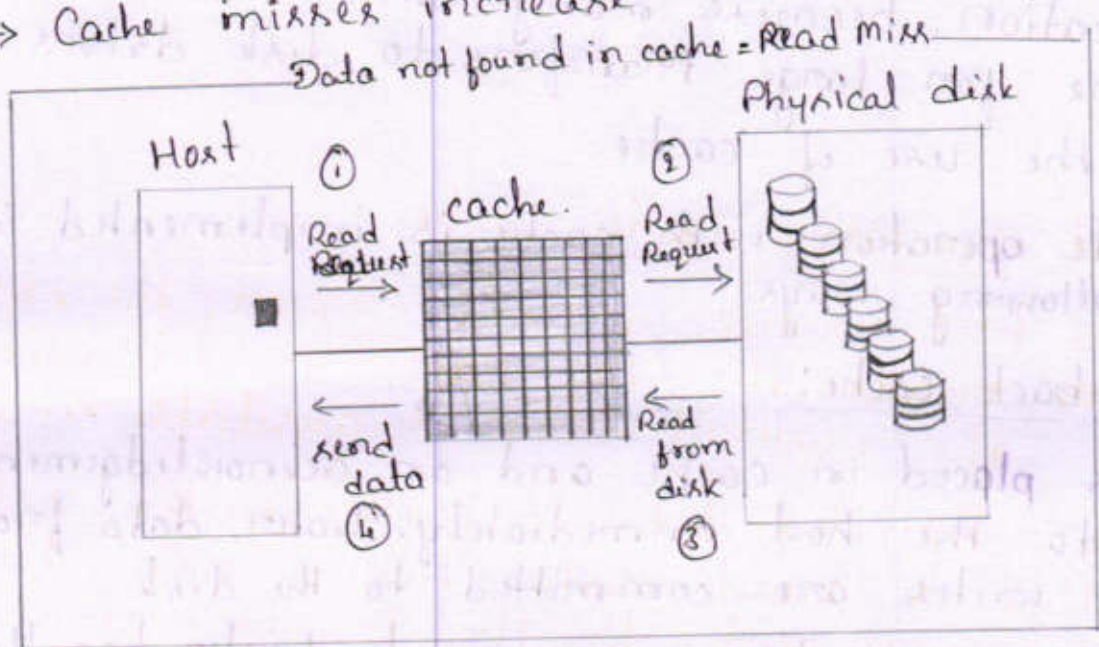


Read Operation with Cache

- When a host issues a read request, the storage controller reads the Tag RAM to determine whether the required data is available in cache.
- If the requested data is found in the cache, it is called a read cache hit or read hit and data is sent directly to the host, without any disk operation as shown in the following fig.
- This provides a fast response time to the host.



- If the requested data is not found in cache, it is called a cache miss and the data must be read from disk as show in the following fig.
- The back end accesses the appropriate disk and retrieves the requested data.
- Data is then placed in cache and finally sent to the host through the front end.
- Cache misses increase the I/O response time.



- A prefetch or read-ahead algorithm is used when read requests are sequential.
- In fixed prefetch, the intelligent storage system prefetches a fixed amount of data. It is most suitable when host I/O sizes are uniform.
- In variable prefetch, the storage system prefetches an amount of data in multiples of the size of the host request.
- Maximum prefetch, limits the number of data blocks that can be prefetched to prevent the disks from being provide busy with prefetch at the expense of other I/O's.

Write Operation with Cache.

- Write operations with cache provide performance advantages over writing directly to disks.
- When I/O is written to cache and acknowledged, it is completed in far less than it would take to write directly to disk.
- Sequential writes also offer opportunities for optimization because many smaller writes can be combine for large transfers to disk drives with the use of cache.
- * A write operation with cache is implemented in the following ways.

1. Write-back Cache:

- Data is placed in cache and an acknowledgement is sent to the host immediately. Later, data from several writes are committed to the disk.
- Write response times are much faster because the write operations are isolated from the mechanical delays of the disk.

→ 2. Write-through cache:

- Data is placed in the cache and immediately written to the disk, and a acknowledgement is sent to the host.
- Because data is committed to disk as it arrives, the risks of data loss are low, but the write-response time is longer because of the disk operations.

* Cache Implementation:

- * cache can be implemented as either dedicated cache (or) global cache.
- * In global cache, both reads and writes can use any of the available memory addresses.
- * Cache management is more efficient in global cache implementation because only one global set of addresses has to be managed.
- * Global cache allows users to specify the percentages of cache available for reads and writes for cache management.
- * Global cache implementations, the ratio of cache available for reads versus writes is dynamically adjusted based on the workload.

* Cache Management:

- * Cache is a finite and expensive resource that needs proper management.
- * Even though modern intelligent storage systems come with a large amount of cache, when all cache pages are filled, some pages have to be free up to accomodate new data and avoid performance refuse.
- * Various cache management algorithms are implemented in intelligent storage systems to proactively maintain a set of free pages and a list of pages that can be potentially free up whenever required.

②

* The most commonly used algorithms are discussed in following list:

1. Least Recently Used (LRU):

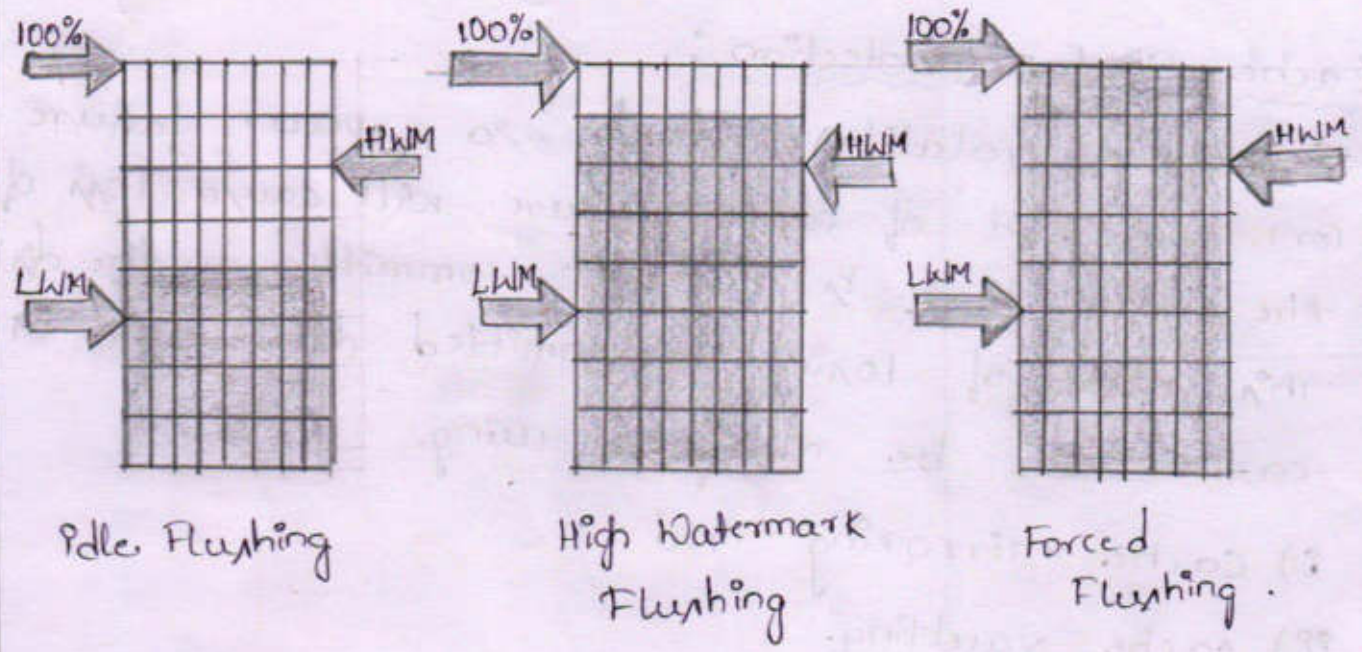
- * An algorithm that continuously monitors data access in cache and identifies the cache pages that have not been accessed for a long time.
- * LRU either frees up these pages (or) marks them for reuse.
- * This algorithm is based on the assumption that data that has not been accessed for a while will not be requested by host.

2. Most Recently Used (MRU):

- * This algorithm is the opposite of LRU, where the pages that have been accessed most recently are freed up (or) marked for reuse.
- * This algorithm is based on the assumption that recently accessed data may not be required for a while.
- * As cache fills, the storage system must take action to flush dirty pages to manage space availability.
- * Flushing is the process that give data from cache to the disk.
- * On the basis of the I/O access rate and model, high and low levels called watermarks are set in cache to manage the flushing process.

* High WaterMark (HWM) is the cache utilization level at which the storage system starts high-speed flushing cache data.

* Low WaterMark (LWM) is the point at which the storage system stops flushing data to the disks.



LWM = Low WaterMark
HWM = High WaterMark.

* Idle flushing: Occurs continuously, at a modest rate, when the cache utilization level is between the high and low watermark.

* High watermark flushing: Activated when cache utilization hits the high watermark. The storage system dedicates some additional resources for flushing. This type of flushing has some crash on I/O processing.

* Forced flushing: Occurs in the event of a large I/O burst when cache reaches 100 percent of its capacity, which significantly affects the I/O response time. In forced flushing, system flushes the cache on priority by allocating more resources.

* Cache Data Protection:

* Cache is volatile memory, so a power failure (or) any kind of cache failure will cause loss of the data that is not yet committed to the disk. This risk of losing uncommitted data held in cache can be mitigated using.

- i) cache mirroring and
- ii) cache vaulting.

i) Cache mirroring:

* Each write to cache is held in two different memory locations on two independent memory cards. If a cache failure occurs, the write data will still be safe in the mirrored location and can be committed to the disk.

* Reads are staged from the disk to the cache; therefore, if a cache failure occurs, the data can still be accessed from the disk.

* In cache mirroring approaches, the problem of maintaining cache coherency is introduced.

* Cache coherency means that data in two different cache locations must be identical at all times.

ii] Cache Vaulting:

* The risk of data loss due to power failure can be addressed in various ways.

* Using battery power to write the cache content to the disk.

* If an extended power failure occurs, using batteries is not a viable option. This is because in intelligent storage systems, large amounts of data might need to be committed to numerous disks, and batteries might not provide power for sufficient time to write each piece of data to its intended disk.

* Therefore, storage vendors use a set of physical disks to dump the contents of cache during power failure.

* This is called cache vaulting and the disks are called vault drives.

* Back End:

* The back end provides an interface between cache and physical disks.

* It consists of two components: back-end ports and back-end controllers.

* The back end controls data transfers between cache and physical disks.

* From cache, data is sent to the back end and then routed to the destination disk.

* Physical disks are connected to ports on the back end.

* The back-end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage.

* Physical Disk :

* Physical disks are connected to the back-end storage controller and provide stable data storage.

* Modern intelligent storage systems provide support to a variety of disk drives with different speeds and types.

Module-3IP SAN and FCoEiSCSI

- iSCSI is an IP based protocol that establishes and manages connections between host and storage over IP, as shown in Fig below.
- iSCSI encapsulates SCSI commands and data into an IP packet and transports them using TCP/IP.
- iSCSI is widely adopted for connecting servers to storage because it is relatively inexpensive and easy to implement, especially in environments in which an FC SAN does not exist.

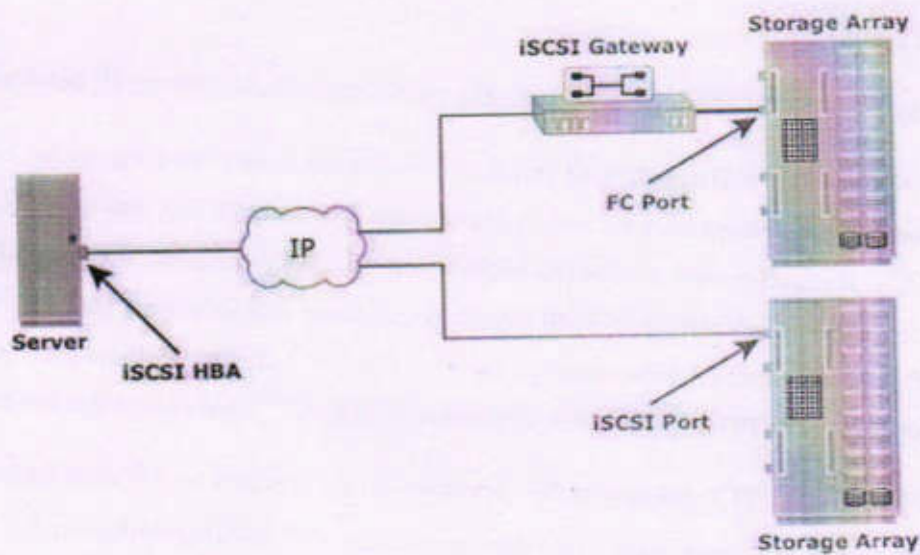


Fig : iSCSI implementation

Components of iSCSI

- An initiator (host), target (storage or iSCSI gateway), and an IP-based network are the key

iSCSI components.

- If an iSCSI-capable storage array is deployed, then a host with the iSCSI initiator can directly communicate with the storage array over an IP network.
- However, in an implementation that uses an existing FC array for iSCSI communication, an iSCSI gateway is used.
- These devices perform the translation of IP packets to FC frames and vice versa, thereby bridging the connectivity between the IP and FC environments.

iSCSI Host Connectivity

The three iSCSI host connectivity options are:

- A standard NIC with software iSCSI initiator.
 - a TCP offload engine (TOE) NIC with software iSCSI initiator,
 - an iSCSI HBA
- The function of the iSCSI initiator is to route the SCSI commands over an IP network.
 - A **standard NIC with a software iSCSI** initiator is the simplest and least expensive connectivity option. It is easy to implement because most servers come with at least one, and in many cases two, embedded NICs. It requires only a software initiator for iSCSI functionality. Because NICs provide standard IP function, encapsulation of SCSI into IP packets and decapsulation are carried out by the host CPU. This places additional overhead on the host CPU. If a standard NIC is used in heavy I/O load situations, the host CPU might become a bottleneck. TOE NIC helps reduce this burden.
 - A **TOE NIC** offloads TCP management functions from the host and leaves only the iSCSI functionality to the host processor. The host passes the iSCSI information to the TOE card, and the TOE card sends the information to the destination using TCP/IP. Although this solution improves performance, the iSCSI functionality is still handled by a software initiator that requires host CPU cycles.
 - An **iSCSI HBA** is capable of providing performance benefits because it offloads the entire

iSCSI and TCP/IP processing from the host processor. The use of an iSCSI HBA is also the simplest way to boot hosts from a SAN environment via iSCSI. If there is no iSCSI HBA, modifications must be made to the basic operating system to boot a host from the storage devices because the NIC needs to obtain an IP address before the operating system loads. The functionality of an iSCSI HBA is similar to the functionality of an FC HBA.

iSCSI Topologies

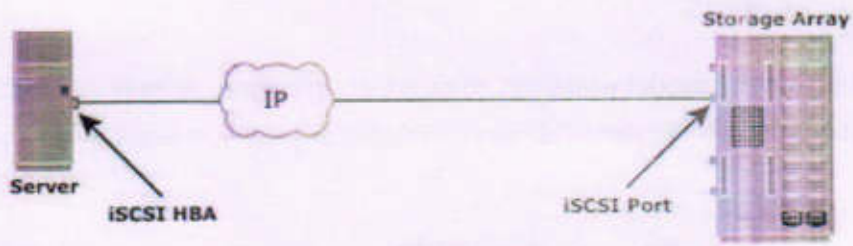
- Two topologies of iSCSI implementations are **native and bridged**.
- Native topology does not have FC components.
- The initiators may be either directly attached to targets or connected through the IP network.
- Bridged topology enables the coexistence of FC with IP by providing iSCSI-to-FC bridging functionality.
- For example, the initiators can exist in an IP environment while the storage remains in an FC environment.

Native iSCSI Connectivity

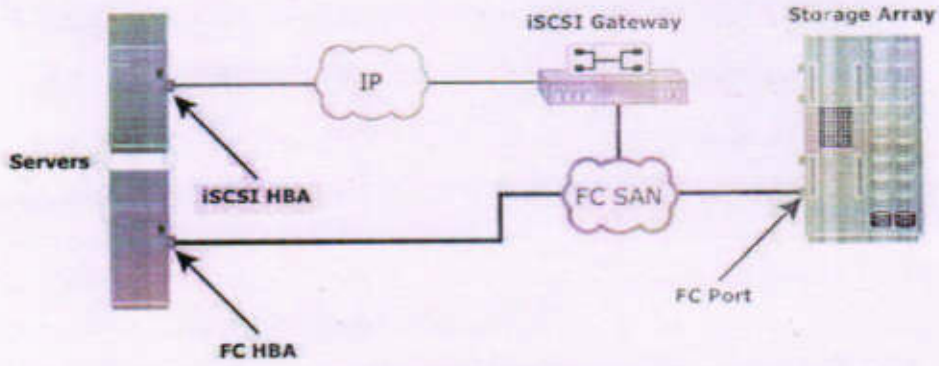
- FC components are not required for iSCSI connectivity if an iSCSI-enabled array is deployed.
- In Fig (a), the array has one or more iSCSI ports configured with an IP address and is connected to a standard Ethernet switch.
- After an initiator is logged on to the network, it can access the available LUNs on the storage array.
- A single array port can service multiple hosts or initiators as long as the array port can handle the amount of storage traffic that the hosts generate.

Bridged iSCSI Connectivity

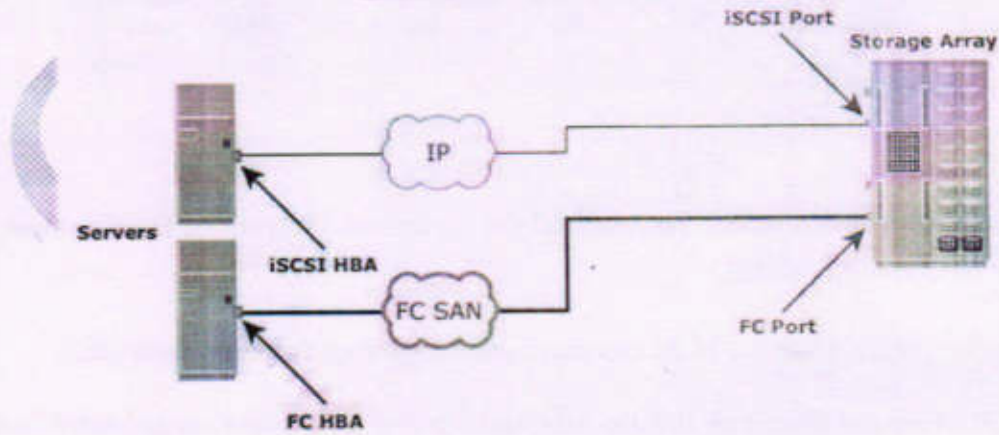
- A bridged iSCSI implementation includes FC components in its configuration.
- Fig (b), illustrates iSCSI host connectivity to an FC storage array. In this case, the array does not have any iSCSI ports. Therefore, an external device, called a gateway or a multiprotocol router, must be used to facilitate the communication between the iSCSI host and FC storage.
- The gateway converts IP packets to FC frames and vice versa.
- The bridge devices contain both FC and Ethernet ports to facilitate the communication between the FC and IP environments.
- In a bridged iSCSI implementation, the iSCSI initiator is configured with the gateway's IP address as its target destination.
- On the other side, the gateway is configured as an FC initiator to the storage array.
- **Combining FC and Native iSCSI Connectivity:** The most common topology is a combination of FC and native iSCSI. Typically, a storage array comes with both FC and iSCSI ports that enable iSCSI and FC connectivity in the same environment, as shown in Fig (c).



(a) Native iSCSI Connectivity



(b) Bridged iSCSI Connectivity



(c) Combining FC and Native iSCSI Connectivity

Fig : iSCSI Topologies

iSCSI Protocol Stack

- Fig 2.23 displays a model of the iSCSI protocol layers and depicts the encapsulation order of the SCSI commands for their delivery through a physical carrier.

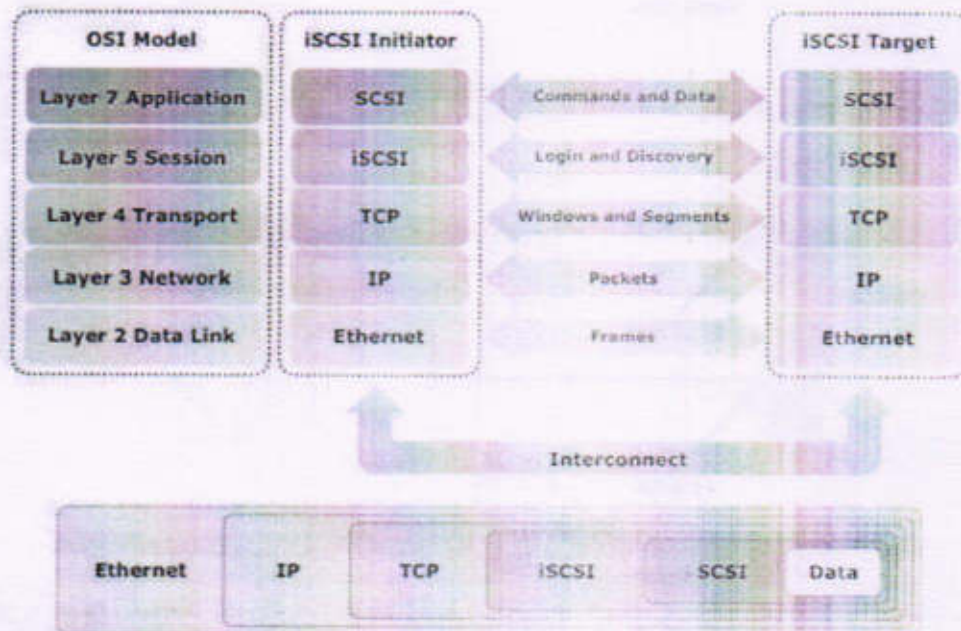


Fig 2.23: iSCSI protocol stack

- SCSI is the command protocol that works at the application layer of the Open System Interconnection (OSI) model.
- The initiators and targets use SCSI commands and responses to talk to each other.
- The SCSI command descriptor blocks, data, and status messages are encapsulated into TCP/IP and transmitted across the network between the initiators and targets.
- iSCSI is the session-layer protocol that initiates a reliable session between devices that recognize SCSI commands and TCP/IP.
- The iSCSI session-layer interface is responsible for handling login, authentication, target discovery, and session management.

- TCP is used with iSCSI at the transport layer to provide reliable transmission.
- TCP controls message flow, windowing, error recovery, and retransmission.
- It relies upon the network layer of the OSI model to provide global addressing and connectivity.
- The Layer 2 protocols at the data link layer of this model enable node-to-node communication through a physical network.

iSCSI PDU

- A *protocol data unit* (PDU) is the basic “information unit” in the iSCSI environment.
- The iSCSI initiators and targets communicate with each other using iSCSI PDUs. This communication includes establishing iSCSI connections and iSCSI sessions, performing iSCSI discovery, sending SCSI commands and data, and receiving SCSI status.
- All iSCSI PDUs contain one or more header segments followed by zero or more data segments.
- The PDU is then encapsulated into an IP packet to facilitate the transport.
- A PDU includes the components shown in Fig below.
- The IP header provides packet-routing information to move the packet across a network.
- The TCP header contains the information required to guarantee the packet delivery to the target.
- The iSCSI header (basic header segment) describes how to extract SCSI commands and data for the target. iSCSI adds an optional CRC, known as the *digest*, to ensure datagram integrity. This is in addition to TCP checksum and Ethernet CRC.
- The header and the data digests are optionally used in the PDU to validate integrity and data placement.

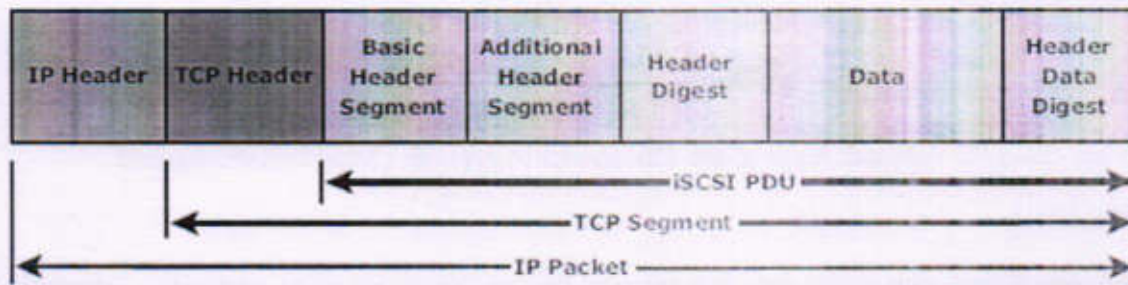


Fig : iSCSI PDU encapsulated in an IP packet

iSCSI Discovery

- An initiator must discover the location of its targets on the network and the names of the targets available to it before it can establish a session.
- This discovery can take place in two ways:
 - **SendTargets discovery**
 - **internet Storage Name Service (iSNS).**
- In *SendTargets discovery*, the initiator is manually configured with the target's network portal to establish a discovery session. The initiator issues the SendTargets command, and the target network portal responds with the names and addresses of the targets available to the host.
- iSNS (Fig below) enables automatic discovery of iSCSI devices on an IP network. The initiators and targets can be configured to automatically register themselves with the iSNS server. Whenever an initiator wants to know the targets that it can access, it can query the iSNS server for a list of available targets.
- The discovery can also take place by using service location protocol (SLP). However, this is less commonly used than SendTargets discovery and iSNS.

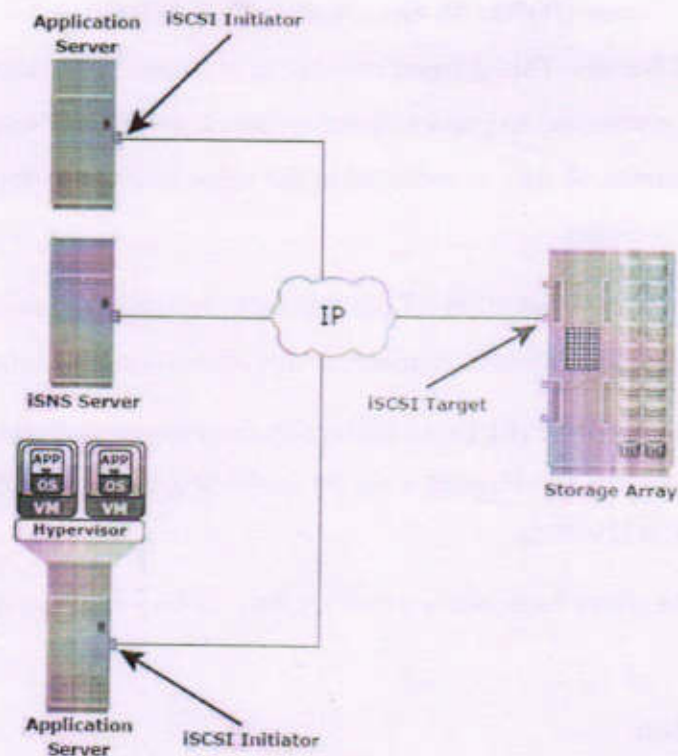


Fig : Discovery using iSNS

iSCSI Names

- A unique worldwide iSCSI identifier, known as an *iSCSI name*, is used to identify the initiators and targets within an iSCSI network to facilitate communication.
- The unique identifier can be a combination of the names of the department, application, or manufacturer, serial number, asset number, or any tag that can be used to recognize and manage the devices.
- Following are two types of iSCSI names commonly used:
 - iSCSI Qualified Name (IQN):
 - Extended Unique Identifier (EUI)

- **iSCSI Qualified Name (IQN):** An organization must own a registered domain name to generate iSCSI Qualified Names. This domain name does not need to be active or resolve to an address. It just needs to be reserved to prevent other organizations from using the same domain name to generate iSCSI names. A date is included in the name to avoid potential conflicts caused by the transfer of domain names.

An example of an IQN is `iqn.2008-02.com.example:optional_string`. The *optional_string* provides a serial number, an asset number, or any other device identifiers.

- **Extended Unique Identifier (EUI):** An EUI is a globally unique identifier based on the IEEE EUI-64 naming standard. An EUI is composed of the eui prefix followed by a 16-character hexadecimal name, such as `aseui.0300732A32598D26`.
- In either format, the allowed special characters are dots, dashes, and blank spaces.

iSCSI Session

- An iSCSI session is established between an initiator and a target, as shown in Fig.
- A session is identified by a session ID (SSID), which includes part of an initiator ID and a target ID.
- The session can be intended for one of the following:
 - The discovery of the available targets by the initiators and the location of a specific target on a network
 - The normal operation of iSCSI (transferring data between initiators and targets)
- There might be one or more TCP connections within each session. Each TCP connection within the session has a unique connection ID (CID).
- An iSCSI session is established via the iSCSI login process. The login process is started when the initiator establishes a TCP connection with the required target either via the well-known port 3260 or a specified target port.

- During the login phase, the initiator and the target authenticate each other and negotiate on various parameters.
- After the login phase is successfully completed, the iSCSI session enters the full-feature phase for normal SCSI transactions. In this phase, the initiator may send SCSI commands and data to the various LUNs on the target.
- The final phase of the iSCSI session is the connection termination phase, which is referred to as the logout procedure.
- The initiator is responsible for commencing the logout procedure; however, the target may also prompt termination by sending an iSCSI message, indicating the occurrence of an internal error condition.
- After the logout request is sent from the initiator and accepted by the target, no further request and response can be sent on that connection.

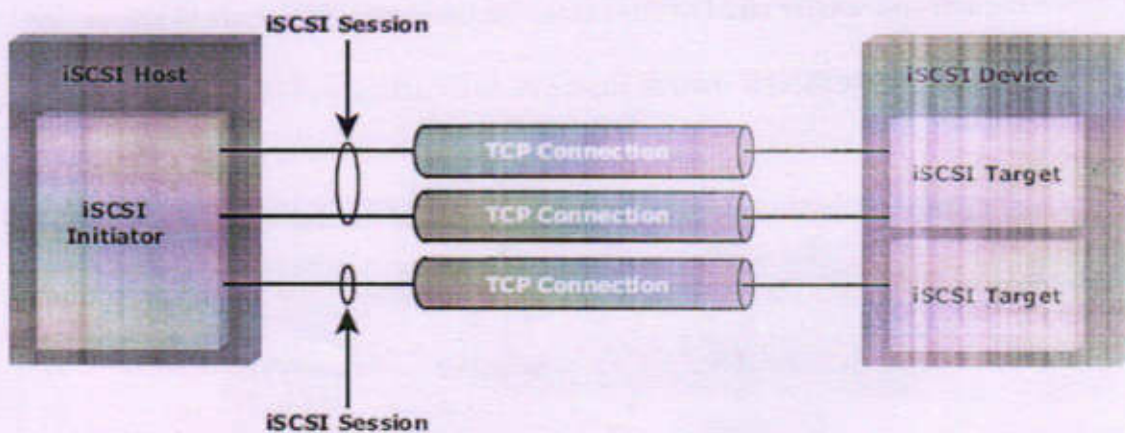


Fig : iSCSI session

Command Sequencing

- The iSCSI communication between the initiators and targets is based on the request-response command sequences.

- A command sequence may generate multiple PDUs.
- A **command sequence number (CmdSN)** within an iSCSI session is used for numbering all initiator-to-target command PDUs belonging to the session.
- This number ensures that every command is delivered in the same order in which it is transmitted, regardless of the TCP connection that carries the command in the session.
- Command sequencing begins with the first login command, and the CmdSN is incremented by one for each subsequent command.
- The iSCSI target layer is responsible for delivering the commands to the SCSI layer in the order of their CmdSN.
- Similar to command numbering, a **status sequence number (StatSN)** is used to sequentially number status responses, as shown in Fig.
- These unique numbers are established at the level of the TCP connection.
- A target sends **request-to-transfer (R2T)** PDUs to the initiator when it is ready to accept data.
- A **data sequence number (DataSN)** is used to ensure in-order delivery of data within the same command.
- The DataSN and R2TSN are used to sequence data PDUs and R2Ts, respectively.

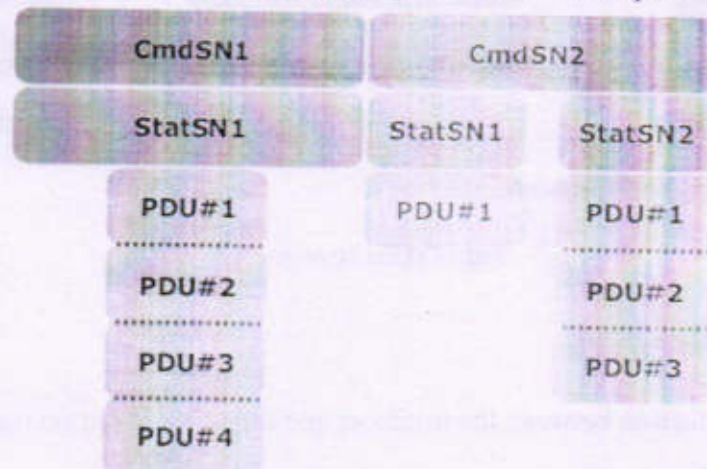


Fig : Command and status sequence number

FCIP (Fibre channel over IP)

- FCIP is a IP-based protocol that is used to connect distributed FC-SAN islands.
- Creates virtual FC links over existing IP network that is used to transport FC data between different FC SANS.
- It encapsulates FC frames into IP packet.
- It provides disaster recovery solution.

FCIP Protocol Stack

- The FCIP protocol stack is shown in Fig below. Applications generate SCSI commands and data, which are processed by various layers of the protocol stack.
- The upper layer protocol SCSI includes the SCSI driver program that executes the read-and-write commands.
- Below the SCSI layer is the Fibre Channel Protocol (FCP) layer, which is simply a Fibre Channel frame whose payload is SCSI.
- The FCP layer rides on top of the Fibre Channel transport layer. This enables the FC frames to run natively within a SAN fabric environment. In addition, the FC frames can be encapsulated into the IP packet and sent to a remote SAN over the IP.

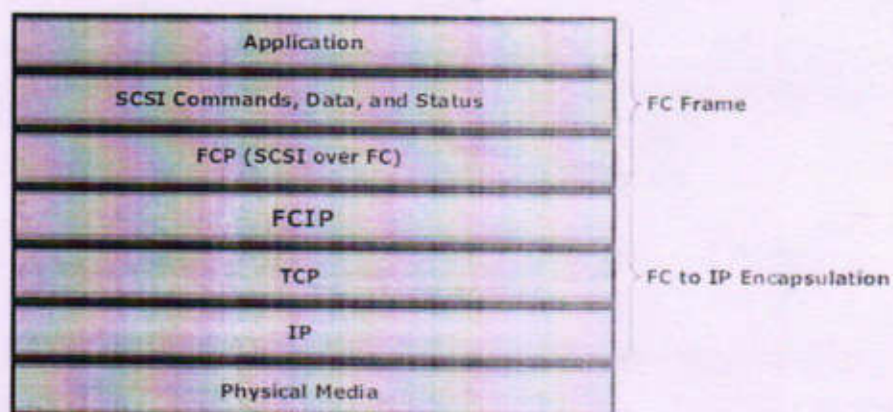


Fig : FCIP protocol stack

- The FCIP layer encapsulates the Fibre Channel frames onto the IP payload and passes them to the TCP layer (see Fig). TCP and IP are used for transporting the encapsulated information across Ethernet, wireless, or other media that support the TCP/IP traffic.
- Encapsulation of FC frame into an IP packet could cause the IP packet to be fragmented when the data link cannot support the maximum transmission unit (MTU) size of an IP packet.
- When an IP packet is fragmented, the required parts of the header must be copied by all fragments.
- When a TCP packet is segmented, normal TCP operations are responsible for receiving and re-sequencing the data prior to passing it on to the FC processing portion of the device.

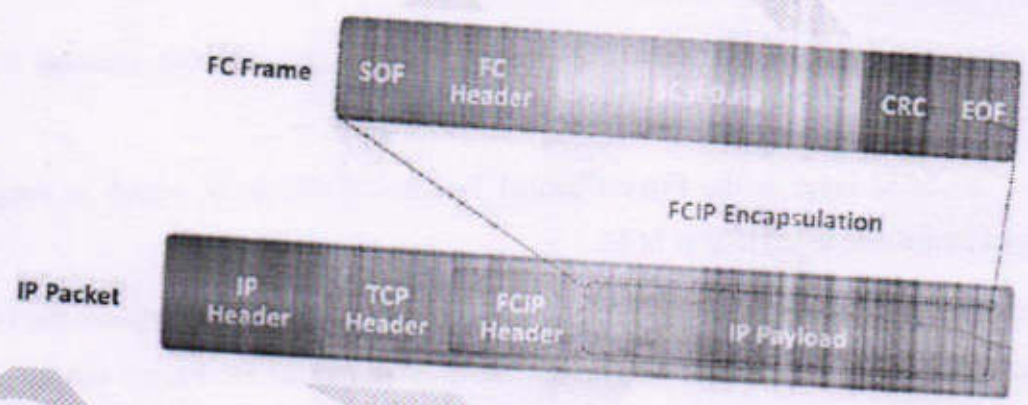


Fig FCIP encapsulation

FCIP Topology

- In an FCIP environment, an FCIP gateway is connected to each fabric via a standard FC connection (Fig).
- The FCIP gateway at one end of the IP network encapsulates the FC frames into IP packets.
- The gateway at the other end removes the IP wrapper and sends the FC data to the layer 2 fabric.
- The fabric treats these gateways as layer 2 fabric switches.
- An IP address is assigned to the port on the gateway, which is connected to an IP network. After the IP connectivity is established, the nodes in the two independent fabrics can communicate with each other.

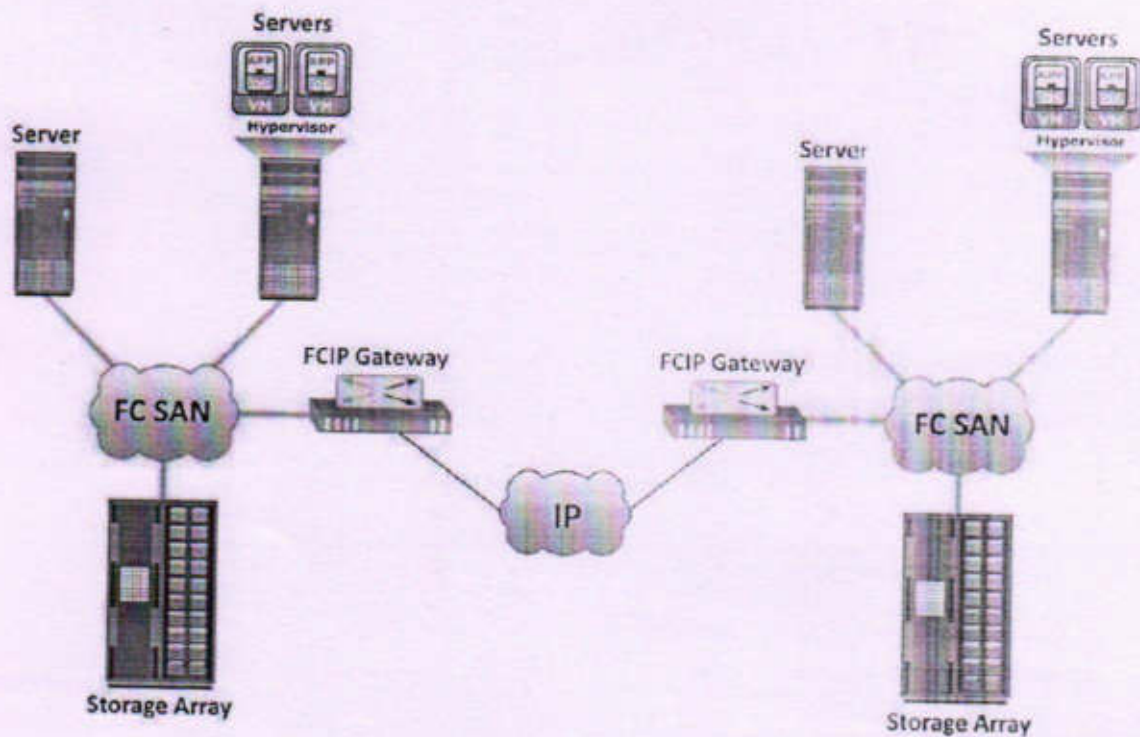


Fig : FCIP topology

FCoE (Fibre Channel over Ethernet)

- Data centers typically have multiple networks to handle various types of I/O traffic — for

example, an Ethernet network for TCP/IP communication and an FC network for FC communication.

- TCP/IP is typically used for client-server communication, data backup, infrastructure management communication, and so on.
- FC is typically used for moving block-level data between storage and servers.
- To support multiple networks, servers in a data center are equipped with multiple redundant physical network interfaces — for example, multiple Ethernet and FC cards/adapters. In addition, to enable the communication, different types of networking switches and physical cabling infrastructure are implemented in data centers.

- The need for two different kinds of physical network infrastructure increases the overall cost and complexity of data center operation.
- Fibre Channel over Ethernet (FCoE) protocol provides consolidation of LAN and SAN traffic over a single physical interface infrastructure.
- FCoE helps organizations address the challenges of having multiple discrete network infrastructures.
- FCoE uses the Converged Enhanced Ethernet (CEE) link (10 Gigabit Ethernet) to send FC frames over Ethernet.

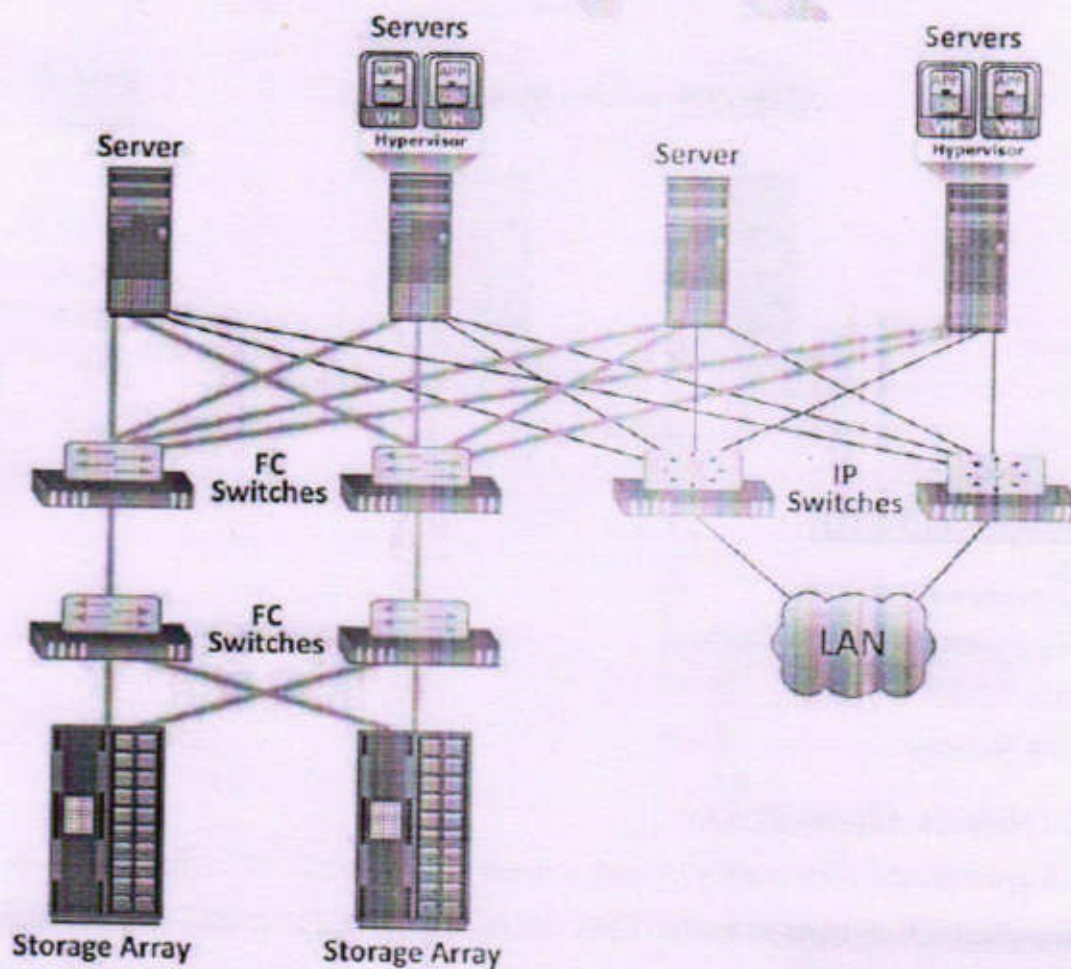


Fig Before using FCOE

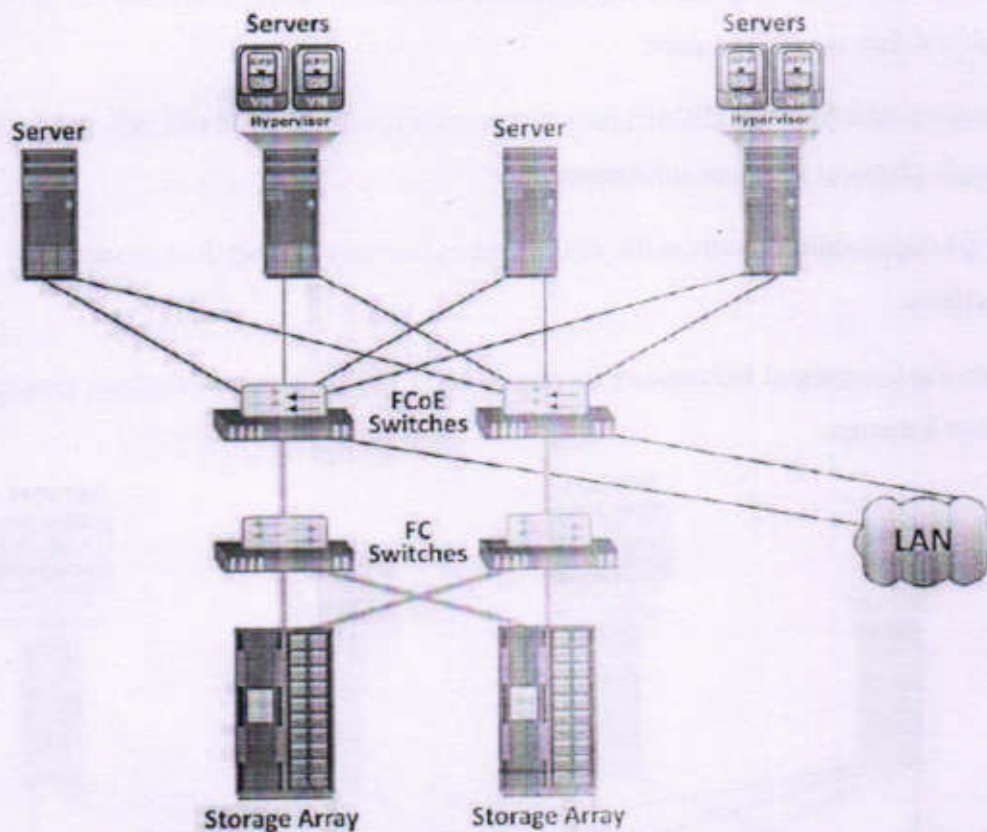


Fig After using FCOE

Components of FCOE

The key components of FCOE are :

- Converged Network Adaptors(CNA)
- Cables
- FCOE Switches

Converged Network Adaptors(CNA)

- A CNA provides the functionality of both a standard NIC and an FC HBA in a single adapter and consolidates both types of traffic. CNA eliminates the need to deploy separate adapters and cables for FC and Ethernet communications, thereby reducing the required number of server slots and switch ports.

- As shown in Fig below, a CNA contains separate modules for 10 Gigabit Ethernet, Fibre Channel, and FCoE Application Specific Integrated Circuits (ASICs). The FCoE ASIC encapsulates FC frames into Ethernet frames. One end of this ASIC is connected to 10GbE and FC ASICs for server connectivity, while the other end provides a 10GbE interface to connect to an FCoE switch.

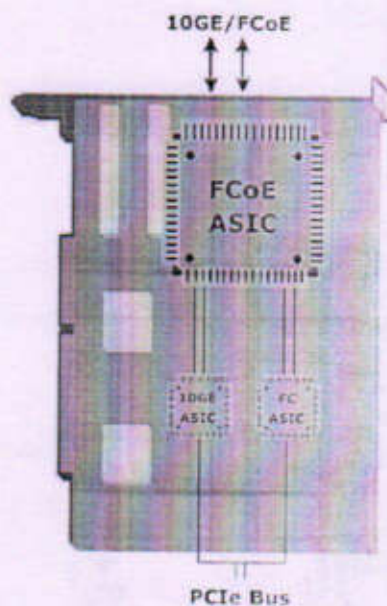


Fig : Converged Network Adapter

Cables

- There are two options available for FCoE cabling:
 1. Copper based Twinax
 2. standard fiber optical cables.
- A Twinax cable is composed of two pairs of copper cables covered with a shielded casing. The Twinax cable can transmit data at the speed of 10 Gbps over shorter distances up to 10 meters. Twinax cables require less power and are less expensive than fiber optic cables.
- The Small Form Factor Pluggable Plus (SFP+) connector is the primary connector used for FCoE links and can be used with both optical and copper cables.

FCoE Switches

- An FCoE switch has both **Ethernet switch** and **Fibre Channel switch** functionalities.
- As shown in Fig below, FCoE switch consists of:
 1. *Fibre Channel Forwarder (FCF)*,
 2. *Ethernet Bridge*,
 3. set of Ethernet ports
 4. optional FC ports
- The function of the FCF is to encapsulate the FC frames, received from the FC port, into the FCoE frames and also to de-encapsulate the FCoE frames, received from the Ethernet Bridge, to the FC frames.
- Upon receiving the incoming traffic, the FCoE switch inspects the **Ethertype** (used to indicate which protocol is encapsulated in the payload of an Ethernet frame) of the incoming frames and uses that to determine the destination.
 - If the Ethertype of the frame is FCoE, the switch recognizes that the frame contains an FC payload and forwards it to the FCF. From there, the FC is extracted from the FCoE frame and transmitted to FC SAN over the FC ports.
 - If the Ethertype is not FCoE, the switch handles the traffic as usual Ethernet traffic and forwards it over the Ethernet ports.

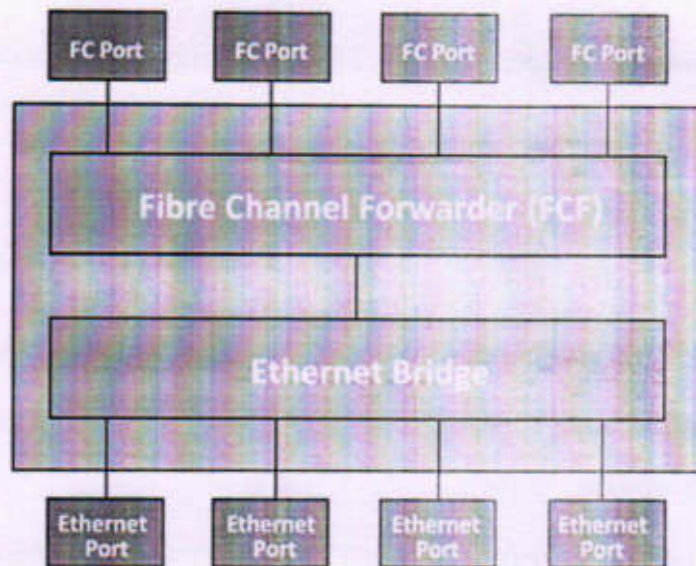


Fig FCoE switch generic architecture

NETWORK ATTACHED STORAGE (NAS)

File Sharing Environment

- File sharing enables users to share files with other users
- In file-sharing environment, the creator or owner of a file determines the type of access to be given to other users and controls changes to the file.
- When multiple access a shared file at the same time, a locking scheme is required to maintain data integrity and also make this sharing possible. This is taken care by file-sharing environment.
- Examples of file sharing methods:
 - File Transfer Protocol (FTP)
 - Distributed File System (DFS)
 - Network File System (NFS) and Common Internet File System (CIFS)
 - Peer-to-Peer (P2P)

What is NAS?

- NAS is an IP based dedicated, high-performance file sharing and storage device.
- Enables NAS clients to share files over an IP network.
- Uses network and file-sharing protocols to provide access to the file data.
- Ex: Common Internet File System (CIFS) and Network File System (NFS).
- Enables both UNIX and Microsoft Windows users to share the same data seamlessly.
- NAS device uses its own operating system and integrated hardware and software components to meet specific file-service needs.
- Its operating system is optimized for file I/O which performs better than a general-purpose server.
- A NAS device can serve more clients than general-purpose servers and provide the benefit of server consolidation.

Components of NAS

- NAS device has *two* key components (as shown in Fig 2.33): **NAS head** and **storage**.
- In some NAS implementations, the storage could be external to the NAS device and shared with other hosts.
- NAS head includes the following components:
 - CPU and memory
 - One or more network interface cards (NICs), which provide connectivity to the client network.
 - An optimized operating system for managing the NAS functionality. It translates file-level requests into block-storage requests and further converts the data supplied at the block level to file data
 - NFS, CIFS, and other protocols for file sharing

- Industry-standard storage protocols and ports to connect and manage physical disk resources
- The NAS environment includes clients accessing a NAS device over an IP network using file-sharing protocols.

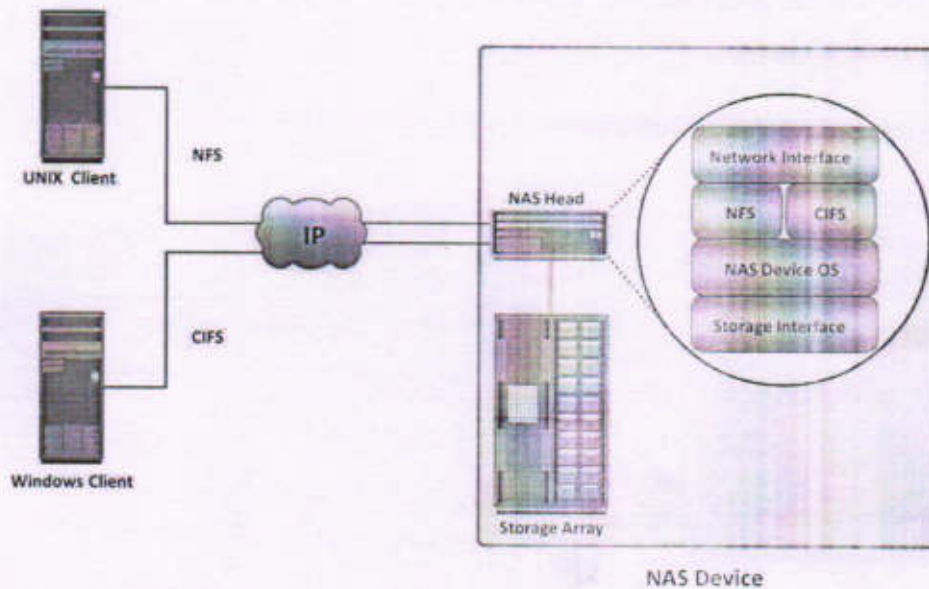


Fig 2.33 Components of NAS

NAS I/O Operation

- NAS provides *file-level data access* to its clients. File I/O is a high-level request that specifies the file to be accessed.
- Eg: a client may request a file by specifying its name, location, or other attributes. The NAS operating system keeps track of the location of files on the disk volume and converts client file I/O into block-level I/O to retrieve data.
- The process of handling I/Os in a NAS environment is as follows:
1. The requestor (client) packages an I/O request into TCP/IP and forwards it through the network stack. The NAS device receives this request from the network.

2. The NAS device converts the I/O request into an appropriate physical storage request, which is a block-level I/O, and then performs the operation on the physical storage.
3. When the NAS device receives data from the storage, it processes and repackages the data into an appropriate file protocol response.
4. The NAS device packages this response into TCP/IP again and forwards it to the client through the network.

➤ Fig 2.34 illustrates the NAS I/O operation

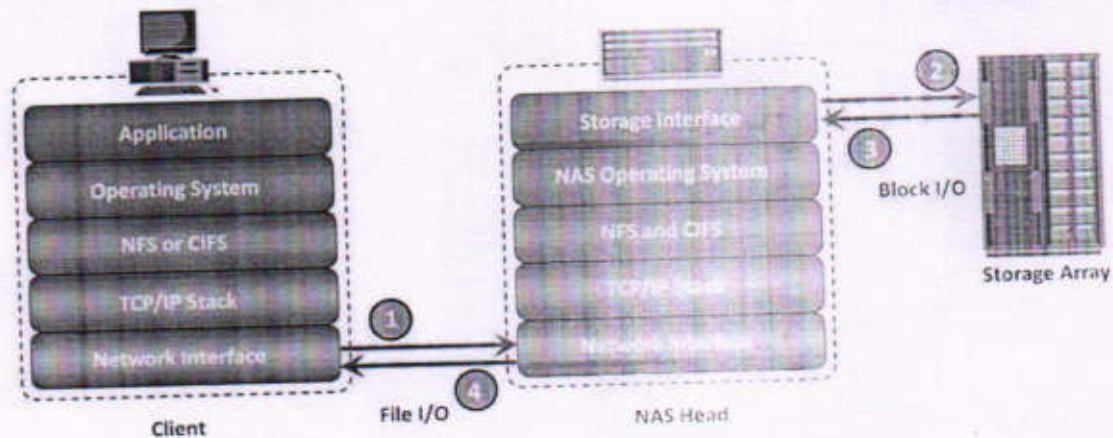


Fig 2.34 NAS I/O Operation

NAS File Sharing Protocols

- NAS devices support multiple file-service protocols to handle file I/O requests
- Two common NAS file sharing protocols are:
 - Common Internet File System (CIFS)
 - Network File System (NFS)
- NAS devices enable users to share file data across different operating environments
- It provides a means for users to migrate transparently from one operating system to another

Network File System (NFS)

- NFS is a **client-server protocol** for file sharing that is commonly used on **UNIX systems**.
- NFS was originally based on the connectionless *User Datagram Protocol (UDP)*.
- It uses *Remote Procedure Call (RPC)* as a method of inter-process communication between two computers.
- The NFS protocol provides a set of RPCs to access a remote file system for the following operations:
 - Searching files and directories
 - Opening, reading, writing to, and closing a file
 - Changing file attributes
 - Modifying file links and directories
- NFS creates a connection between the client and the remote system to transfer data.
- NFSv3 and earlier is a *stateless protocol*
- It does not maintain any kind of table to store information about open files and associated pointers. Each call provides a full set of arguments - a file handle, a particular position to read or write, and the versions of NFS - to access files on the server .
- Currently, three versions of NFS are in use:
 1. **NFS version 2 (NFSv2):** Uses *UDP* to provide a *stateless* network connection between a client and a server. Features, such as locking, are handled outside the protocol.
 2. **NFS version 3 (NFSv3):** Uses *UDP or TCP*, and is based on the *stateless protocol* design. It includes some new features, such as a 64-bit file size, asynchronous writes, and additional file attributes to reduce refetching.
 3. **NFS version 4 (NFSv4):** Uses *TCP* and is based on a *stateful protocol* design. It offers enhanced security. The latest NFS version 4.1 is the enhancement of NFSv4 and includes some new features, such as session model, parallel NFS (pNFS), and data retention.

Common Internet File System (CIFS)

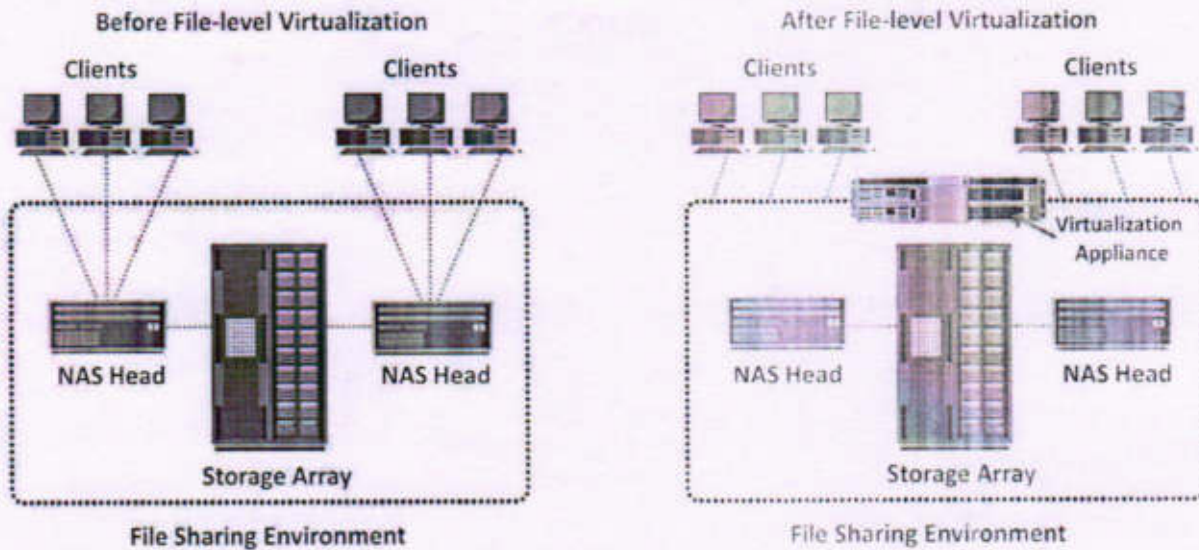
- CIFS is a *client-server application* protocol
- It enables clients to access files and services on remote computers over TCP/IP.
- It is a public, or open, variation of **Server Message Block (SMB)** protocol.
- It provides following features to ensure data integrity:
 - It uses file and record locking to prevent users from overwriting the work of another user on a file or a record.
 - It supports fault tolerance and can automatically restore connections and reopen files that were open prior to an interruption. This feature depends on whether an application is written to take advantage of this.
 - CIFS is a stateful protocol because the CIFS server maintains connection information regarding every connected client. If a network failure or CIFS server failure occurs, the client receives a disconnection notification. User disruption is minimized if the application has the embedded intelligence to restore the connection. However, if the embedded intelligence is missing, the user must take steps to reestablish the CIFS connection.
- Users refer to remote file systems with an easy-to-use file-naming scheme:
- Eg: \\server\share or \\servername.domain.suffix\share

File-level Virtualization

- File-level virtualization, implemented in NAS or the file server environment, provides a simple, non disruptive file-mobility solution.
- It eliminates the dependencies between data accessed at the file level and the location where the files are physically stored.
- It creates a logical pool of storage, enabling users to use a logical path, rather than a physical path, to access files.

- A global namespace is used to map the logical path of a file to the physical path names. File-level virtualization enables the movement of files across NAS devices, even if the files are being accessed.

Before and After File-level Virtualization



- Dependency between client access and file location
- Underutilized storage resources
- Downtime is caused by data migrations
- Break dependencies between client access and file location
- Storage utilization is optimized
- Non-disruptive migrations



MODULE – 4

INTRODUCTION TO BUSINESS CONTINUITY, BACKUP AND ARCHIVE

❖ INTRODUCTION TO BUSINESS CONTINUITY

Business Continuity (BC):

Business continuity (BC) is an integrated and enterprise wide process that includes all activities (internal and external to IT) that a business must perform to mitigate the impact of planned and unplanned downtime.

BC entails preparing for, responding to, and recovering from a system outage that adversely affects business operations. It involves proactive measures, such as business impact analysis, risk assessments, deployment of BC technology solutions (backup and replication), and reactive measures, such as disaster recovery and restart, to be invoked in the event of a failure.

The goal of a BC solution is to ensure the “**information availability**” required to conduct vital business operations.

Information Availability:

Information availability (IA) refers to the ability of the infrastructure to function according to business expectations during its specified time of operation. Information availability ensures that people (employees, customers, suppliers, and partners) can access information whenever they need it. Information availability can be defined in terms of:

1. Reliability,
2. Accessibility
3. Timeliness.

1. **Reliability:** This reflects a component’s ability to function without failure, under stated conditions, for a specified amount of time.
2. **Accessibility:** This is the state within which the required information is accessible at the right place, to the right user. The period of time during which the system is in an accessible state is termed **system uptime**; when it is not accessible it is termed **system**

downtime.

- 3. Timeliness:** Defines the exact moment or the time window (a particular time of the day, week, month, and/or year as specified) during which information must be accessible. For example, if online access to an application is required between 8:00 am and 10:00 pm each day, any disruptions to data availability outside of this time slot are not considered to affect timeliness.

Causes of Information Unavailability

Various planned and unplanned incidents result in data unavailability.

- **Planned outages** include installation/integration/maintenance of new hardware, software upgrades or patches, taking backups, application and data restores, facility operations (renovation and construction), and refresh/migration of the testing to the production environment.
- **Unplanned outages** include failure caused by database corruption, component failure, and human errors.
- **Disasters (natural or man-made)** such as flood, fire, earthquake, and contamination are another type of incident that may cause data unavailability.

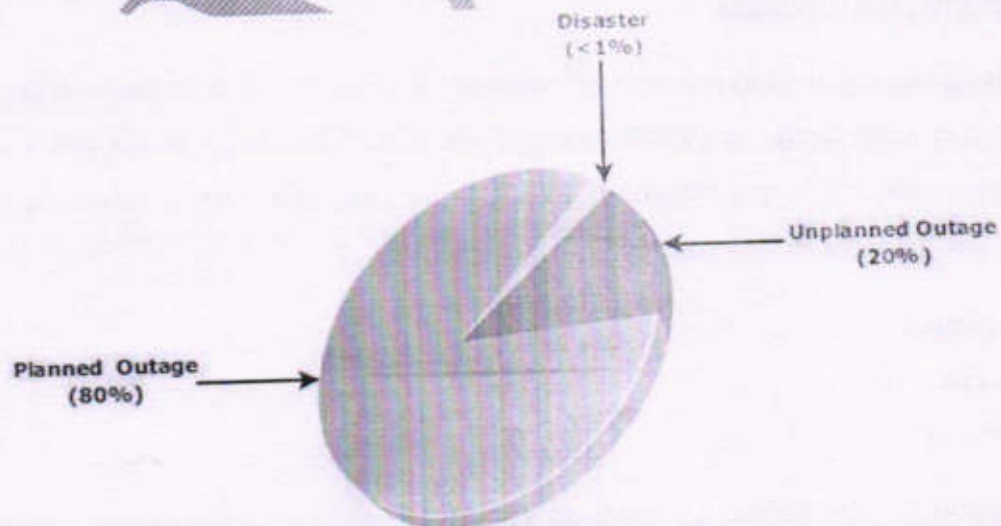


Fig 3.1: Disruptors of Information Availability

As illustrated in Fig 3.1 above, the majority of outages are planned. Planned outages are expected and scheduled, but still cause data to be unavailable.

Consequences of Downtime

- Information unavailability or downtime results in loss of productivity, loss of revenue, poor financial performance, and damage to reputation.
- Loss of productivity includes reduced output per unit of labor, equipment, and capital.
- Loss of revenue includes direct loss, compensatory payments, future revenue loss, billing loss, and investment loss.
- Poor financial performance affects revenue recognition, cash flow, discounts, payment guarantees, credit rating, and stock price.
- Damages to reputations may result in a loss of confidence or credibility with customers, suppliers, financial markets, banks, and business partners.
- An important metric, *average cost of downtime per hour*, provides a key estimate in determining the appropriate BC solutions. It is calculated as follows:

$$\text{Average cost of downtime per hour} = \text{average productivity loss per hour} + \text{average revenue loss per hour}$$

Where:

$$\text{Productivity loss per hour} = \frac{\text{(total salaries and benefits of all employees per week)}}{\text{(average number of working hours per week)}}$$

$$\text{Average revenue loss per hour} = \frac{\text{(total revenue of an organization per week)}}{\text{(average number of hours per week that an organization is open for business)}}$$

Measuring Information Availability

- Information availability (IA) relies on the availability of physical and virtual components of a data center. Failure of these components might disrupt IA. A failure is the termination of a component's capability to perform a required function. The component's capability can be restored by performing an external corrective action, such as a manual reboot, a repair, or replacement of the failed component(s).
- Proactive risk analysis performed as part of the BC planning process considers the component failure rate and average repair time, which are measured by MTBF and MTTR:

- **Mean Time Between Failure (MTBF):** It is the average time available for a system or component to perform its normal operations between failures.
- **Mean Time To Repair (MTTR):** It is the average time required to repair a failed component. MTTR includes the total time required to do the following activities: Detect the fault, mobilize the maintenance team, diagnose the fault, obtain the spare parts, repair, test, and restore the data.

Fig 3.2 illustrates the various information availability metrics that represent system uptime and downtime.

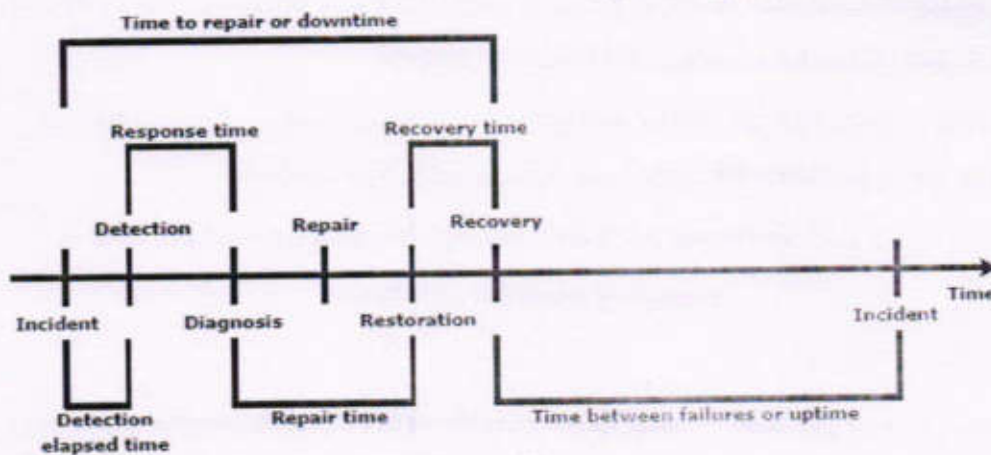


Fig 3-2: Information availability metrics

IA is the time period that a system is in a condition to perform its intended function upon demand. It can be expressed in terms of system uptime and downtime and measured as the amount or percentage of system uptime:

$$IA = \text{system uptime} / (\text{system uptime} + \text{system downtime})$$

In terms of MTBF and MTTR, IA could also be expressed as

$$IA = \text{MTBF} / (\text{MTBF} + \text{MTTR})$$

Uptime per year is based on the exact timeliness requirements of the service, this calculation leads to the number of "9s" representation for availability metrics.

Table 3-1 lists the approximate amount of downtime allowed for a service to achieve certain levels of 9s availability. For example, a service that is said to be "five 9s available" is available for 99.999 percent of the scheduled time in a year (24×365).

UPTIME (%)	DOWNTIME (%)	DOWNTIME PER YEAR	DOWNTIME PER WEEK
98	2	7.3 days	3 hr, 22 minutes
99	1	3.65 days	1 hr, 41 minutes
99.8	0.2	17 hr, 31 minutes	20 minutes, 10 secs
99.9	0.1	8 hr, 45 minutes	10 minutes, 5 secs
99.99	0.01	52.5 minutes	1 minute
99.999	0.001	5.25 minutes	6 secs
99.9999	0.0001	31.5 secs	0.6 secs

Table 3-1: Availability percentage and Allowable downtime

BC Terminology

This section defines common terms related to BC operations which are used in this module to explain advanced concepts:

- **Disaster recovery:** This is the coordinated process of restoring systems, data, and the infrastructure required to support key ongoing business operations in the event of a disaster. It is the process of restoring a previous copy of the data and applying logs or other necessary processes to that copy to bring it to a known point of consistency. Once all recoveries are completed, the data is validated to ensure that it is correct.
- **Disaster restart:** This is the process of restarting business operations with mirrored consistent copies of data and applications.
- **Recovery-Point Objective (RPO):** This is the point in time to which systems and data must be recovered after an outage. It defines the amount of data loss that a business can endure. A large RPO signifies high tolerance to information loss in a business. Based on the RPO, organizations plan for the minimum frequency with which a backup or replica must be made. For example, if the RPO is six hours, backups or replicas must be made at least once in 6 hours. Fig 3.3 (a) shows various RPOs and their corresponding ideal recovery strategies. An organization can plan for an appropriate BC technology solution on the basis of the RPO it sets. For example:
 - **RPO of 24 hours:** This ensures that backups are created on an offsite tape drive every midnight. The corresponding recovery strategy is to restore data from the set of last

backup tapes.

- **RPO of 1 hour:** Shipping database logs to the remote site every hour. The corresponding recovery strategy is to recover the database at the point of the last log shipment.
- **RPO in the order of minutes:** Mirroring data asynchronously to a remote site
- **Near zero RPO:** This mirrors mission-critical data synchronously to a remote site.

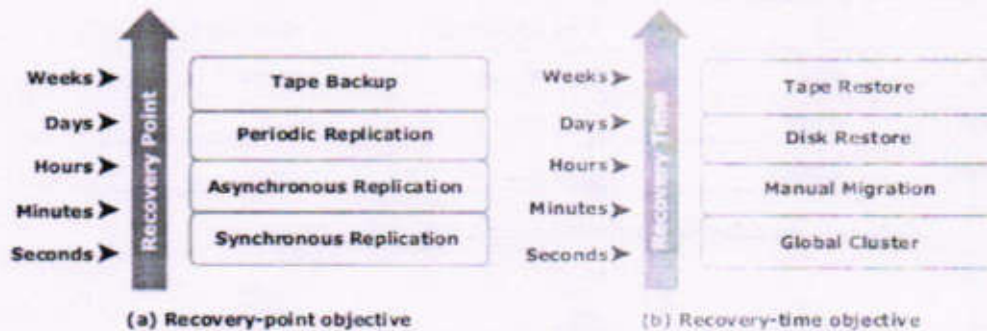


Fig 3.3: Strategies to meet RPO and RTO targets

- **Recovery-Time Objective (RTO):** The time within which systems and applications must be recovered after an outage. It defines the amount of downtime that a business can endure and survive. Businesses can optimize disaster recovery plans after defining the RTO for a given system. For example, if the RTO is two hours, then use a disk backup because it enables a faster restore than a tape backup. However, for an RTO of one week, tape backup will likely meet requirements. Some examples of RTOs and the recovery strategies to ensure data availability are listed below (refer to Fig 3.3 (b)):

- **RTO of 72 hours:** Restore from backup tapes at a cold site.
- **RTO of 12 hours:** Restore from tapes at a hot site.
- **RTO of few hours:** Use a data vault to a hot site.
- **RTO of a few seconds:** Cluster production servers with bidirectional mirroring, enabling the applications to run at both sites simultaneously.

BC Planning Life Cycle

BC planning must follow a disciplined approach like any other planning process. Organizations today dedicate specialized resources to develop and maintain BC plans. From the conceptualization to the realization of the BC plan, a life cycle of activities can be defined for the BC process.

The BC planning lifecycle includes five stages shown below (Fig 3.4):

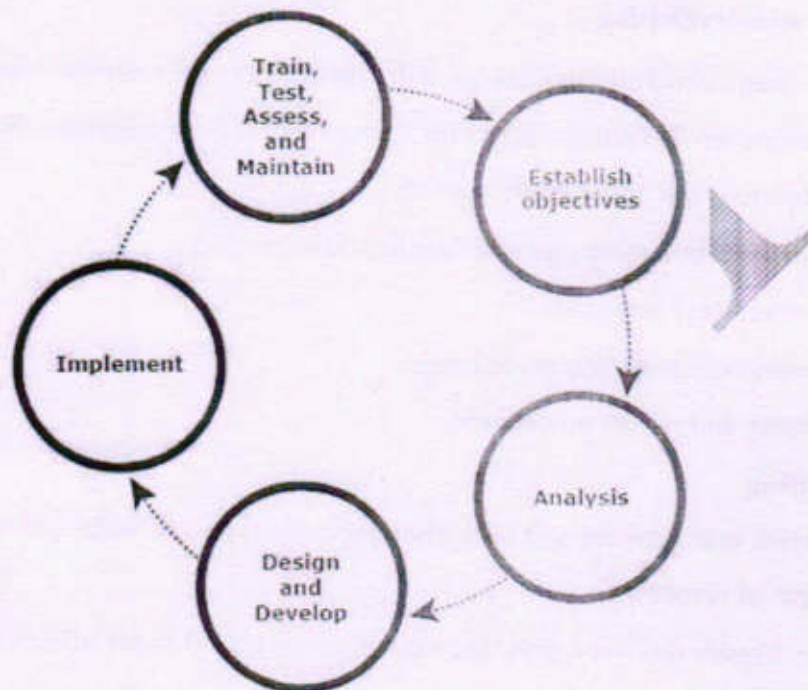


Fig 3.4: BC Planning Lifecycle

Several activities are performed at each stage of the BC planning lifecycle, including the following key activities:

1. Establishing objectives

- Determine BC requirements.
- Estimate the scope and budget to achieve requirements.
- Select a BC team by considering subject matter experts from all areas of the business, whether internal or external.
- Create BC policies.

2. Analyzing

- Collect information on data profiles, business processes, infrastructure support, dependencies, and frequency of using business infrastructure.
- Identify critical business needs and assign recovery priorities.
- Create a risk analysis for critical areas and mitigation strategies.
- Conduct a Business Impact Analysis (BIA).
- Create a cost and benefit analysis based on the consequences of data unavailability.

3. Designing and developing

- Define the team structure and assign individual roles and responsibilities. For example, different teams are formed for activities such as emergency response, damage assessment, and infrastructure and application recovery.
- Design data protection strategies and develop infrastructure.
- Develop contingency scenarios.
- Develop emergency response procedures.
- Detail recovery and restart procedures.

4. Implementing

- Implement risk management and mitigation procedures that include backup, replication, and management of resources.
- Prepare the disaster recovery sites that can be utilized if a disaster affects the primary data center.
- Implement redundancy for every resource in a data center to avoid single points of failure.

5. Training, testing, assessing, and maintaining

- Train the employees who are responsible for backup and replication of business-critical data on a regular basis or whenever there is a modification in the BC plan
- Train employees on emergency response procedures when disasters are declared.
- Train the recovery team on recovery procedures based on contingency scenarios.
- Perform damage assessment processes and review recovery plans.
- Test the BC plan regularly to evaluate its performance and identify its limitations.
- Assess the performance reports and identify limitations.

→ Update the BC plans and recovery/restart procedures to reflect regular changes within the data center.

Failure Analysis

Single Point of Failure

- A **single point of failure** refers to the failure of a component that can terminate the availability of the entire system or IT service.
- Fig 3.5 depicts a system setup in which an application, running on a VM, provides an interface to the client and performs I/O operations.
- The client is connected to the server through an IP network, the server is connected to the storage array through a FC connection, an HBA installed at the server sends or receives data to and from a storage array, and an FC switch connects the HBA to the storage port
- In a setup where **each component must function as required to ensure data availability**, the failure of a single physical or virtual component causes the failure of the entire data center or an application, resulting in disruption of business operations.
- In this example, failure of a hypervisor can affect all the running VMs and the virtual network, which are hosted on it.
- There can be several similar single points of failure identified in this example. A VM, a hypervisor, an HBA/NIC on the server, the physical server, the IP network, the FC switch, the storage array ports, or even the storage array could be a potential single point of failure. To avoid single points of failure, it is essential to implement a fault-tolerant mechanism.

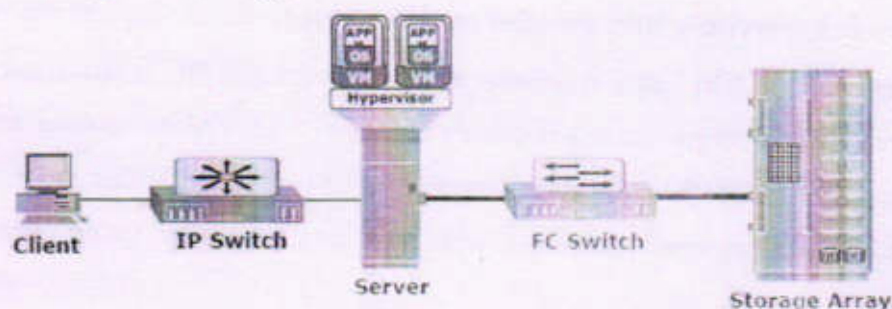


Fig 3.5: Single Point of Failure

Resolving Single Points of Failure

- To mitigate a single point of failure, systems are designed with redundancy, such that the system will fail only if all the components in the redundancy group fail. This ensures that the failure of a single component does not affect data availability.
- Data centers follow stringent guidelines to implement fault tolerance for uninterrupted information availability. Careful analysis is performed to eliminate every single point of failure.
- The example shown in Fig 3.6 represents all enhancements of the system shown in Fig 3.5 in the infrastructure to mitigate single points of failure:
 - Configuration of redundant HBAs at a server to mitigate single HBA failure
 - Configuration of NIC (network interface card) teaming at a server allows protection against single physical NIC failure. It allows grouping of two or more physical NICs and treating them as a single logical device. NIC teaming eliminates the single point of failure associated with a single physical NIC.
 - Configuration of redundant switches to account for a switch failure
 - Configuration of multiple storage array ports to mitigate a port failure
 - RAID and hot spare configuration to ensure continuous operation in the event of disk failure
 - Implementation of a redundant storage array at a remote site to mitigate local site failure
 - Implementing server (or compute) clustering, a fault-tolerance mechanism whereby two or more servers in a cluster access the same set of data volumes. Clustered servers exchange a heartbeat to inform each other about their health. If one of the servers or hypervisors fails, the other server or hypervisor can take up the workload.
 - Implementing a VM Fault Tolerance mechanism ensures BC in the event of a server failure. This technique creates duplicate copies of each VM on another server so that when a VM failure is detected, the duplicate VM can be used for failover. The two VMs are kept in synchronization with each other in order to perform successful failover.

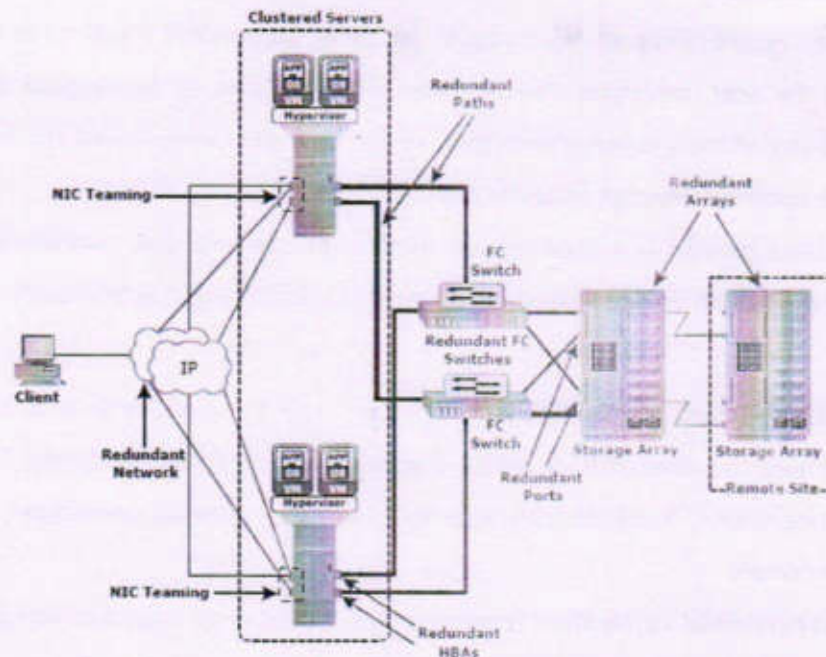


Fig 3.6: Resolving single points of failure

Multipathing Software

- Configuration of multiple paths increases the data availability through path failover. If servers are configured with one I/O path to the data there will be no access to the data if that path fails. Redundant paths eliminate the path to become single points of failure.
- Multiple paths to data also improve I/O performance through load sharing and maximize server, storage, and data path utilization.
- In practice, merely configuring multiple paths does not serve the purpose. Even with multiple paths, if one path fails, I/O will not reroute unless the system recognizes that it has an alternate path.
- Multipathing software provides the functionality to recognize and utilize alternate I/O path to data. Multipathing software also manages the load balancing by distributing I/Os to all available, active paths.
- In a virtual environment, multipathing is enabled either by using the hypervisor's built-in capability or by running a third-party software module, added to the hypervisor.

BC Technology Solutions

After analyzing the business impact of an outage, designing appropriate solutions to recover from a failure is the next important activity. One or more copies of the original data are maintained using any of the following strategies, so that data can be recovered and business operations can be restarted using an alternate copy:

1. **Backup:** Data backup is a predominant method of ensuring data availability. The frequency of backup is determined based on RPO, RTO, and the frequency of data changes.
2. **Storage array-based replication (local):** Data can be replicated to a separate location within the same storage array. The replica is used independently for other business operations. Replicas can also be used for restoring operations if data corruption occurs.
3. **Storage array-based replication (remote):** Data in a storage array can be replicated to another storage array located at a remote site. If the storage array is lost due to a disaster, business operations can be started from the remote storage array.

BACKUP AND ARCHIVE

- **Data Backup** is a copy of production data, created and retained for the sole purpose of recovering lost or corrupted data.
- Evaluating the various backup methods along with their recovery considerations and retention requirements is an essential step to implement a successful backup and recovery solution.
- Organizations generate and maintain large volumes of data, and most of the data is fixed content. This fixed content is rarely accessed after a period of time. Still, this data needs to be retained for several years to meet regulatory compliance.
- **Data archiving** is the process of moving data that is no longer actively used, from primary storage to a low-cost secondary storage. This data is retained in the secondary storage for a long term to meet regulatory requirements. This reduces the amount of data to be backed up and the time required to back up the data.

Backup Purpose

Backups are performed to serve three purposes: *disaster recovery*, *operational recovery*, and *archival*. These are discussed in the following sections.

Disaster Recovery

- Backups are performed to address disaster recovery needs.
- The backup copies are used for restoring data at an alternate site when the primary site is incapacitated due to a disaster. Based on RPO and RTO requirements, organizations use different backup strategies for disaster recovery.
- When a tape-based backup method is used as a disaster recovery strategy, the backup tape media is shipped and stored at an offsite location. These tapes can be recalled for restoration at the disaster recovery site.
- Organizations with stringent RPO and RTO requirements use remote replication technology to replicate data to a disaster recovery site. Organizations can bring production systems online in a relatively short period of time if a disaster occurs.

Operational Recovery

- Data in the production environment changes with every business transaction and operation.
- Operational recovery is the use of backups to restore data if data loss or logical

corruption occurs during routine processing.

- For example, it is common for a user to accidentally delete an important email or for a file to become corrupted, which can be restored from operational backup.

Archival

- Backups are also performed to address archival requirements.
- Traditional backups are still used by small and medium enterprises for long-term preservation of transaction records, e- mail messages, and other business records required for regulatory compliance.

Apart from addressing disaster recovery, archival, and operational requirements, backups serve as a protection against data loss due to physical damage of a storage device, software failures, or virus attacks. Backups can also be used to protect against accidents such as a deletion or intentional data destruction.

Backup Granularity

- Backup granularity depends on business needs and the required RTO/RPO.
- Based on the granularity, backups can be categorized as full, incremental and cumulative (differential).
- Most organizations use a combination of these three backup types to meet their backup and recovery requirements.
- The below figure shows the different backup granularity levels.

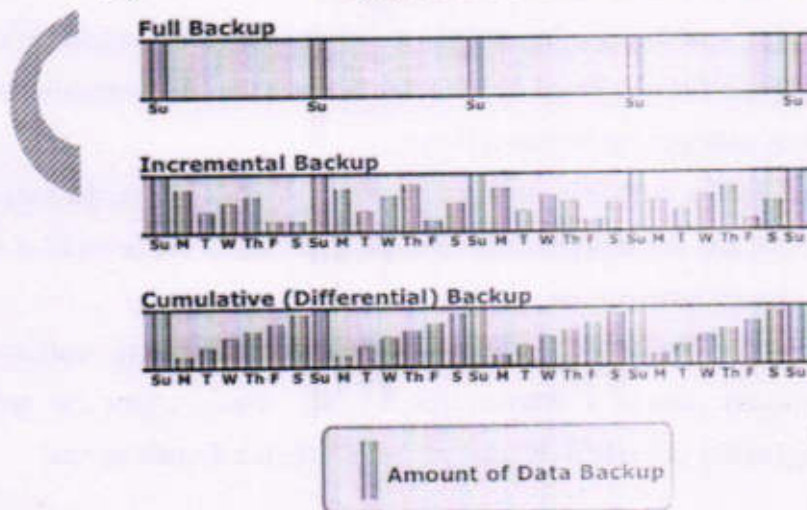


Figure: Different Granularity levels

Full Backup

- Full backup is a backup of the complete data on the production volumes.
- A full backup copy is created by copying the data in the production volumes to a backup

storage device.

- It provides a faster recovery but requires more storage space and also takes more time to back up.

Incremental Backup

- Incremental backup copies the data that has changed since the last full or incremental backup, whichever has occurred more recently.
- This is much faster than a full backup (because the volume of data backed up is restricted to the changed data only) but takes longer to restore.

Cumulative Backup

- Cumulative backup copies the data that has changed since the last full backup.
 - This method takes longer than an incremental backup but is faster to restore.
- Restore operations vary with the granularity of the backup.
- A full backup provides a single repository from which the data can be easily restored.
- The process of restoration from an incremental backup requires the last full backup and all the incremental backups available until the point of restoration.
- A restore from a cumulative backup requires the last full backup and the most recent cumulative backup.
- The below figure shows an example of restoring data from incremental backup.

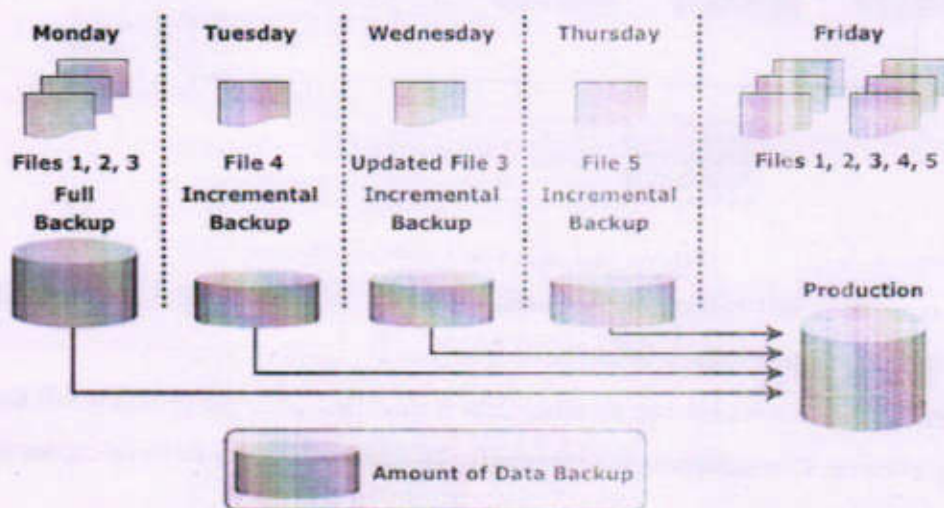


Figure: Restoring from Incremental Backup

- In this example, a full backup is performed on Monday evening. Each day after that, an incremental backup is performed.

- On Tuesday, a new file (File 4 in the figure) is added, and no other files have changed.
- Consequently, only File 4 is copied during the incremental backup performed on Tuesday evening.
- On Wednesday, no new files are added, but File 3 has been modified. Therefore, only the modified File 3 is copied during the incremental backup on Wednesday evening.
- Similarly, the incremental backup on Thursday copies only File 5.
- On Friday morning, there is data corruption, which requires data restoration from the backup.
- The first step toward data restoration is restoring all data from the full backup of Monday evening. The next step is applying the incremental backups of Tuesday, Wednesday, and Thursday.
- In this manner, data can be successfully recovered to its previous state, as it existed on Thursday evening.
- The below figure shows an example of restoring data from cumulative backup:

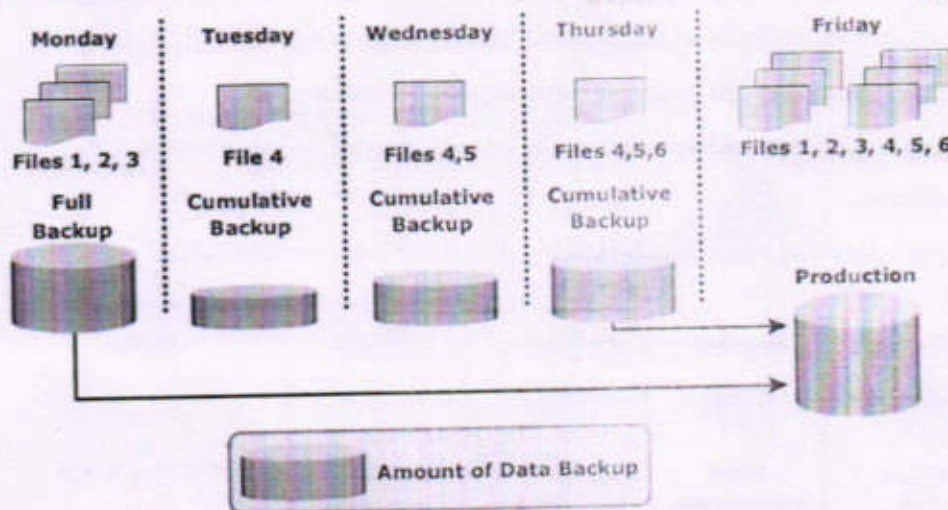


Figure: Restoring from Cumulative Backup

- In this example, a full backup of the business data is taken on Monday evening. Each day after that, a cumulative backup is taken.
- On Tuesday, File 4 is added and no other data is modified since the previous full backup of Monday evening. Consequently, the cumulative backup on Tuesday evening copies only File 4.
- On Wednesday, File 5 is added. The cumulative backup taking place on Wednesday evening copies both File 4 and File 5 because these files have been added or modified since the last

full backup.

- Similarly, on Thursday, File 6 is added. Therefore, the cumulative backup on Thursday evening copies all three files: File 4, File 5, and File 6.
- On Friday morning, data corruption occurs that requires data restoration using backup copies.
- The first step in restoring data is to restore all the data from the full backup of Monday evening. The next step is to apply only the latest cumulative backup, which is taken on Thursday evening.
- In this way, the production data can be recovered faster because it needs only two copies of data — the last full backup and the latest cumulative backup.

Backup Methods

- **Hot backup and cold backup** are the two methods deployed for backup. They are based on the state of the application when the backup is performed.
- In a **hot backup**, the application is up and running, with users accessing their data during the backup process. This method of backup is also referred to as an *online backup*.
- In a **cold backup**, the application is not active or shutdown during the backup process and is also called as *offline backup*.
- The hot backup of online production data becomes more challenging because data is actively used and changed.
- An open file is locked by the operating system and is not backed up during the backup process. In such situations, an *open file agent* is required to back up the open file.
- In database environments, the use of open file agents is not enough, because the agent should also support a consistent backup of all the database components.
- For example, a database is composed of many files of varying sizes occupying several file systems. To ensure a consistent database backup, all files need to be backed up in the same state. That does not necessarily mean that all files need to be backed up at the same time, but they all must be synchronized so that the database can be restored with consistency.
- The disadvantage associated with a hot backup is that the agents usually affect the overall application performance.

Storage Area Networks(18CS822)

- Consistent backups of databases can also be done by using a cold backup. This requires the database to remain inactive during the backup. Of course, the disadvantage of a cold backup is that the database is inaccessible to users during the backup process.
- Hot backup is used in situations where it is not possible to shut down the database. This is facilitated by database backup agents that can perform a backup while the database is active. The disadvantage associated with a hot backup is that the agents usually affect overall application performance.
- A **point-in-time (PIT)** copy method is deployed in environments where the impact of downtime from a cold backup or the performance resulting from a hot backup is unacceptable. The PIT copy is created from the production volume and used as the source for the backup. This reduces the impact on the production volume.
- Certain attributes and properties attached to a file, such as permissions, owner, and other metadata, also need to be backed up. These attributes are as important as the data itself and must be backed up for consistency.
- Backup of boot sector and partition layout information is also critical for successful recovery.
- In a disaster recovery environment, **bare-metal recovery (BMR)** refers to a backup in which all metadata, system information, and application configurations are appropriately backed up for a full system recovery. BMR builds the base system, which includes partitioning, the file system layout, the operating system, the applications, and all the relevant configurations. BMR recovers the base system first, before starting the recovery of data files. Some BMR technologies can recover a server onto dissimilar hardware.

Backup Architecture

- A backup system commonly uses the client-server architecture with a backup server and multiple backup clients.
- The below figure illustrates the backup architecture.
- The backup server manages the backup operations and maintains the backup catalog, which contains information about the backup configuration and backup metadata.
- Backup configuration contains information about when to run backups, which client data to be backed up, and so on, and the backup metadata contains information about the backed up data.
- The role of a backup client is to gather the data that is to be backed up and send it to the storage node. It also sends the tracking information to the backup server.

- The storage node is responsible for writing the data to the backup device. (In a backup environment, a storage node is a host that controls backup devices.)
- The storage node also sends tracking information to the backup server. In many cases, the storage node is integrated with the backup server, and both are hosted on the same physical platform.
- A backup device is attached directly or through a network to the storage node's host platform. Some backup architecture refers to the storage node as the media server because it manages the storage device.
- Backup software provides reporting capabilities based on the backup catalog and the log files. These reports include information, such as the amount of data backed up, the number of completed and incomplete backups, and the types of errors that might have occurred.
- Reports can be customized depending on the specific backup software used.

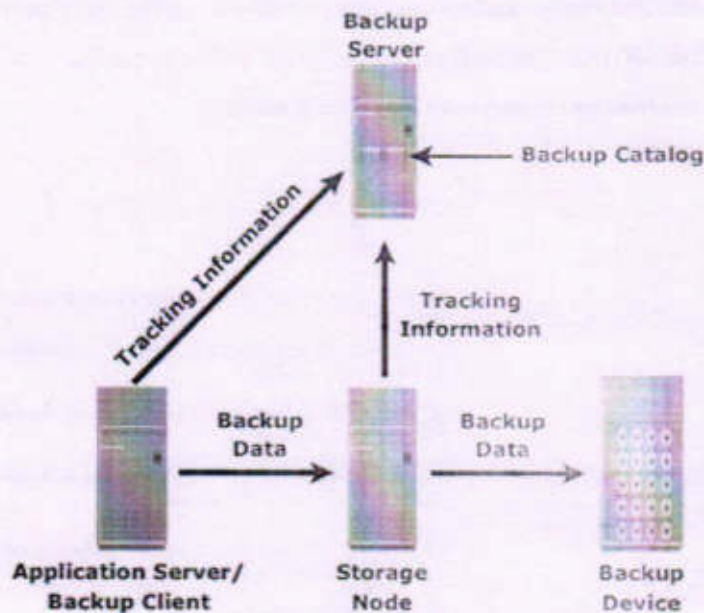


Figure: Backup Architecture

Backup and Restore Operations

- When a backup operation is initiated, significant network communication takes place between the different components of a backup infrastructure.
- The backup operation is typically initiated by a server, but it can also be initiated by a client.
- The backup server initiates the backup process for different clients based on the backup schedule

configured for them.

- For example, the backup for a group of clients may be scheduled to start at 11:00 p.m. every day.
- The backup server coordinates the backup process with all the components in a backup environment (see Figure below).
- The backup server maintains the information about backup clients to be backed up and storage nodes to be used in a backup operation.
- The backup server retrieves the backup-related information from the backup catalog and, based on this information, instructs the storage node to load the appropriate backup media into the backup devices.
- Simultaneously, it instructs the backup clients to gather the data to be backed up and send it over the network to the assigned storage node.
- After the backup data is sent to the storage node, the client sends some backup metadata (the number of files, name of the files, storage node details, and so on) to the backup server.
- The storage node receives the client data, organizes it, and sends it to the backup device.
- The storage node then sends additional backup metadata (location of the data on the backup device, time of backup, and so on) to the backup server.
- The backup server updates the backup catalog with this information.

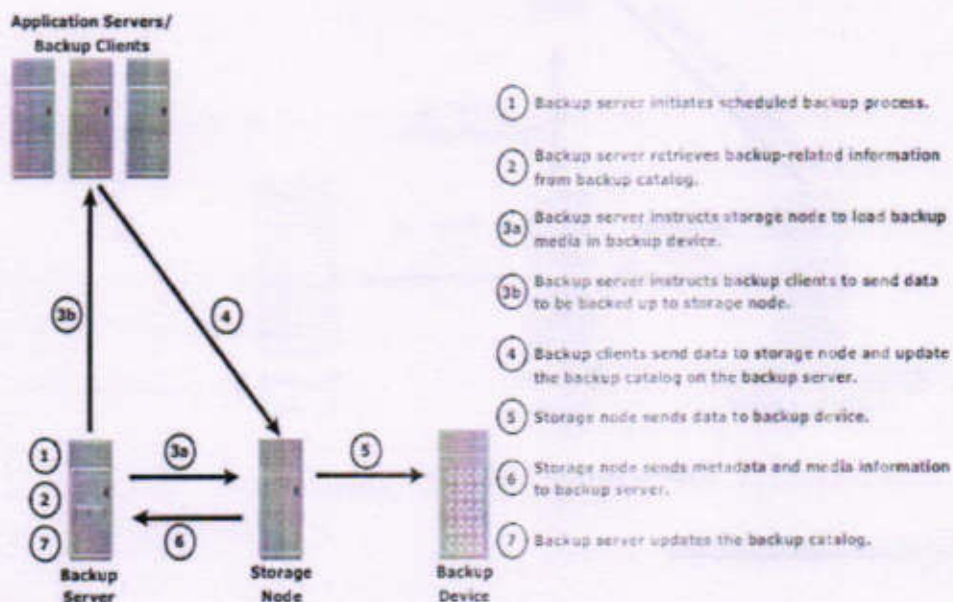


Figure: Backup Operation

- After the data is backed up, it can be restored when required.
- A restore process must be manually initiated from the client.

- Some backup software has a separate application for restore operations.
- These restore applications are usually accessible only to the administrators or backup operators.
- The below Figure shows a restore operation.
- Upon receiving a restore request, an administrator opens the restore application to view the list of clients that have been backed up.
- While selecting the client for which a restore request has been made, the administrator also needs to identify the client that will receive the restored data. Data can be restored on the same client for whom the restore request has been made or on any other client.
- The administrator then selects the data to be restored and the specified point in time to which the data has to be restored based on the RPO.
- Because all this information comes from the backup catalog, the restore application needs to communicate with the backup server.
- The backup server instructs the appropriate storage node to mount the specific backup media onto the backup device.
- Data is then read and sent to the client that has been identified to receive the restored data.
- Some restorations are successfully accomplished by recovering only the requested production data. For example, the recovery process of a spreadsheet is completed when the specific file is restored.
- In database restorations, additional data, such as log files, must be restored along with the production data. This ensures consistency for the restored data.
- In these cases, the RTO is extended due to the additional steps in the restore operation.

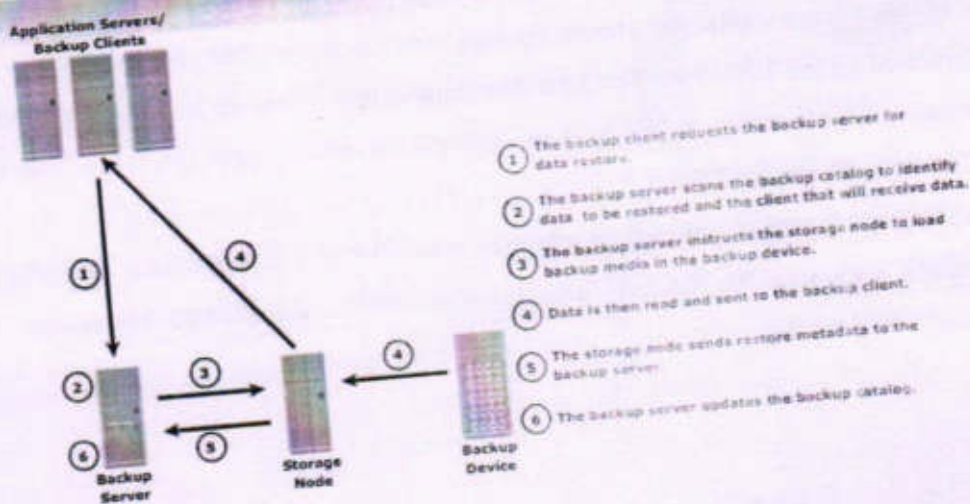


Figure: Restore Operation

Backup Topologies

- Three basic topologies are used in a backup environment:
 1. Direct attached backup
 2. LAN based backup, and
 3. SAN based backup.
- A **mixed topology** is also used by combining LAN based and SAN based topologies.
- In a **direct-attached backup**, a backup device is attached directly to the client. Only the metadata is sent to the backup server through the LAN. This configuration frees the LAN from backup traffic.
- The example shown in Fig 3.7 device is directly attached and dedicated to the backup client. As the environment grows, however, there will be a need for central management of all backup devices and to share the resources to optimize costs. An appropriate solution is to share the backup devices among multiple servers. Network-based topologies (LAN-based and SAN-based) provide the solution to optimize the utilization of backup devices.

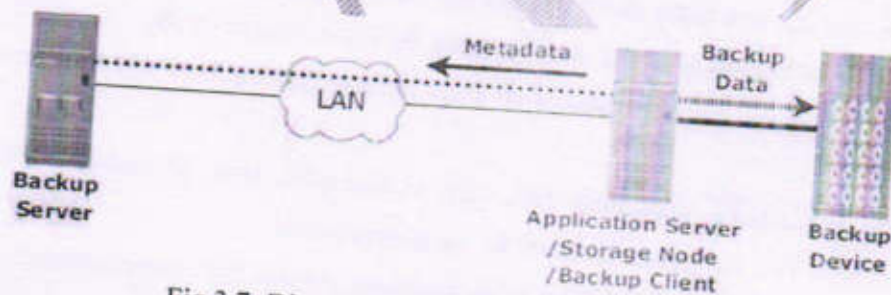


Fig 3.7: Direct-attached backup topology

- In **LAN-based backup**, the clients, backup server, storage node, and backup device are connected to the LAN (see Fig 3.8). The data to be backed up is transferred from the backup client (source), to the backup device (destination) over the LAN, which may affect network performance.
- This impact can be minimized by adopting a number of measures, such as configuring separate networks for backup and installing dedicated storage nodes for some application servers.

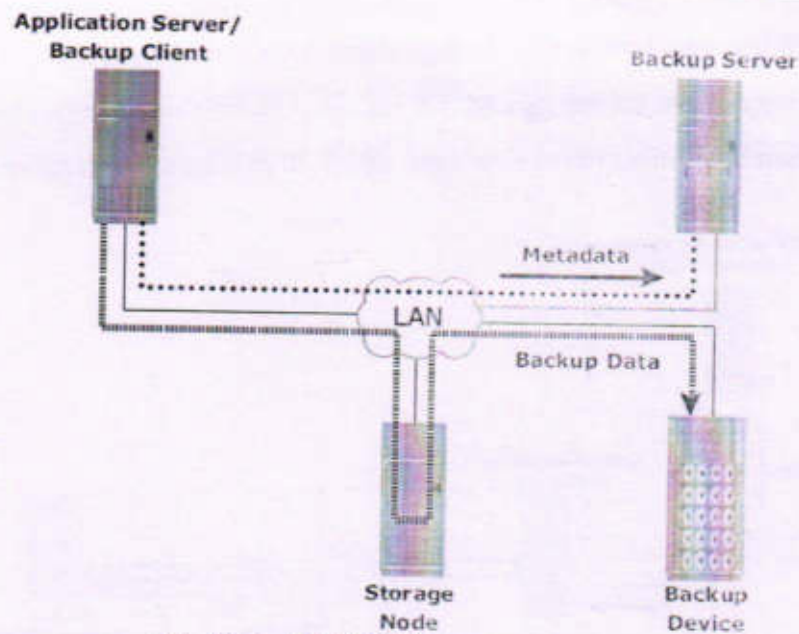


Fig 3.8: LAN-based backup topology

- The **SAN-based backup** is also known as the *LAN-free backup*. Fig 3.9 illustrates a SAN-based backup. The SAN-based backup topology is the most appropriate solution when a backup device needs to be shared among the clients. In this case the backup device and clients are attached to the SAN.
- In the example from Fig 3.9, a client sends the data to be backed up to the backup device over the SAN. Therefore, the backup data traffic is restricted to the SAN, and only the backup metadata is transported over the LAN. The volume of metadata is insignificant when compared to the production data; the LAN performance is not degraded in this configuration.

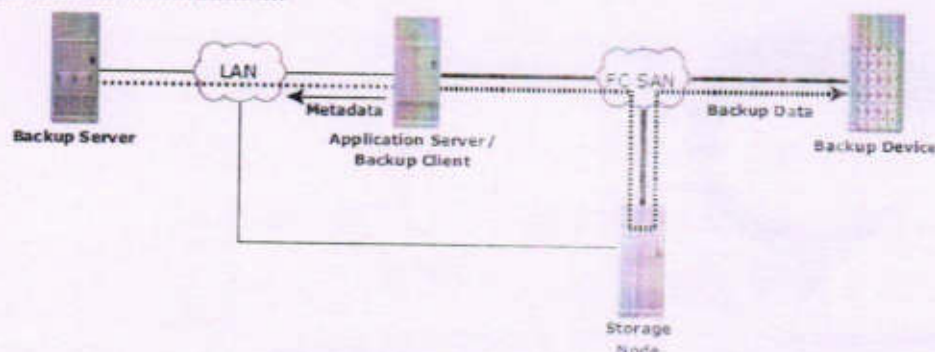


Fig 3.9: SAN-based backup topology

- The emergence of low-cost disks as a backup medium has enabled disk arrays to be attached to the SAN and used as backup devices. A tape backup of these data backups on the disks can be created and shipped offsite for disaster recovery and long-term

retention.

- The mixed topology uses both the LAN-based and SAN-based topologies, as shown in Fig 3.10. This topology might be implemented for several reasons, including cost, server location, reduction in administrative overhead, and performance considerations.

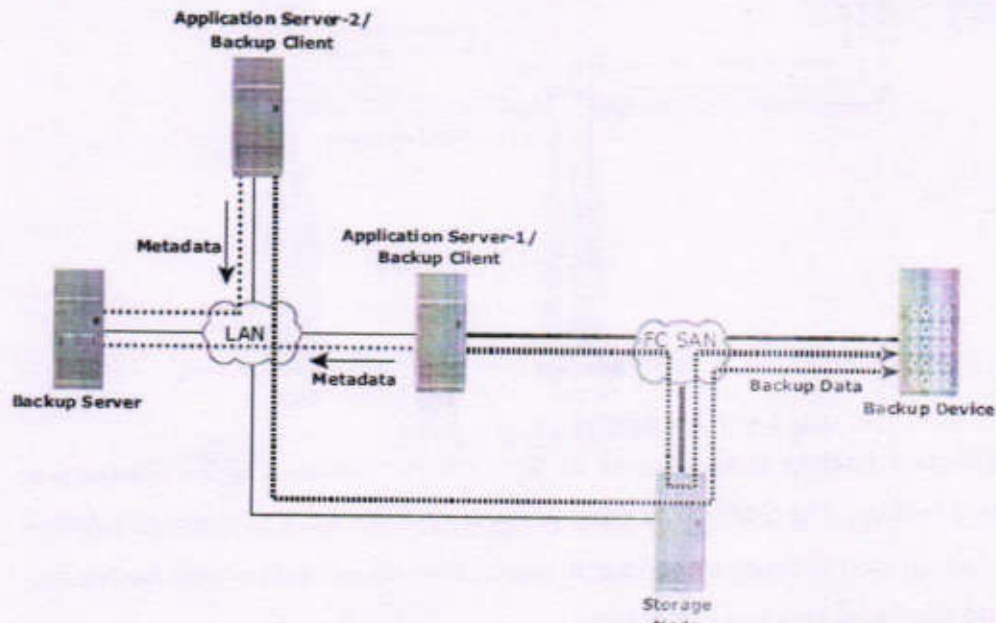


Fig 3.10: Mixed backup topology

SW

Module – 3: Backup, Archive, and Replication

Introduction:

1. Continuous access to information is a must for the smooth functioning of business operations today, as the cost of business disruption could be catastrophic.
2. There are many threats to information availability, such as natural disasters (e.g., flood, fire, earthquake), unplanned occurrences (e.g., cybercrime, human error, network and computer failure), and planned occurrences (e.g., upgrades, backup, restore) that result in the inaccessibility of information.
3. It is critical for businesses to define appropriate plans that can help them overcome these crises. Business continuity is an important process to define and implement these plans.
4. **Business continuity (BC)** is an integrated and enterprise-wide process that includes all activities (internal and external to IT) that a business must perform to mitigate the impact of planned and unplanned downtime.
5. BC entails preparing for, responding to, and recovering from a system outage that adversely affects business operations.
6. It involves proactive measures, such as business impact analysis and risk assessments, data protection, and security, and reactive countermeasures, such as disaster recovery and restart, to be invoked in the event of a failure.
7. The goal of a business continuity solution is to ensure the “information availability” required to conduct vital business operations.

Chapter Objective:

This chapter describes the factors that affect information availability. It also explains how to create an effective BC plan and design fault-tolerant mechanisms to protect against single points of failure.

11.1 Information Availability

Information availability (IA) refers to the ability of the infrastructure to function according to business expectations during its specified time of operation. Information availability ensures that people (employees, customers, suppliers, and partners) can access information whenever they need it. Information availability can be defined with the help of reliability, accessibility and timeliness.

1. **Reliability:** This reflects a component’s ability to function without failure, under stated conditions, for a specified amount of time.
2. **Accessibility:** This is the state within which the required information is accessible at the right place, to the right user. The period of time during which the system is in an accessible state is termed *system uptime*; when it is not accessible it is termed *system downtime*.
3. **Timeliness:** Defines the exact moment or the time window (a particular time of the day, week, month, and/or year as specified) during which information must be accessible. For example, if online access to an application is required between 8:00 am and 10:00 pm each day, any disruptions to data availability outside of this time slot are not considered to affect timeliness.

Causes of Information Unavailability

Various planned and unplanned incidents result in data unavailability.

1. **Planned outages** include installation/ integration/ maintenance of new hardware, software upgrades or patches, taking backups, application and data restores, facility operations (renovation and construction), and refresh/migration of the testing to the production environment.
2. **Unplanned outages** include failure caused by database corruption, component failure, and human errors. Another type of incident that may cause data unavailability is natural or man-made disasters such as flood, fire, earthquake, and contamination. As illustrated in Figure 11-1, the majority of outages are planned. Planned outages are expected and scheduled, but still cause data to be unavailable. Statistically, less than 1 percent is likely to be the result of an unforeseen disaster.

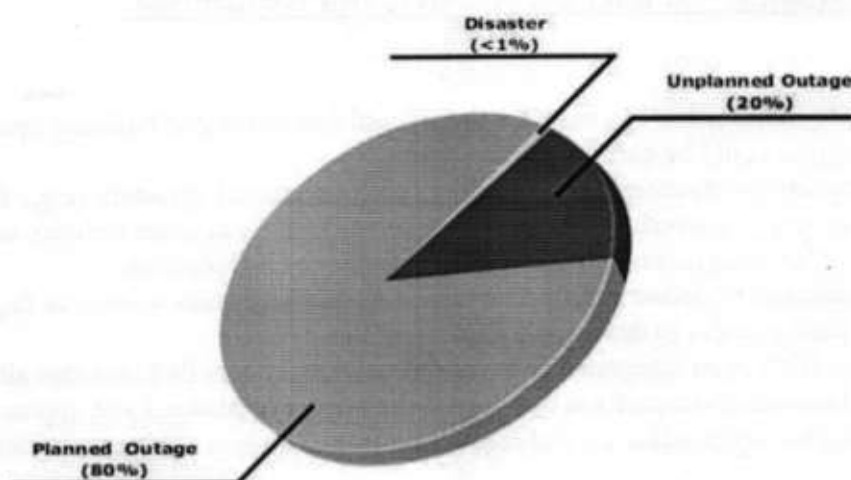


Figure 11-1: Disruptors of data availability

Measuring Information Availability

- Information availability relies on the availability of the hardware and software components of a data center. Failure of these components might disrupt information availability.
- A failure is the termination of a component's ability to perform a required function. The component's ability can be restored by performing an external corrective action, such as a manual reboot, a repair, or replacement of the failed component(s).
- Repair involves restoring a component to a condition that enables it to perform a required function within a specified time by using procedures and resources.
- Proactive risk analysis performed as part of the BC planning process considers the component failure rate and average repair time, which are measured by MTBF and MTTR:

- 1. Mean Time Between Failure (MTBF):** It is the average time available for a system or component to perform its normal operations between failures.
- 2. Mean Time To Repair (MTTR):** It is the average time required to repair a failed component. While calculating MTTR, it is assumed that the fault responsible for the failure is correctly identified and that the required spares and personnel are available. Note that a fault is a physical defect at the component level, which may result in data unavailability. MTTR includes the time required to do the following: detect the fault, mobilize the maintenance team, diagnose the fault, obtain the spare parts, repair, test, and resume normal operations.

IA is the fraction of a time period that a system is in a condition to perform its intended function upon demand. It can be expressed in terms of system uptime and downtime and measured as the amount or percentage of system uptime:

$$IA = \text{system uptime} / (\text{system uptime} + \text{system downtime})$$

In terms of MTBF and MTTR, IA could also be expressed as

$$IA = MTBF / (MTBF + MTTR)$$

Uptime per year is based on the exact timeliness requirements of the service, this calculation leads to the number of "9s" representation for availability metrics.

Table 11-1 lists the approximate amount of downtime allowed for a service to achieve certain levels of 9s availability.

Table 11-1: Availability Percentage and Allowable Downtime

UPTIME (%)	DOWNTIME (%)	DOWNTIME PER YEAR	DOWNTIME PER WEEK
98	2	7.3 days	3 hr 22 minutes
99	1	3.65 days	1 hr 41 minutes
99.8	0.2	17 hr 31 minutes	20 minutes 10 sec
99.9	0.1	8 hr 45 minutes	10 minutes 5 sec
99.99	0.01	52.5 minutes	1 minute
99.999	0.001	5.25 minutes	6 sec
99.9999	0.0001	31.5 sec	0.6 sec

Consequences of Downtime

Data unavailability, or downtime, results in loss of productivity, loss of revenue, poor financial performance, and damages to reputation. Loss of productivity reduces the output per unit of labor, equipment, and capital. Loss of revenue includes direct loss, compensatory payments, future revenue losses, billing losses, and investment losses.

The business impact of downtime is the sum of all losses sustained as a result of a given disruption. An important metric, *average cost of downtime per hour*, provides a key estimate in determining the appropriate BC solutions. It is calculated as follows:

Average cost of downtime per hour = average productivity loss per hour + average revenue loss per hour

Where:

Productivity loss per hour = (total salaries and benefits of all employees per week) / (average number of working hours per week)

Average revenue loss per hour = (total revenue of an organization per week) / (average number of hours per week that an organization is open for business)

The average downtime cost per hour may also include estimates of projected revenue loss due to other consequences such as damaged reputations and the additional cost of repairing the system.

BC Terminology

This section introduces and defines common terms related to BC operations and are used in the next few chapters to explain advanced concepts:

1. **Disaster recovery:** This is the coordinated process of restoring systems, data, and the infrastructure required to support key ongoing business operations in the event of a disaster. It is the process of restoring a previous copy of the data and applying logs or other necessary processes to that copy to bring it to a known point of consistency. Once all recoveries are completed, the data is validated to ensure that it is correct.
2. **Disaster restart:** This is the process of restarting business operations with mirrored consistent copies of data and applications.
3. **Recovery-Point Objective (RPO):** This is the point in time to which systems and data must be recovered after an outage. It defines the amount of data loss that a business can endure. A large RPO signifies high tolerance to information loss in a business. Based on the RPO, organizations plan for the minimum frequency with which a backup or replica must be made. For example, if the RPO is six hours, backups or replicas must be made at least once in 6 hours. Figure 11-2 shows various RPOs and their corresponding ideal recovery strategies. An organization can plan for an appropriate BC technology solution on the basis of the RPO it sets. For example:

- RPO of 24 hours:** This ensures that backups are created on an offsite tape drive every midnight. The corresponding recovery strategy is to restore data from the set of last backup tapes.
- RPO of 1 hour:** This ships database logs to the remote site every hour. The corresponding recovery strategy is to recover the database at the point of the last log shipment.
- RPO of zero:** This mirrors mission-critical data synchronously to a remote site.

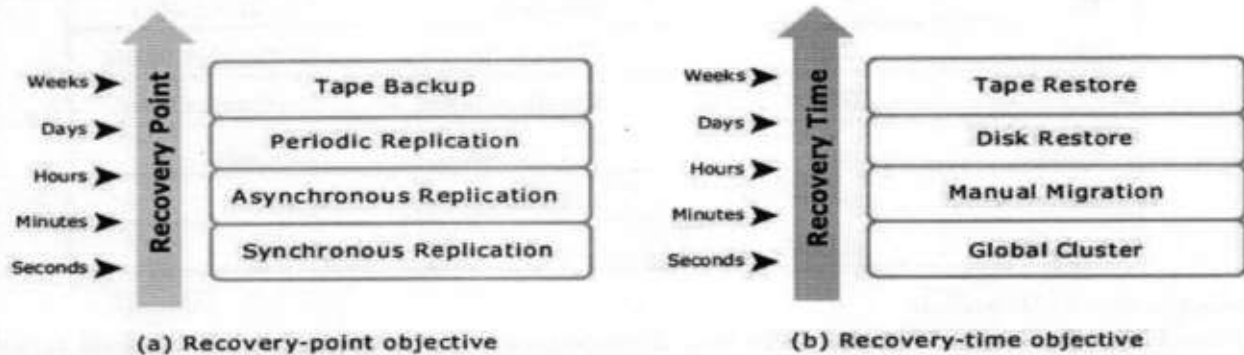


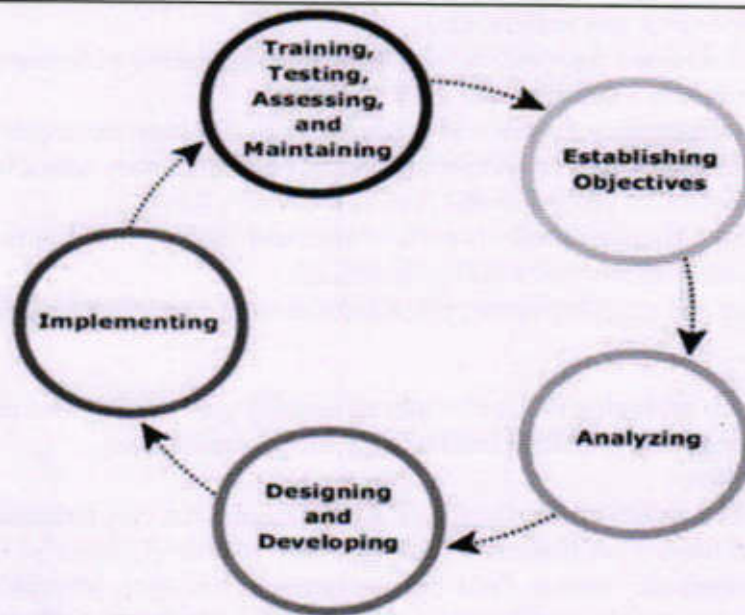
Figure 11-2: Strategies to meet RPO and RTO targets

- Recovery-Time Objective (RTO):** The time within which systems, applications, or functions must be recovered after an outage. It defines the amount of downtime that a business can endure and survive. Businesses can optimize disaster recovery plans after defining the RTO for a given data center or network. For example, if the RTO is two hours, then use a disk backup because it enables a faster restore than a tape backup. However, for an RTO of one week, tape backup will likely meet requirements. Some examples of RTOs and the recovery strategies to ensure data availability are listed below (refer to Figure 11-2):
 - RTO of 72 hours:** Restore from backup tapes at a cold site.
 - RTO of 12 hours:** Restore from tapes at a hot site.
 - RTO of 4 hours:** Use a data vault to a hot site.
 - RTO of 1 hour:** Cluster production servers with controller-based disk mirroring.
 - RTO of a few seconds:** Cluster production servers with bidirectional mirroring, enabling the applications to run at both sites simultaneously.

BC Planning Lifecycle

BC planning must follow a disciplined approach like any other planning process. Organizations today dedicate specialized resources to develop and maintain BC plans. From the conceptualization to the realization of the BC plan, a lifecycle of activities can be defined for the BC process. The BC planning lifecycle includes five stages (see Figure 11-3):

1. Establishing objectives
2. Analyzing
3. Designing and developing
4. Implementing
5. Training, testing, assessing, and maintaining



e 11-3: BC planning lifecycle

Several activities are performed at each stage of the BC planning lifecycle, including the following key activities:

1. Establishing objectives

- Determine BC requirements.
- Estimate the scope and budget to achieve requirements.
- Select a BC team by considering subject matter experts from all areas of the business, whether internal or external.
- Create BC policies.

2. Analyzing

- Collect information on data profiles, business processes, infrastructure support, dependencies, and frequency of using business infrastructure.
- Identify critical business needs and assign recovery priorities.
- Create a risk analysis for critical areas and mitigation strategies.
- ▼ Conduct a Business Impact Analysis (BIA).
- Create a cost and benefit analysis based on the consequences of data unavailability.
- Evaluate options.

3. Designing and developing

- Define the team structure and assign individual roles and responsibilities. For example, different teams are formed for activities such as emergency response, damage assessment, and infrastructure and application recovery.
- Design data protection strategies and develop infrastructure.
- Develop contingency scenarios.
- Develop emergency response procedures.
- Detail recovery and restart procedures.

4. Implementing

- Implement risk management and mitigation procedures that include backup, replication, and management of resources.
- Prepare the disaster recovery sites that can be utilized if a disaster affects the primary data center.
- Implement redundancy for every resource in a data center to avoid single points of failure.

5. Training, testing, assessing, and maintaining

- Train the employees who are responsible for backup and replication of business-critical data on a regular basis or whenever there is a modification in the BC plan.
- Train employees on emergency response procedures when disasters are declared.
- Train the recovery team on recovery procedures based on contingency scenarios.
- Perform damage assessment processes and review recovery plans.
- Test the BC plan regularly to evaluate its performance and identify its limitations.
- Assess the performance reports and identify limitations.
- Update the BC plans and recovery/restart procedures to reflect regular changes within the data center.

Failure Analysis

Failure analysis involves analyzing the data center to identify systems that are susceptible to a single point of failure and implementing fault-tolerance mechanisms such as redundancy.

Single Point of Failure

A single point of failure refers to the failure of a component that can terminate the availability of the entire system or IT service. Figure 11-4 illustrates the possibility of a single point of failure in a system with various components: server, network, switch, and storage array. The figure depicts a system setup in which an application running on the server provides an interface to the client and performs I/O operations. The client is connected to the server through an IP network, the server is connected to the storage array through a FC connection, an HBA installed at the server sends or receives data to and from a storage array, and an FC switch connects the HBA to the storage port.

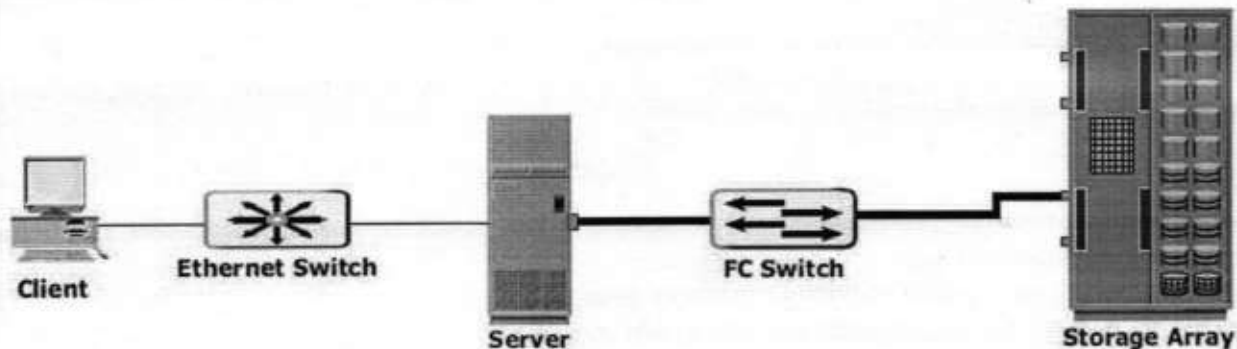


Figure 11-4: Single point of failure

In a setup where each component must function as required to ensure data availability, the failure of a single component causes the failure of the entire data center or an application, resulting in disruption of business operations. In this example, several single points of failure can be identified. The single HBA on the server, the server itself, the IP network, the FC switch, the storage array ports, or even the storage array could become potential single points of failure. To avoid single points of failure, it is essential to implement a fault-tolerant mechanism.

Fault Tolerance

To mitigate a single point of failure, systems are designed with redundancy, such that the system will fail only if all the components in the redundancy group fail. This ensures that the failure of a single component does not affect data availability. Figure 11-5 illustrates the fault-tolerant implementation of the system just described (and shown in Figure 11-4).

Data centers follow stringent guidelines to implement fault tolerance. Careful analysis is performed to eliminate every single point of failure. In the example shown in Figure 11-5, all enhancements in the infrastructure to mitigate single points of failures are emphasized:

1. Configuration of multiple HBAs to mitigate single HBA failure.

2. Configuration of multiple fabrics to account for a switch failure.
3. Configuration of multiple storage array ports to enhance the storage array's availability.
4. RAID configuration to ensure continuous operation in the event of disk failure.
5. Implementing a storage array at a remote site to mitigate local site failure.
6. Implementing server (host) clustering, a fault-tolerance mechanism whereby two or more servers in a cluster access the same set of volumes. Clustered servers exchange *heartbeats* to inform each other about their health. If one of the servers fails, the other server takes up the complete workload.

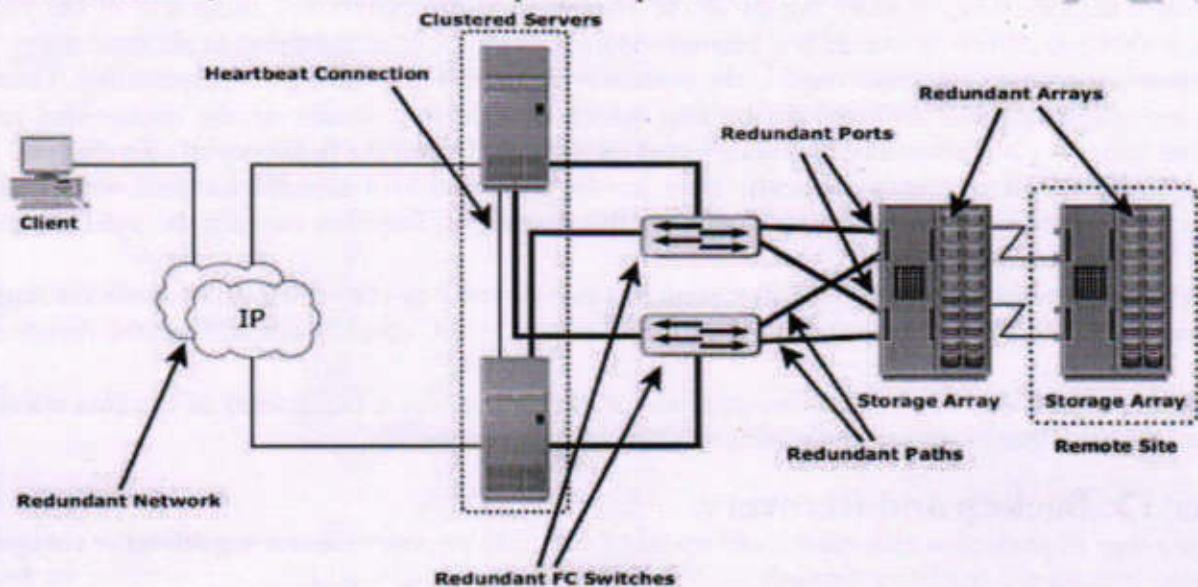


Figure 11-5: Implementation of fault tolerance

Multipathing Software

Configuration of multiple paths increases the data availability through path failover. If servers are configured with one I/O path to the data there will be no access to the data if that path fails. Redundant paths eliminate the path to become single points of failure. Multiple paths to data also improve I/O performance through load sharing and maximize server, storage, and data path utilization.

In practice, merely configuring multiple paths does not serve the purpose. Even with multiple paths, if one path fails, I/O will not reroute unless the system recognizes that it has an alternate path. Multipathing software provides the functionality to recognize and utilize alternate I/O path to data. Multipathing software also manages the load balancing by distributing I/Os to all available, active paths.

Business Impact Analysis

A *business impact analysis (BIA)* identifies and evaluates financial, operational, and service impacts of a disruption to essential business processes. Selected functional areas are evaluated to determine resilience of the infrastructure to support information availability. The BIA process leads to a report detailing the incidents and their impact over business functions. The impact may be specified in terms of money or in terms of time. Based on the potential impacts associated with downtime, businesses can prioritize and implement countermeasures to mitigate the likelihood of such disruptions. These are detailed in the BC plan.

A BIA includes the following set of tasks:

1. Identify the key business processes critical to its operation.
2. Determine the attributes of the business process in terms of applications, databases, and hardware and software requirements.
3. Estimate the costs of failure for each business process.
4. Calculate the maximum tolerable outage and define RTO and RPO for each business process.

5. Establish the minimum resources required for the operation of business processes.
6. Determine recovery strategies and the cost for implementing them.
7. Optimize the backup and business recovery strategy based on business priorities.
8. Analyze the current state of BC readiness and optimize future BC planning.

BC Technology Solutions

After analyzing the business impact of an outage, designing appropriate solutions to recover from a failure is the next important activity. One or more copies of the original data are maintained using any of the following strategies, so that data can be recovered and business operations can be restarted using an alternate copy:

1. **Backup and recovery:** Backup to tape is the predominant method of ensuring data availability. These days, low-cost, high-capacity disks are used for backup, which considerably speeds up the backup and recovery process. The frequency of backup is determined based on RPO, RTO, and the frequency of data changes.
2. **Storage array-based replication (local):** Data can be replicated to a separate location within the same storage array. The replica is used independently for BC operations. Replicas can also be used for restoring operations if data corruption occurs.
3. **Storage array-based replication (remote):** Data in a storage array can be replicated to another storage array located at a remote site. If the storage array is lost due to a disaster, BC operations start from the remote storage array.
4. **Host-based replication:** The application software or the LVM ensures that a copy of the data managed by them is maintained either locally or at a remote site for recovery purposes.

Chapter 12: Backup and Recovery

A *backup* is a copy of production data, created and retained for the sole purpose of recovering deleted or corrupted data. With growing business and regulatory demands for data storage, retention, and availability, organizations are faced with the task of backing up an ever-increasing amount of data.

Organizations must ensure that the right data is in the right place at the right time. Evaluating backup technologies, recovery, and retention requirements for data and applications is an essential step to ensure successful implementation of the backup and recovery solution. The solution must facilitate easy recovery and retrieval from backups and archives as required by the business.

Backup Purpose

Backups are performed to serve three purposes: disaster recovery, operational backup, and archival.

Disaster Recovery

Backups can be performed to address disaster recovery needs. The backup copies are used for restoring data at an alternate site when the primary site is incapacitated due to a disaster. Based on RPO and RTO requirements, organizations use different backup strategies for disaster recovery. When a tape-based backup method is used as a disaster recovery strategy, the backup tape media is shipped and stored at an offsite location. These tapes can be recalled for restoration at the disaster recovery site.

Operational Backup

Data in the production environment changes with every business transaction and operation. *Operational backup* is a backup of data at a point in time and is used to restore data in the event of data loss or logical corruptions that may occur during routine processing. The majority of restore requests in most organizations fall in this category. For example, it is common for a user to accidentally delete an important e-mail or for a file to become corrupted, which can be restored from operational backup.

Operational backups are created for the active production information by using incremental or differential backup techniques, detailed later in this chapter. An example of an operational backup is a backup performed for a production database just before a bulk batch update. This ensures the availability of a clean copy of the production database if the batch update corrupts the production database.

Archival

Backups are also performed to address archival requirements. Although CAS has emerged as the primary solution for archives, traditional backups are still used by small and medium enterprises for long-term preservation of transaction records, e-mail messages, and other business records required for regulatory compliance.

Apart from addressing disaster recovery, archival, and operational requirements, backups serve as a protection against data loss due to physical damage of a storage device, software failures, or virus attacks. Backups can also be used to protect against accidents such as a deletion or intentional data destruction.

Backup Considerations

The amount of data loss and downtime that a business can endure in terms of RTO and RPO are the primary considerations in selecting and implementing a specific backup strategy. Another consideration is the retention period, which defines the duration for which a business needs to retain the backup copies. Some data is retained for years and some only for a few days. For example, data backed up for archival is retained for a longer period than data backed up for operational recovery.

It is also important to consider the backup media type, based on the retention period and data accessibility. Organizations must also consider the granularity of backups, explained later in this chapter. The development of a backup strategy must include a decision about the most appropriate time for performing a backup in order to minimize any disruption to production operations. Similarly, the location and time of the restore operation must be considered, along with file characteristics and data compression that influences the backup process.

Location, size, and number of files should also be considered, as they may affect the backup process. Location is an important consideration for the data to be backed up. Many organizations have dozens of heterogeneous platforms supporting complex solutions. Consider a data warehouse environment that uses backup data from many sources. The backup process must address these sources in terms of transactional and content integrity. This process must be coordinated with all heterogeneous platforms on which the data resides.

File size also influences the backup process. Backing up large-size files (example: ten 1 MB files) may use less system resources than backing up an equal amount of data comprising a large number of small-size files (example: ten thousand 1 KB files). The backup and restore operation takes more time when a file system contains many small files.

Like file size, the number of files to be backed up also influences the backup process. For example, in incremental backup, a file system containing one million files with a 10 percent daily change rate will have to create 100,000 entries in the backup catalog, which contains the table of contents for the backed up data set and information about the backup session. This large number of entries in the file system affects the performance of the backup and restore process because it takes a long time to search through a file system.

Backup performance also depends on the media used for the backup. The time-consuming operation of starting and stopping in a tape-based system affects backup performance, especially while backing up a large number of small files.

Data compression is widely used in backup systems because compression saves space on the media. Many backup devices, such as tape drives, have built-in support for hardware-based data compression. To effectively use this, it is important to understand the characteristics of the data.

Backup Granularity

Backup granularity depends on business needs and required RTO/RPO. Based on granularity, backups can be categorized as full, cumulative, and incremental.

Most organizations use a combination of these three backup types to meet their backup and recovery requirements. Figure 12-1 depicts the categories of backup granularity.

1. Full backup is a backup of the complete data on the production volumes at a certain point in time. A full backup copy is created by copying the data on the production volumes to a secondary storage device.

Synthetic (or constructed) full backup is another type of backup that is used in implementations where the production volume resources cannot be exclusively reserved for a backup process for extended periods to perform a full backup.

It is usually created from the most recent full backup and all the incremental backups performed after that full backup. A synthetic full backup enables a full backup copy to be created offline without disrupting the I/O operation on the production volume. This also frees up network resources from the backup process, making them available for other production uses.

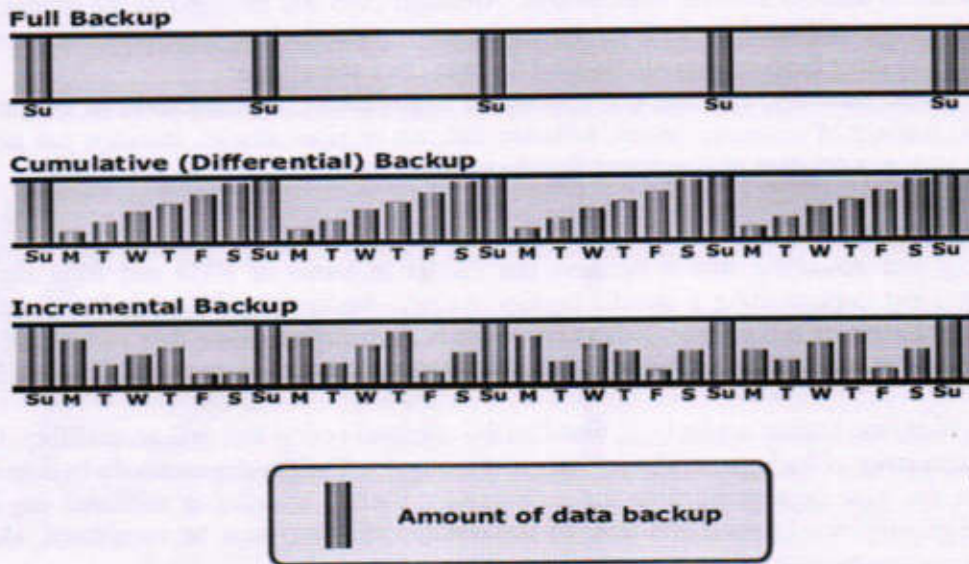


Figure 12-1: Backup granularity levels

2. **Incremental backup** copies the data that has changed since the last full or incremental backup, whichever has occurred more recently. This is much faster (because the volume of data backed up is restricted to changed data), but it takes longer to restore.
3. **Cumulative (or differential) backup** copies the data that has changed since the last full backup. This method takes longer than incremental backup but is faster to restore.

Restore operations vary with the granularity of the backup.

A full backup provides a single repository from which data can be easily restored.

The process of restoration from an incremental backup requires the last full backup and all the incremental backups available until the point of restoration.

A restore from a cumulative backup requires the last full backup and the most recent cumulative backup.

Restoring from an incremental backup

Figure 12-2 illustrates an example of an incremental backup and restoration.

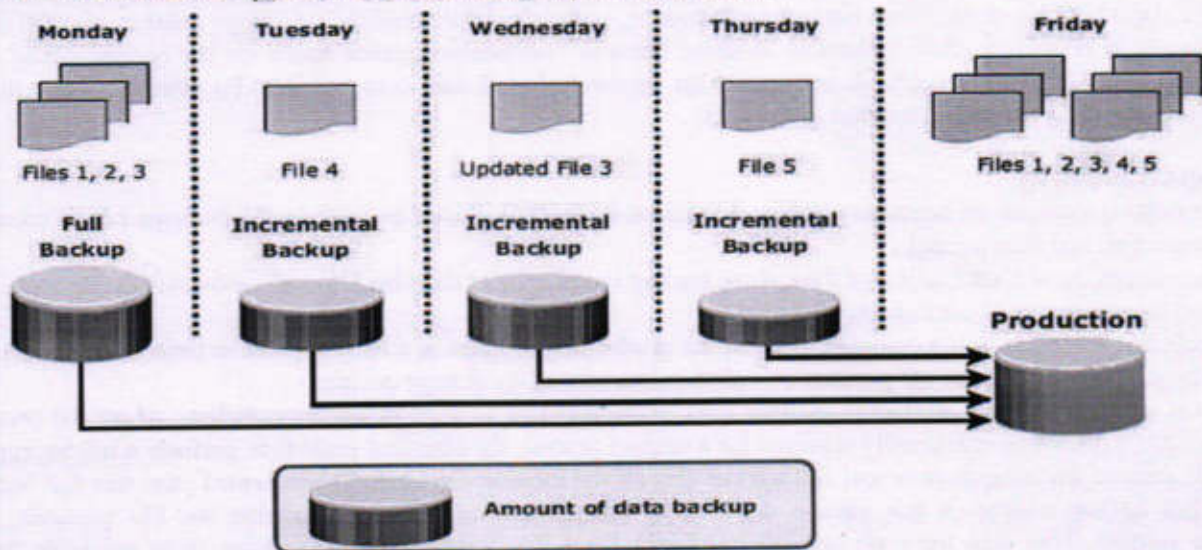


Figure 12-2: Restoring from an incremental backup

In this example, a full backup is performed on Monday evening. Each day after that, an incremental backup is performed. On Tuesday, a new file (File 4 in the figure) is added, and no other files have changed. Consequently, only File 4 is copied during the incremental backup performed on Tuesday evening. On Wednesday, no new files are added, but File 3 has been modified. Therefore, only the modified File 3 is copied during the incremental backup on Wednesday evening. Similarly, the incremental backup on Thursday copies only File 5. On Friday morning, there is data corruption, which requires data restoration from the backup. The first step toward data restoration is restoring all data from the full backup of Monday evening. The next step is applying the incremental backups of Tuesday, Wednesday, and Thursday. In this manner, data can be successfully restored to its previous state, as it existed on Thursday evening.

Restoring a cumulative backup

Figure 12-3 illustrates an example of cumulative backup and restoration.

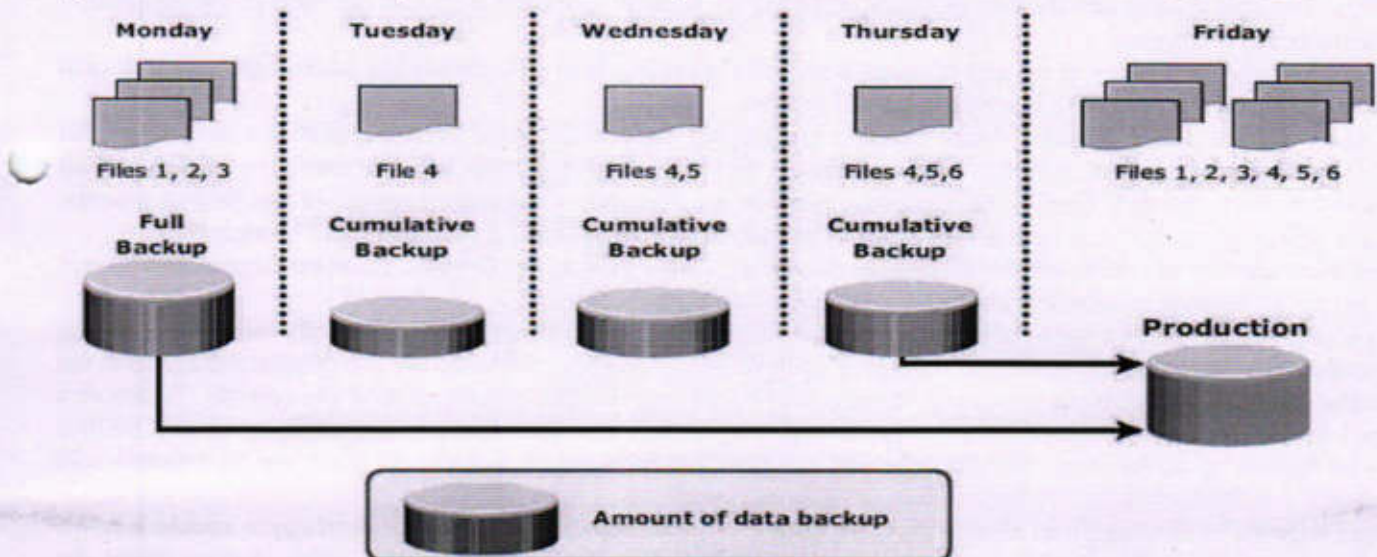


Figure 12-3: Restoring a cumulative backup

In this example, a full backup of the business data is taken on Monday evening. Each day after that, a cumulative backup is taken. On Tuesday, File 4 is added and no other data is modified since the previous full backup of Monday evening. Consequently, the cumulative backup on Tuesday evening copies only File 4. On Wednesday, File 5 is added. The cumulative backup taking place on Wednesday evening copies both File 4 and File 5 because these files have been added or modified since the last full backup. Similarly, on Thursday, File 6 is added. Therefore, the cumulative backup on Thursday evening copies all three files: File 4, File 5, and File 6.

On Friday morning, data corruption occurs that requires data restoration using backup copies. The first step in restoring data from a cumulative backup is restoring all data from the full backup of Monday evening. The next step is to apply only the latest cumulative backup — Thursday evening.

Recovery Considerations

RPO and RTO are major considerations when planning a backup strategy. RPO defines the tolerable limit of data loss for a business and specifies the time interval between two backups. In other words, the RPO determines backup frequency. For example, if application A requires an RPO of one day, it would need the data to be backed up at least once every day. The retention period for a backup is also derived from an RPO specified for operational recovery. For example, users of application "A" may request to restore the application data from its operational backup copy, which was created a month ago. This determines the retention period for the backup. The RPO for application A can therefore range from one day to one month based on operational recovery needs. However, the organization may choose to retain the backup for a longer period of time because of internal policies or external factors, such as regulatory directives.

If short retention periods are specified for backups, it may not be possible to recover all the data needed for the requested recovery point, as some data may be older than the retention period. Long retention periods can be defined for all backups, making it possible to meet any RPO within the defined retention periods. However, this requires a large storage space,

which translates into higher cost. Therefore, it is important to define the retention period based on an analysis of all the restore requests in the past and the allocated budget.

RTO relates to the time taken by the recovery process. To meet the defined RTO, the business may choose to use a combination of different backup solutions to minimize recovery time. In a backup environment, RTO influences the type of backup media that should be used. For example, recovery from data streams multiplexed in tape takes longer to complete than recovery from tapes with no multiplexing.

Organizations perform more full backups than they actually need because of recovery constraints. Cumulative and incremental backups depend on a previous full backup. When restoring from tape media, several tapes are needed to fully recover the system. With a full backup, recovery can be achieved with a lower RTO and fewer tapes.

Backup Methods

Hot backup and cold backup are the two methods deployed for backup. They are based on the state of the application when the backup is performed.

In a *hot backup*, the application is up and running, with users accessing their data during the backup process. In a *cold backup*, the application is not active during the backup process.

The backup of online *production data* becomes more challenging because data is actively being used and changed. An open file is locked by the operating system and is not copied during the backup process until the user closes it. The backup application can back up open files by retrying the operation on files that were opened earlier in the backup process. During the backup process, it may be possible that files opened earlier will be closed and a retry will be successful.

The maximum number of retries can be configured depending on the backup application. However, this method is not considered robust because in some environments certain files are always open.

In such situations, the backup application provides *open file agents*. These agents interact directly with the operating system and enable the creation of consistent copies of open files. In some environments, the use of open file agents is not enough. For example, a database is composed of many files of varying sizes, occupying several file systems. To ensure a consistent database backup, all files need to be backed up in the same state. That does not necessarily mean that all files need to be backed up at the same time, but they all must be synchronized so that the database can be restored with consistency.

Consistent backups of databases can also be done by using a cold backup. This requires the database to remain inactive during the backup. Of course, the disadvantage of a cold backup is that the database is inaccessible to users during the backup process.

Hot backup is used in situations where it is not possible to shut down the database. This is facilitated by *database backup agents* that can perform a backup while the database is active. The disadvantage associated with a hot backup is that the agents usually affect overall application performance.

A *point-in-time (PIT)* copy method is deployed in environments where the impact of downtime from a cold backup or the performance resulting from a hot backup is unacceptable. A pointer-based PIT copy consumes only a fraction of the storage space and can be created very quickly. A pointer-based PIT copy is implemented in a disk-based solution whereby a virtual LUN is created and holds pointers to the data stored on the production LUN or save location. In this method of backup, the database is stopped or frozen momentarily while the PIT copy is created. The PIT copy is then mounted on a secondary server and the backup occurs on the primary server. This technique is detailed in Chapter 13. To ensure consistency, it is not enough to back up only production data for recovery. Certain attributes and properties attached to a file, such as permissions, owner, and other metadata, also need to be backed up. These attributes are as important as the data itself and must be backed up for consistency. Backup of boot sector and partition layout information is also critical for successful recovery.

In a disaster recovery environment, *bare-metal recovery (BMR)* refers to a backup in which all metadata, system information, and application configurations are appropriately backed up for a full system recovery. BMR builds the base system, which includes partitioning, the file system layout, the operating system, the applications, and all the relevant configurations. BMR recovers the base system first, before starting the recovery of data files. Some BMR technologies can recover a server onto dissimilar hardware.

Backup Process

A backup system uses client/server architecture with a backup server and multiple backup clients. The backup server manages the backup operations and maintains the backup catalog, which contains information about the backup process

and backup metadata. The backup server depends on backup clients to gather the data to be backed up. The backup clients can be local to the server or they can reside on another server, presumably to back up the data visible to that server. The backup server receives backup metadata from the backup clients to perform its activities. Figure 12-4 illustrates the backup process.

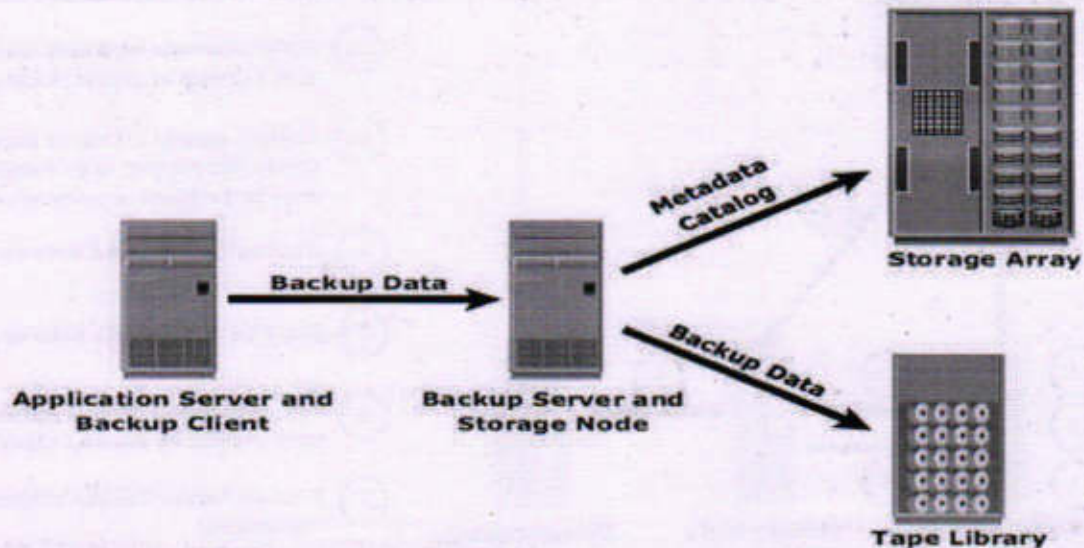


Figure 12-4: Backup architecture and process

The storage node is responsible for writing data to the backup device (in a backup environment, a storage node is a host that controls backup devices). Typically, the storage node is integrated with the backup server and both are hosted on the same physical platform. A backup device is attached directly to the storage node's host platform. Some backup architecture refers to the storage node as the *media server* because it connects to the storage device. Storage nodes play an important role in backup planning because they can be used to consolidate backup servers.

The backup process is based on the policies defined on the backup server, such as the time of day or completion of an event. The backup server then initiates the process by sending a request to a backup client (backups can also be initiated by a client). This request instructs the backup client to send its metadata to the backup server, and the data to be backed up to the appropriate storage node. On receiving this request, the backup client sends the metadata to the backup server. The backup server writes this metadata on its metadata catalog.

The backup client also sends the data to the storage node, and the storage node writes the data to the storage device.

After all the data is backed up, the storage node closes the connection to the backup device. The backup server writes backup completion status to the metadata catalog.

Backup and Restore Operations

When a backup process is initiated, significant network communication takes place between the different components of a backup infrastructure. The backup server initiates the backup process for different clients based on the backup schedule configured for them. For example, the backup process for a group of clients may be scheduled to start at 3:00 am every day.

The backup server coordinates the backup process with all the components in a backup configuration (see Figure 12-5). The backup server maintains the information about backup clients to be contacted and storage nodes to be used in a backup operation. The backup server retrieves the backup-related information from the backup catalog and, based on this information, instructs the storage node to load the appropriate backup media into the backup devices. Simultaneously, it instructs the backup clients to start scanning the data, package it, and send it over the network to the assigned storage node. The storage node, in turn, sends metadata to the backup server to keep it updated about the media being used in the backup process. The backup server continuously updates the backup catalog with this information.

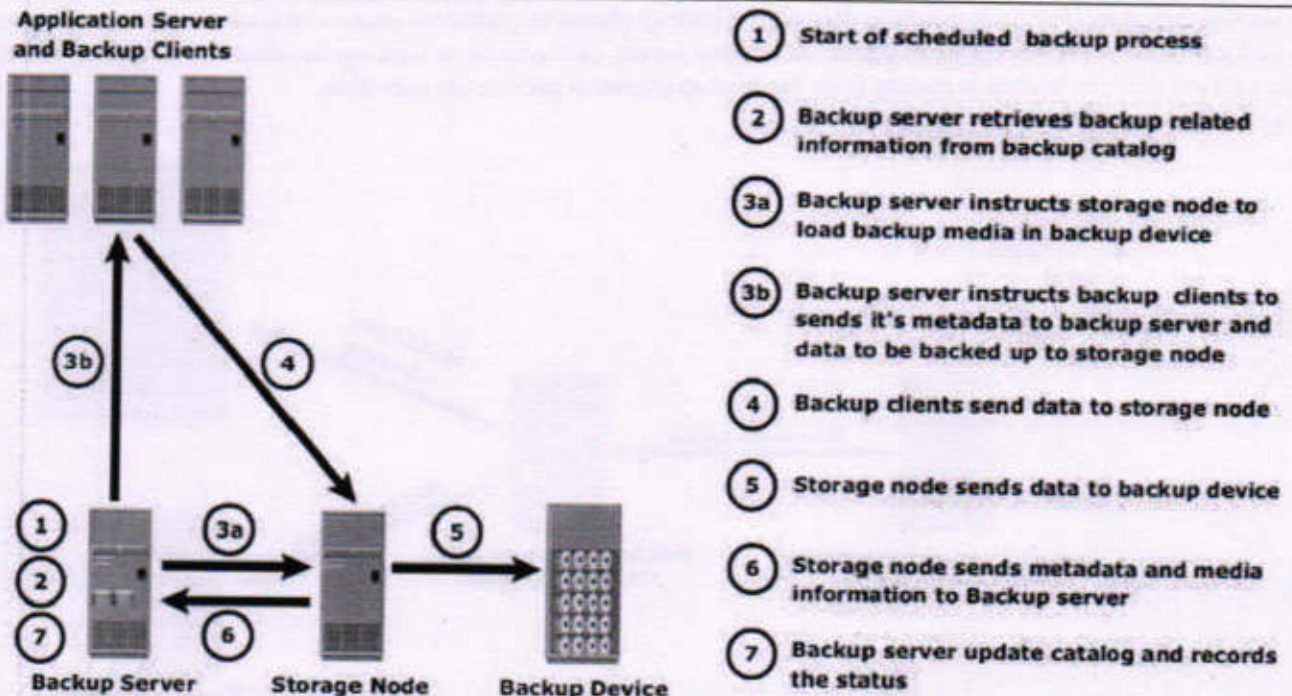


Figure 12-5: Backup operation

After the data is backed up, it can be restored when required. A restore process must be manually initiated. Some backup software has a separate application for restore operations. These restore applications are accessible only to the administrators.

Figure 12-6 depicts a restore process.

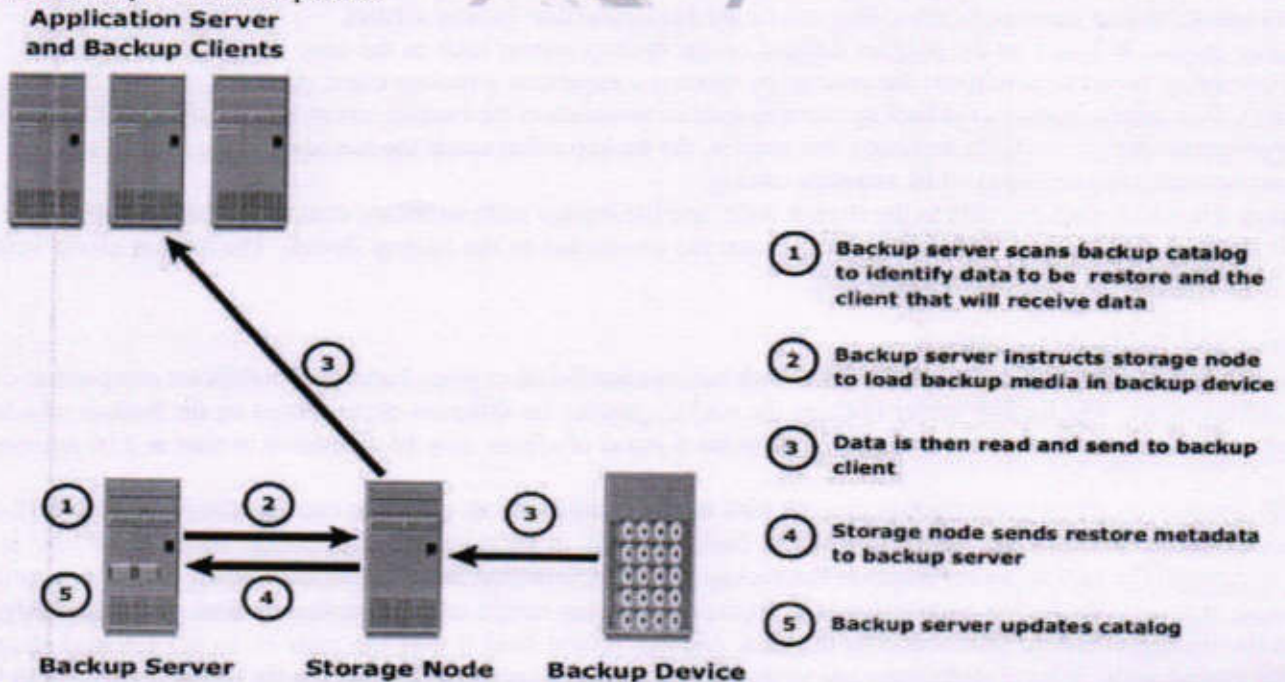


Figure 12-6: Restore operation

Upon receiving a restore request, an administrator opens the restore application to view the list of clients that have been backed up. While selecting the client for which a restore request has been made, the administrator also needs to identify

the client that will receive the restored data. Data can be restored on the same client for whom the restore request has been made or on any other client.

The administrator then selects the data to be restored and the specified point in time to which the data has to be restored based on the RPO. Note that because all of this information comes from the backup catalog, the restore application must also communicate to the backup server.

The administrator first selects the data to be restored and initiates the restore process. The backup server, using the appropriate storage node, then identifies the backup media that needs to be mounted on the backup devices. Data is then read and sent to the client that has been identified to receive the restored data.

Backup Topologies

Three basic topologies are used in a backup environment: direct attached backup, LAN based backup, and SAN based backup. A mixed topology is also used by combining LAN based and SAN based topologies.

1. In a *direct-attached backup*, a backup device is attached directly to the client. Only the metadata is sent to the backup server through the LAN. This configuration frees the LAN from backup traffic. The example shown in Figure 12-7 depicts use of a backup device that is not shared. As the environment grows, however, there will be a need for central management of all backup devices and to share the resources to optimize costs.

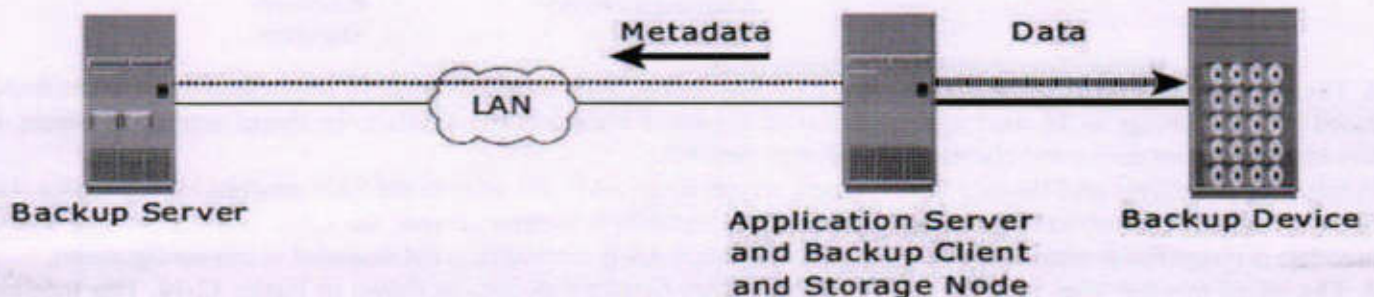


Figure 12-7: Direct-attached backup topology

2. In *LAN-based backup*, all servers are connected to the LAN and all storage devices are directly attached to the storage node (see Figure 12-8). The data to be backed up is transferred from the backup client (source), to the backup device (destination) over the LAN, which may affect network performance. Streaming across the LAN also affects network performance of all systems connected to the same segment as the backup server. Network resources are severely constrained when multiple clients access and share the same tape library unit (TLU).

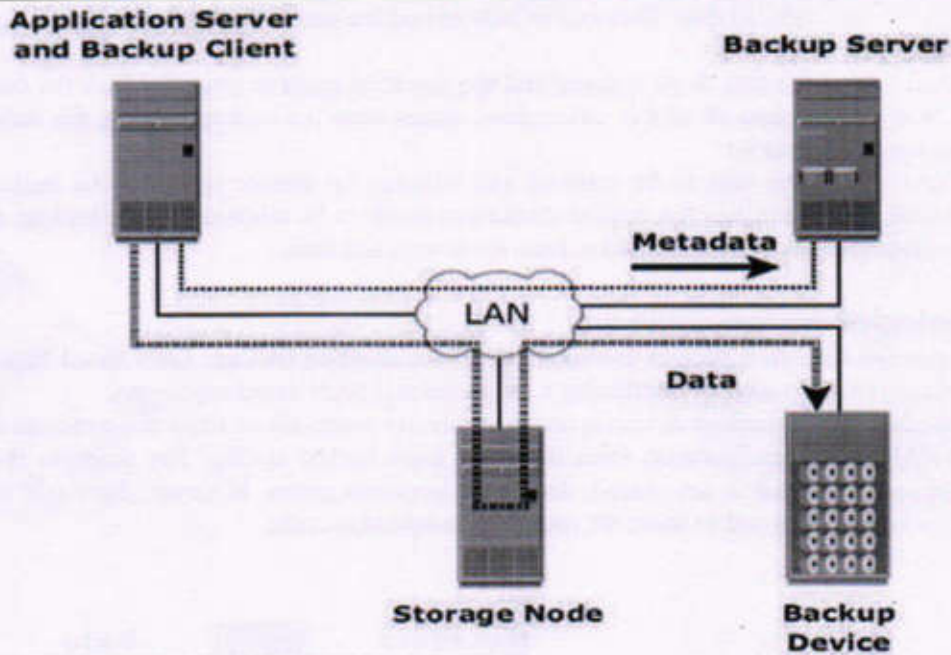


Figure 12-8: LAN-based backup topology

3. The **SAN-based backup** is also known as the **LAN-free backup**. Figure 12-9 illustrates a SAN-based backup. The SAN-based backup topology is the most appropriate solution when a backup device needs to be shared among the clients. In this case the backup device and clients are attached to the SAN.

In this example, clients read the data from the mail servers in the SAN and write to the SAN attached backup device. The backup data traffic is restricted to the SAN, and backup metadata is transported over the LAN. However, the volume of metadata is insignificant when compared to production data. LAN performance is not degraded in this configuration.

4. The **mixed topology** uses both the LAN-based and SAN-based topologies, as shown in Figure 12-10. This topology might be implemented for several reasons, including cost, server location, reduction in administrative overhead, and performance considerations.

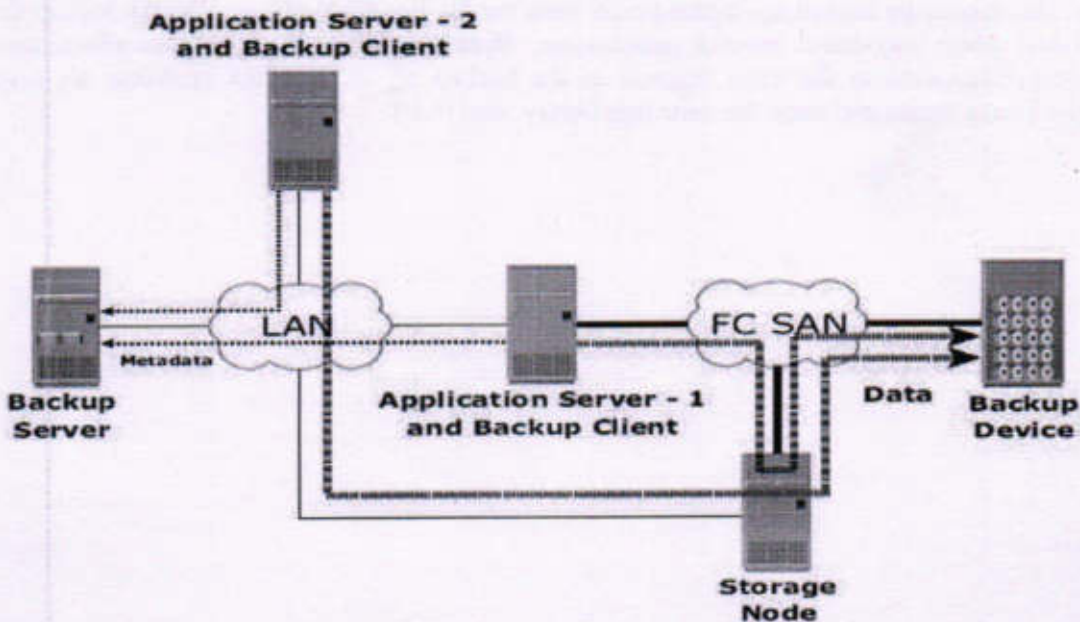


Figure 12-10: Mixed backup topology

12.8.1 Serverless Backup

Serverless backup is a LAN-free backup methodology that does not involve a backup server to copy data. The copy may be created by a network-attached controller, utilizing a SCSI extended copy or an appliance within the SAN. These backups are called serverless because they use SAN resources instead of host resources to transport backup data from its source to the backup device, reducing the impact on the application server.

Backup in NAS Environments

The use of NAS heads imposes a new set of considerations on the backup and recovery strategy in NAS environments. NAS heads use a proprietary operating system and file system structure supporting multiple file-sharing protocols.

In the NAS environment, backups can be implemented in four different ways: Server based, Serverless, Network Data Management Protocol (NDMP) in either NDMP 2-way and NDMP 3-way.

1. In *application server-based backup*, the NAS head retrieves data from storage over the network and transfers it to the backup client running on the application server. The backup client sends this data to a storage node, which in turn writes the data to the backup device. This results in overloading the network with the backup data and the use of production (application) server resources to move backup data. Figure 12-11 illustrates server-based backup in the NAS environment.

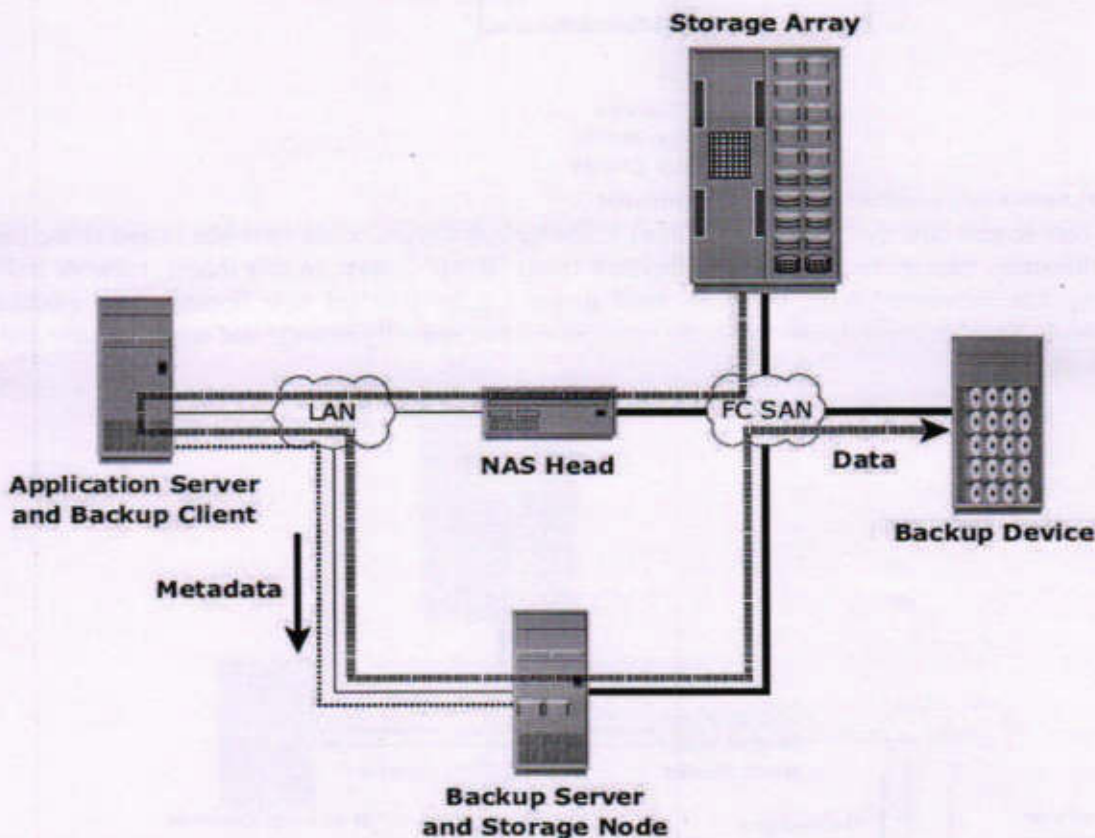


Figure 12-11: Server-based backup in NAS environment

2. In *serverless backup*, the network share is mounted directly on the storage node. This avoids overloading the network during the backup process and eliminates the need to use resources on the production server. Figure 12-12 illustrates serverless backup in the NAS environment. In this scenario, the storage node, which is also a backup client, reads the data from the NAS head and writes it to the backup device without involving the application server. Compared to the previous solution, this eliminates one network hop.

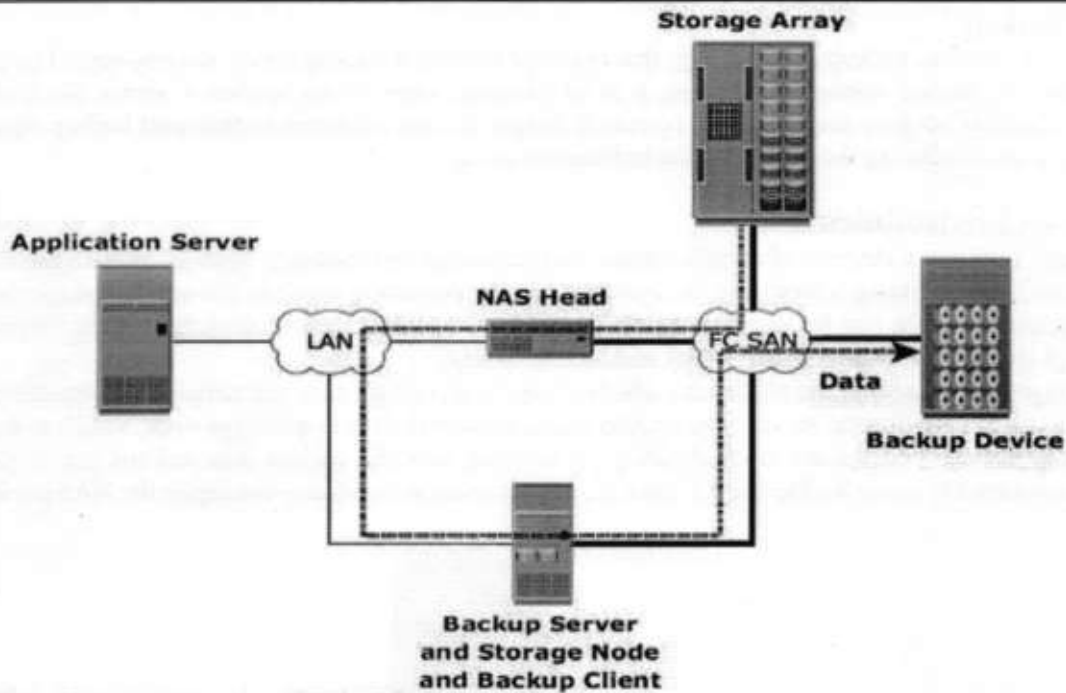


Figure 12-12: Serverless backup in NAS environment

3. In NDMP, backup data is sent directly from the NAS head to the backup device, while metadata is sent to the backup server. Figure 12-13 illustrates backup in the NAS environment using NDMP 2-way. In this model, network traffic is minimized by isolating data movement from the NAS head to the locally attached tape library. Only metadata is transported on the network. This backup solution meets the strategic need to centrally manage and control distributed data while minimizing network traffic.

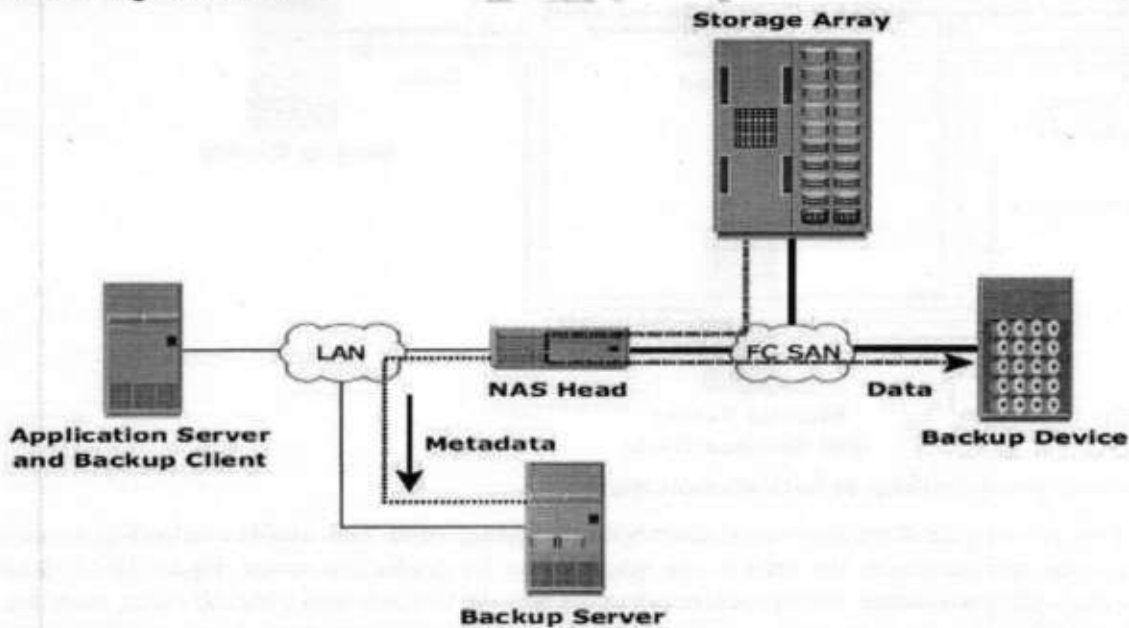


Figure 12-13: NDMP 2-way in NAS environment

4. In an NDMP 3-way file system, data is not transferred over the public network. A separate private backup network must be established between all NAS heads and the “backup” NAS head to prevent any data transfer on the public network in order to avoid any congestion or affect production operations. Metadata and NDMP control data is still transferred across the public network. Figure 12-14 depicts NDMP 3-way backup.

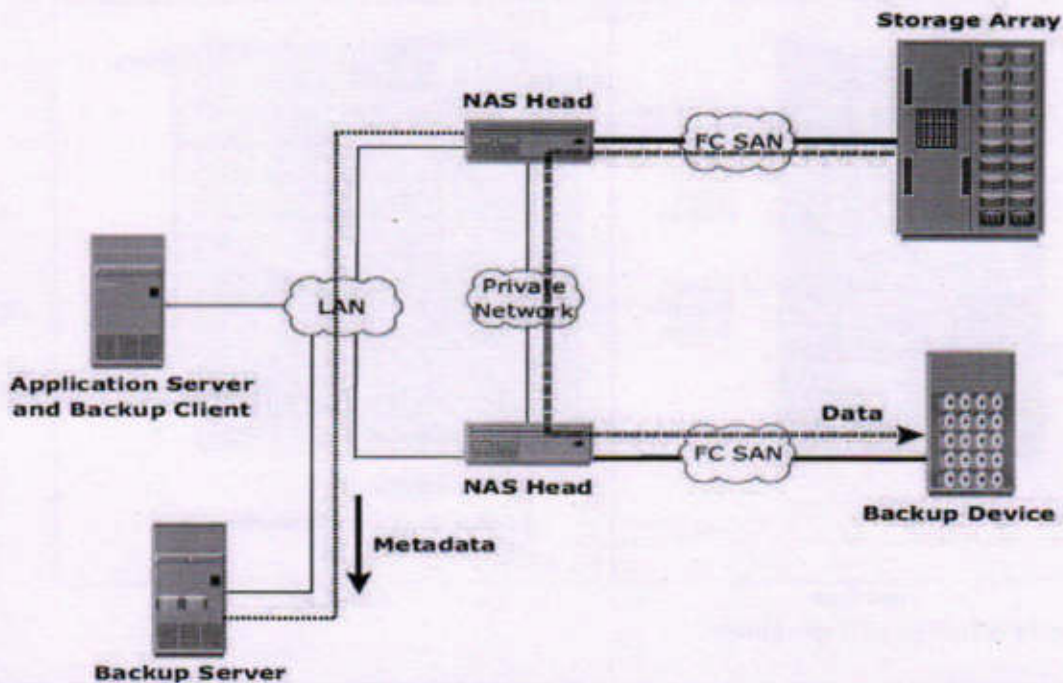


Figure 12-14: NDMP 3-way in NAS environment

Backup Technologies

A wide range of technology solutions are currently available for backup. Tapes and disks are the two most commonly used backup media. The tape technology has matured to scale to enterprise demands, whereas backup to disk is emerging as a viable option with the availability of low-cost disks. Virtual tape libraries use disks as backup medium emulating tapes, providing enhanced backup and recovery capabilities.

Backup to Tape

Tapes, a low-cost technology, are used extensively for backup. Tape drives are used to read/write data from/to a tape cartridge. Tape drives are referred to as sequential, or linear, access devices because the data is written or read sequentially.

Tape Mounting is the process of inserting a tape cartridge into a tape drive. The tape drive has motorized controls to move the magnetic tape around, enabling the head to read or write data.

Several types of tape cartridges are available. They vary in size, capacity, shape, number of reels, density, tape length, tape thickness, tape tracks, and supported speed. Today, a tape cartridge is composed of a magnetic tape with single or dual reels in a plastic enclosure.

Physical Tape Library

The physical tape library provides housing and power for a number of tape drives and tape cartridges, along with a robotic arm or picker mechanism. The backup software has intelligence to manage the robotic arm and entire backup process.

Tape drives read and write data from and to a tape. *Tape cartridges* are placed in the *slots* when not in use by a tape drive.

Robotic arms are used to move tapes around the library, such as moving a tape drive into a slot. Another type of slot called a *mail or import/export slot* is used to add or remove tapes from the library without opening the access doors (refer to Figure 12-15 Front View) because opening the access doors causes a library to go offline. In addition, each physical component in a tape library has an individual *element address* that is used as an addressing mechanism for moving tapes around the library.

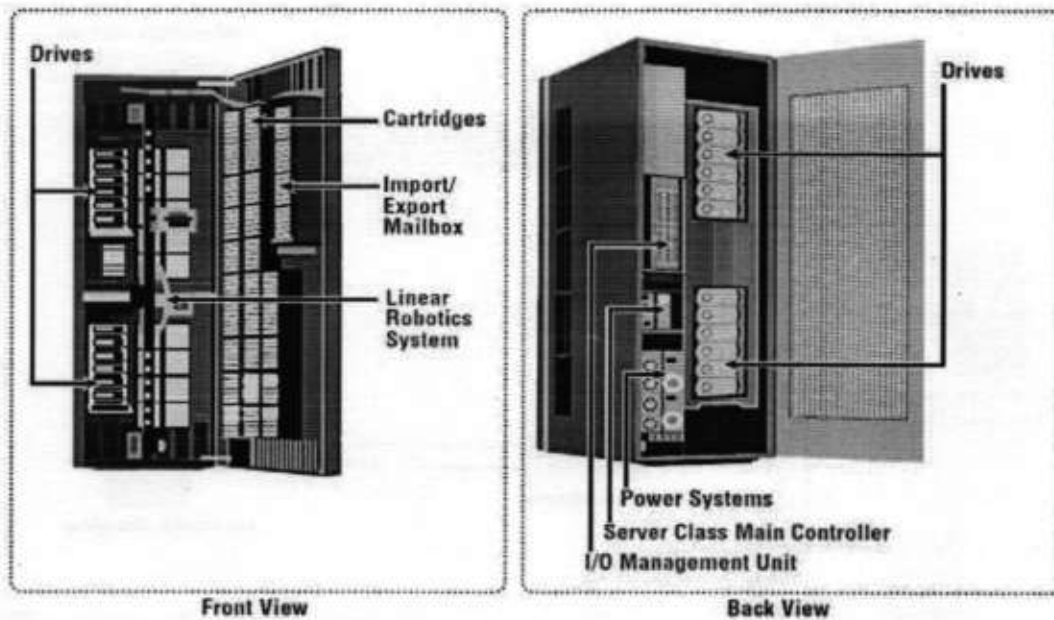


Figure 12-15: Physical tape library

Tape drive *streaming* or *multiple streaming* writes data from multiple streams on a single tape to keep the drive busy. Shown in Figure 12-16, multiple streaming improves media performance, but it has an associated disadvantage. The backup data is interleaved because data from multiple streams is written on it. Consequently, the data recovery time is increased.

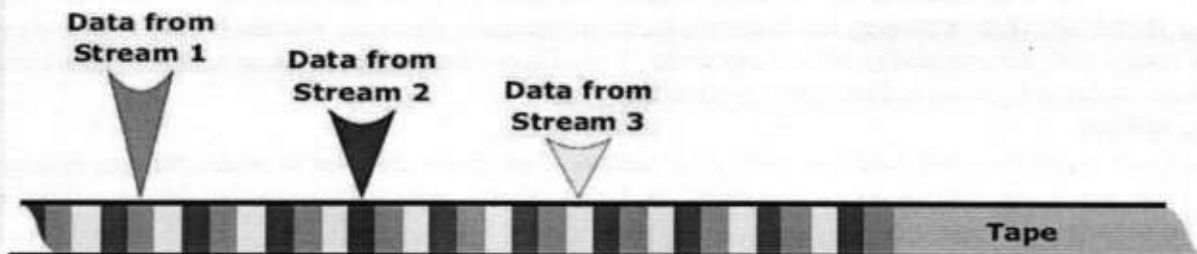


Figure 12-16: Multiple streams on tape media

Limitations of Tape

Tapes are primarily used for long-term offsite storage because of their low cost. Tapes must be stored in locations with a controlled environment to ensure preservation of the media and prevent data corruption. Data access in a tape is sequential, which can slow backup and recovery operations. Physical transportation of the tapes to offsite locations also adds management overhead.

Backup to Disk

Disks have now replaced tapes as the primary device for storing backup data because of their performance advantages. Backup-to-disk systems offer ease of implementation, reduced cost, and improved quality of service. Apart from performance benefits in terms of data transfer rates, disks also offer faster recovery when compared to tapes.

Backing up to disk storage systems offers clear advantages due to their inherent random access and RAID-protection capabilities. In most backup environments, backup to disk is used as a staging area where the data is copied temporarily before transferring or staging it to tapes later. This enhances backup performance. Some backup products allow for backup images to remain on the disk for a period of time even after they have been staged. This enables a much faster restore.

Virtual Tape Library

A *virtual tape library (VTL)* has the same components as that of a physical tape library except that the majority of the components are presented as virtual resources. For backup software, there is no difference between a physical tape library

and a virtual tape library. Figure 12-18 shows a virtual tape library. Virtual tape libraries use disks as backup media. Emulation software has a database with a list of virtual tapes, and each virtual tape is assigned a portion of a LUN on the disk. A virtual tape can span multiple LUNs if required.

Similar to a physical tape library, a robot mount is performed when a backup process starts in a virtual tape library. However, unlike a physical tape library, where this process involves some mechanical delays, in a virtual tape library it is almost instantaneous. Even the *load to ready* time is much less than in a physical tape library.

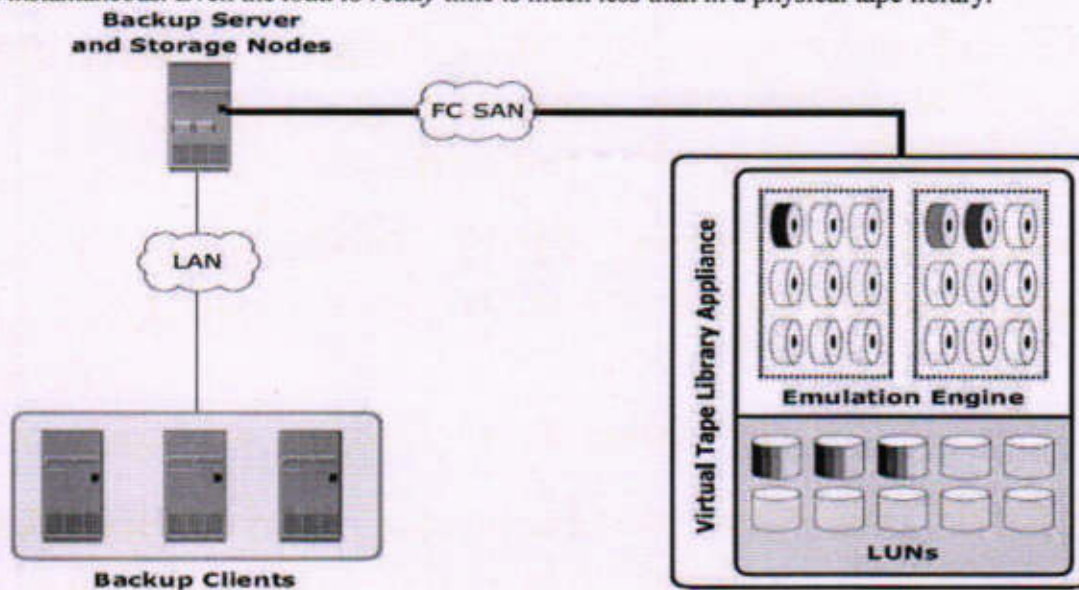


Figure 12-18: Virtual tape library

After the virtual tape is mounted and the tape drive is positioned, the virtual tape is ready to be used, and backup data can be written to it. Unlike a physical tape library, the virtual tape library is not constrained by the shoe shining effect. In most cases, data is written to the virtual tape immediately. When the operation is complete, the backup software issues a rewind command and then the tape can be unmounted. This rewind is also instantaneous. The virtual tape is then unmounted, and the virtual robotic arm is instructed to move it back to a virtual slot.

Advantages

Using virtual tape offers several advantages over both physical tapes and disks. Compared to physical tape, virtual tape offers better single stream performance, better reliability, and random disk access characteristics. Backup and restore operations are sequential by nature, but they benefit from the disk's random access characteristics because they are always online and ready to be used, improving backup and recovery times. Virtual tape does not require the usual maintenance tasks associated with a physical tape drive, such as periodic cleaning and drive calibration. Compared to backup-to-disk devices, virtual tapes offer easy installation and administration and inherent offsite capabilities. In addition, virtual tapes do not require any additional modules or changes on the backup software.

MODULE - 03

BUSINESS CONTINUITY

INTRODUCTION TO BUSINESS CONTINUITY

1) Define Information availability. Explain the causes of information unavailability and consequence of downtime?

⇒ Information availability:-

* Information availability (IA) refers to the ability of an IT infrastructure to function according to business expectations during its specified time of operation.

* IA ensures that people (employees, customers, suppliers and partners) can access information whenever they need it.

* IA can be defined in terms of

a) Accessibility:- Information should be accessible at the right place, to the right user.

b) Reliability:- Information should be reliable and correct in all aspects. It is "the same" as what was stored, and there is no alternation or corruption to the information.

c) Timeliness:- Defines the exact moment or the time window during which information must be accessible.

Causes Of Information Unavailability:-

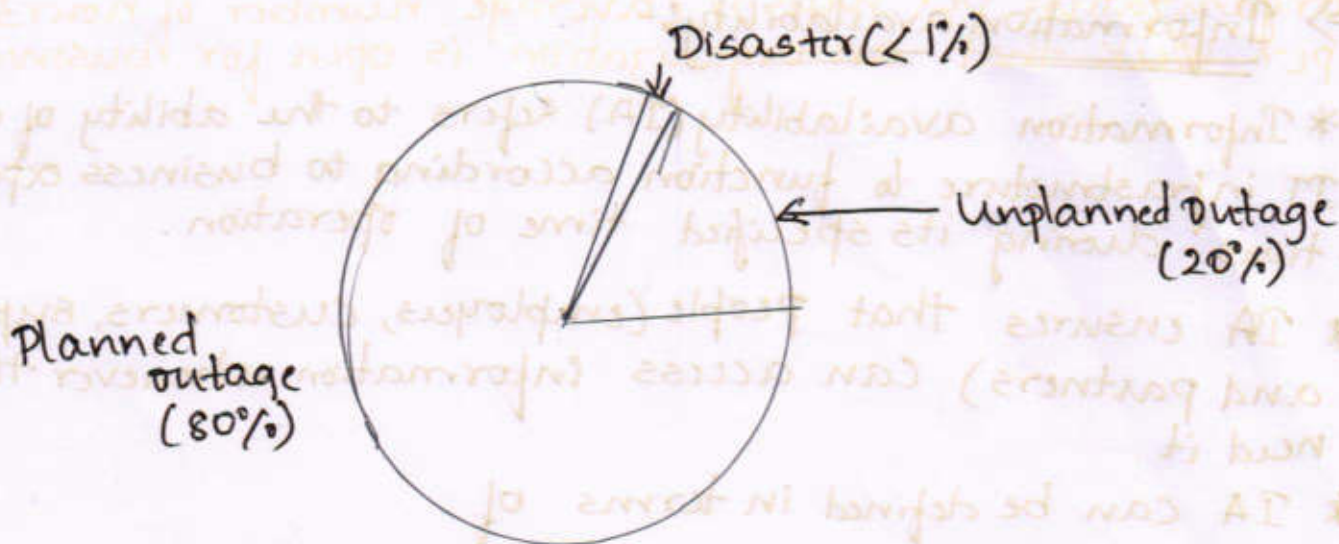
* Various planned and unplanned incidents results in information unavailability

* Planned outages include installation, integration, maintenance of new hardware, slw upgrades, taking backups application and data restores, facility operations and refresh /migration of the testing to the production environment.

* Unplanned outages includes failure of physical and virtual components.

* Another type of incident that may cause data unavailability is natural or manmade disasters, such as fire, earthquake.

* As illustrated in the following Figure, the majority of outages are planned.



Consequences of Downtime:

- * loss of productivity includes reduced output per unit of labor, equipment and capital.
 - * loss of revenue includes direct loss, compensatory payments, future revenue loss, billing loss, and investment loss.
 - * Poor financial performance affects revenue recognition, cash flow, discounts, payment guarantees, credit rating and stock price.
 - * Damages to reputations may result in a loss of confidences or trustworthiness with customers, suppliers, financial markets, banks and business partners.
 - * Other possible consequences of downtime include the cost of additional equipment payment, overtime and extra shipping.
 - * Average cost of downtime per hours, provides a key estimate in determining the appropriate BC solⁿ.
- It is calculated as follows:
- * Average cost of downtime per hour = avg productivity loss per hour + avg revenue loss per hour.

where

* Productivity loss per hour = (total salaries and benefits of all employees per week) / (average number of working hours per week)

* Average revenue loss per hour = (total revenue of an organization per week) / (average number of hours per week that an organization is open for business).

Productivity loss per hour

Average revenue loss per hour

Q2) Explain MTBF & MTTR along with Information availability matrix?

Ans: * Measuring Information Availability:-

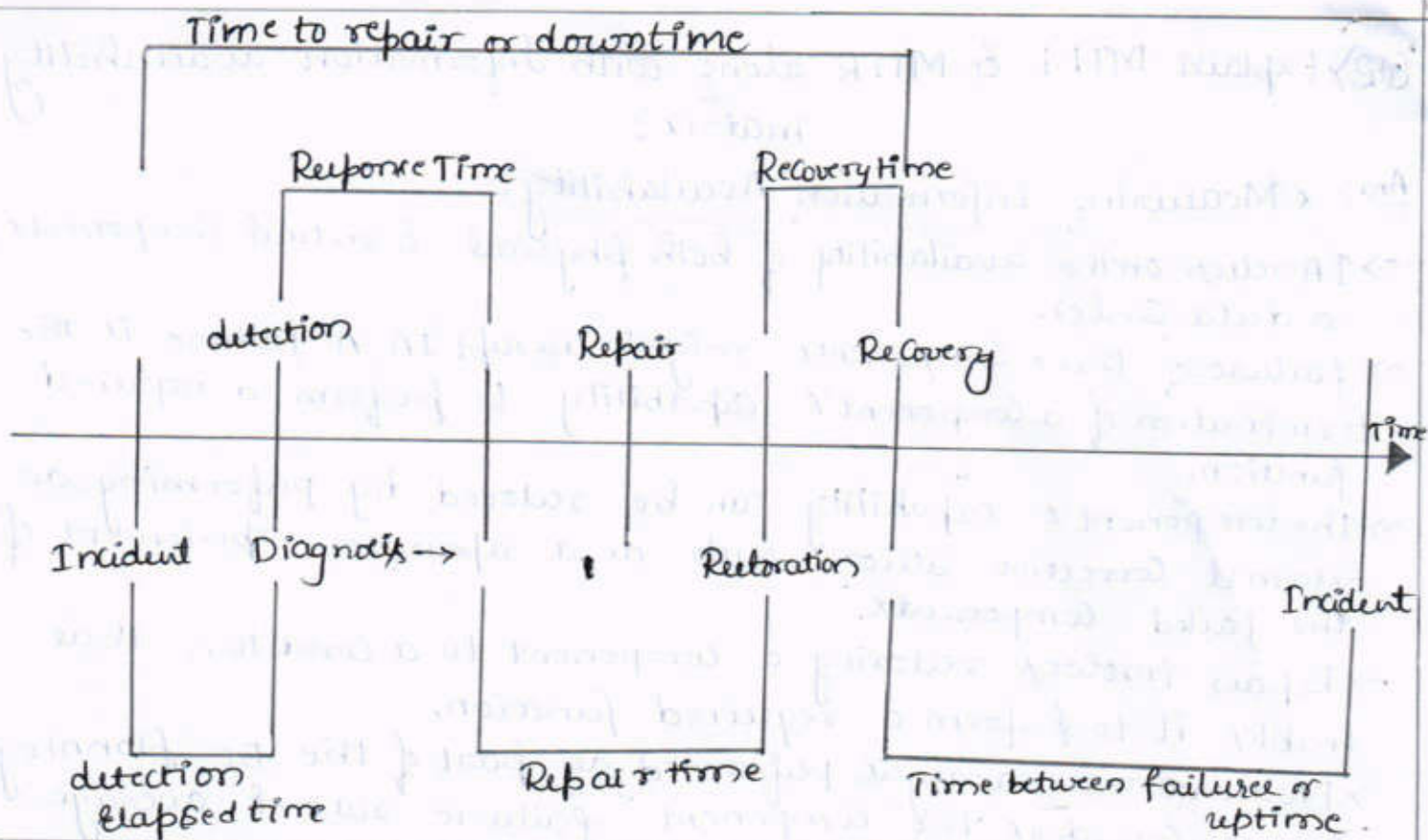
- IA relies on the availability of both physical & virtual components of a data center.
- failure of these components might disrupt IA. A failure is the termination of a component's capability to perform a required function.
- The component's capability can be restored by performing an external corrective action, such as a repair, or replacement of the failed components.
- Repair involves restoring a component to a condition that enables it to perform a required function.
- Proactive risk analysis, performed as part of the BC planning process, considers the component failure rate & average repair time, which are measured by:-
 - ① Mean time between failure (MTBF).
 - ② Mean time to repair (MTTR).

1) Mean Time Between Failure (MTBF):-

- * It is the average time available for a system or component to perform its normal operations between failures.
- * It is the measure of system or component reliability & is usually expressed in hours.

2) Mean Time To Repair (MTTR):-

- * It is the average time required to repair a failed component.
- * While calculating MTTR, it is assumed that the fault responsible for the failure is correctly identified and the required spares & personnel are available.
- * A fault is a physical defect at the component level, which may result in information unavailability.
- * MTTR includes the total time required to do the following activities:-
 - Detect the fault, the repair team, diagnose the fault, obtain the spare parts, test, and restore the data.



- * IA is the time period during which a system is in a condition to perform its intended function upon demand.
- * It can be expressed in terms of system uptime and downtime and measured as the amount or percentage of system uptime:

$$\rightarrow IA = \text{System uptime} / (\text{System uptime} + \text{System downtime})$$

where

- * System uptime is the period of time during which the system is in an accessible state;
- * when it is not accessible, it is termed as system downtime.
- * In terms of MTBF and MTR, IA could also be expressed as:-

$$\rightarrow IA = \text{MTBF} / (\text{MTBF} + \text{MTR})$$

3]. Explain the common terms related to BC operation (30)
explain BC terminology (8) explain the following

① disaster recovery

② disaster restart

③ RPO

④ RTO

⇒ common terms related to BC operations are.

① Disaster recovery:-

- * This is the coordinated process of restoring systems, data, and the infrastructure required to support ongoing business operations after a disaster occurs.
- * It is the process of restoring a previous copy of the data and other necessary processes to that copy to bring it to a known point of reliability.
- * After all recovery efforts are completed, the data is validated to ensure that it is correct.

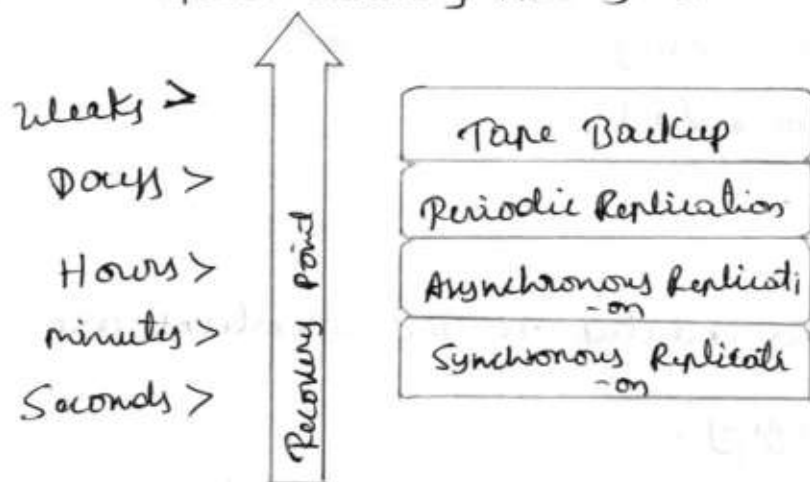
② Disaster restart:-

- * This is the process of restarting business operations with mirrored consistent copies of data and applications.

③ Recovery - Point objective (RPO):-

- * It defines the amount of data loss that a business can tolerate.
- * A large RPO signifies high tolerance to information loss in a business.
- * Based on the RPO, organizations plan for the frequency with which a backup or replica must be made.
- * This is the point in time to which systems and data must be recovered after an outage.

* following figure shows various RPOs and their corresponding ideal recovery strategies.



* RPO of 24 hours; Backups are created at an offsite tape library every midnight. the corresponding recovery strategy is to restore data from the set of last backup tapes.

* RPO of 1 hour; Shipping database logs to the remote site every hour. the corresponding recovery strategy is to restore the database to the point of the last log shipment.

* RPO in the order of minutes; mirroring data asynchronously to a inaccessible site.

* Near Zero RPO; mirroring data synchronously to a inaccessible site.

④ - Recovery-Time Objective (RTO):-

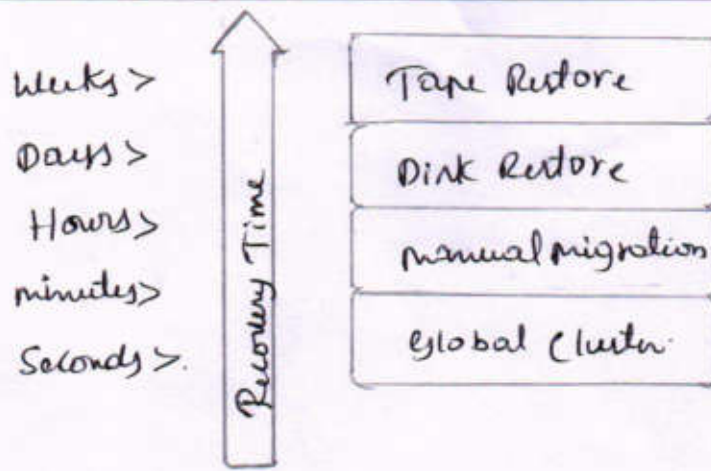
* The time within which systems and applications must be recovered after an outage.

* IT defines the amount of downtime that a business can endure and survive.

* Businesses can optimize disaster recovery plans after defining the RTO for a given system.

* RTOs and the recovery strategies to ensure data availability are listed here.

*



- * RTO of 72 hours: Restore from tapes available at a cold site
- * RTO of 12 hours: Restore from tapes available at a hot site
- * RTO of few hours: use of data vault at a hot site
- * RTO of a few seconds: cluster production servers with bidirectional mirroring, enabling the applications to run at both sites simultaneously.

* Data Vault:

A repository at a remote site where data can be periodically & continuously copied so that there is always a copy at another site.

* hot site:

A site where an enterprise's operations can be moved in the event of disaster. It is a site with the required hardware - are, operating system, application, and network support to perform business operations, where the equipment is available and running at all times.

* Cold site:

A site where an enterprise's operations can be moved in the event of disaster, with minimum IT infrastructure and environment facilities in place, but not activated.

Q4) Explain BC planning lifecycle with a neat diagram?

Ans: * The Conceptualization to the realization of the BC plan, a life cycle of activities can be defined for the BC process.

* The BC planning life cycle includes five stages:-

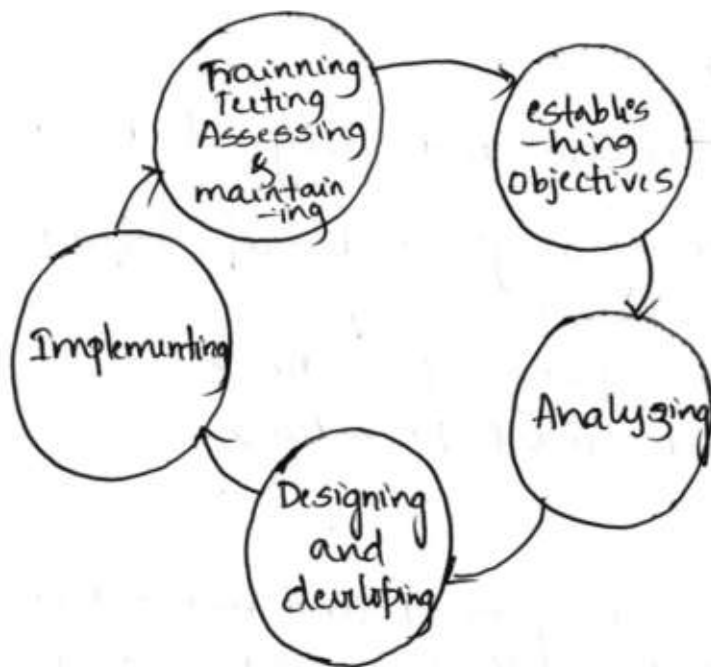
1). Establishing objectives.

2). Analyzing.

3). Designing & developing.

4). Implementing.

5). Training, testing, assessing, and maintaining.



* Several activities are performed at each stage of the BC planning lifecycle, including the following key activities:-

1) Establish objectives:-

* Determine BC requirements.

* Estimate the scope & budget to achieve requirements.

* Select a BC team that includes subject matter experts from all areas of the business, whether internal or external.

* Create BC policies.

2) Analysis:-

- * Collect information on data profiles, business processes, infrastructure support, dependencies, & frequency of using business infrastructure.
- * Conduct a Business Impact Analysis (BIA).
- * Identify critical business processes and assign recovery priorities.
- * Perform risk analysis for critical functions & create mitigation strategies.
- * Perform cost benefit analysis for available solutions based on the mitigation strategy.
- * Evaluate options.

3) Design and develop:

- * define the team structure and assign individual roles and responsibilities.
- * design data protection strategies & develop infrastructure.
- * develop emergency solutions.
- * develop emergency response procedures.
- * detail recovery and restart procedures.

4) Implement:

- * Implement risk management and mitigation procedures that include backup, replication, and management of resources.
- * Prepare the disaster recovery site that can be utilized if a disaster affects the primary data center.
- * Implement redundancy for every resource in a data center to avoid single points of failure.

⇒

5). Train, test, assess, and maintain:

- * Train the employees who are responsible for backup & replication of business-critical data on a regular basis or whenever there is a modification in the BC plan.
- * Train employees on emergency response procedures when disasters are declared.
- * Train the recovery team on recovery procedures based on contingency scenarios.
- * Perform damage-assessment procedures & review recovery plans.
- * Test the BC plan regularly to evaluate its performance & identify its limitations.
- * Assess the performance reports & identify limitations.
- * Update the BC plan & recovery/restart procedures to reflect regular changes within the data center.

- 5) Define Single point of failure and explain the idea to resolving single point of failure with neat diagram.
- ⇒ A single point of failure refers to the failure of a component that can terminate the availability of the entire system @ IT service.

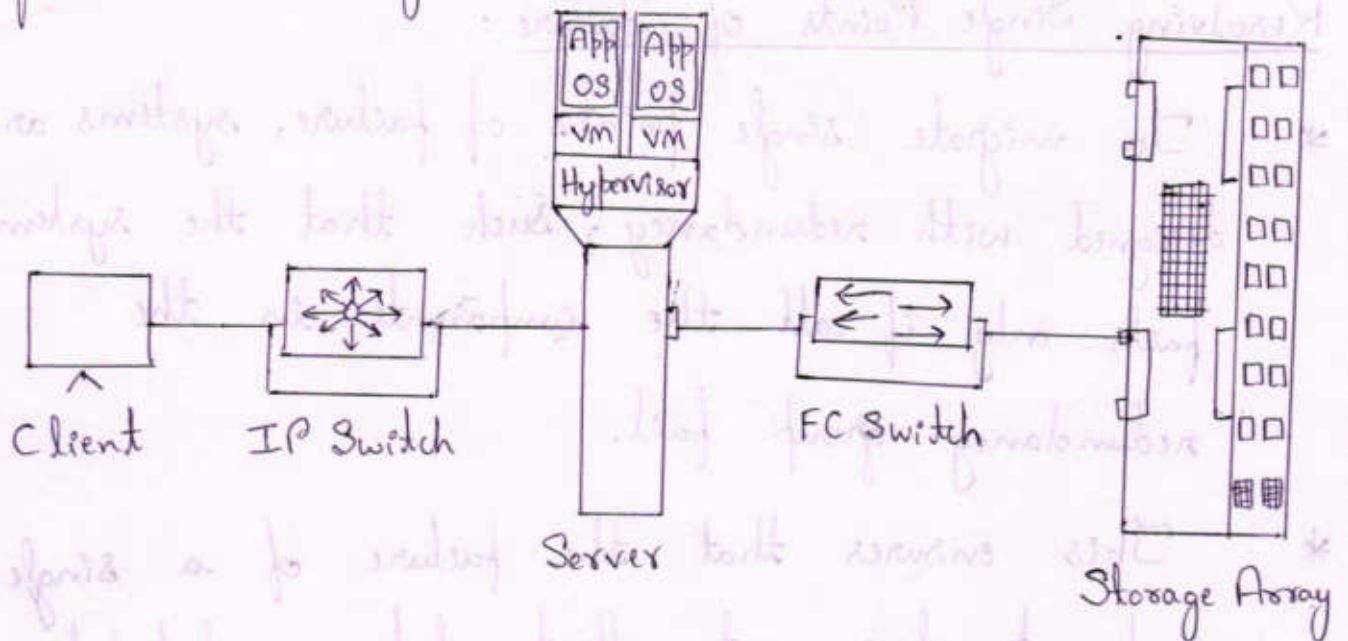


Fig: Single Point of Failure

- * In a setup in which each component must function as required to ensure data availability, the failure of a single physical or virtual component causes the unavailability of an application.
- * This failure results in disruption of business operations.
- * A VM, a hypervisor, an HBA (Host Bus Adapter) on the server, the physical server, the IP network, the FC switch, the storage array ports, or even

or even the array storage array could become potential single points of failure. To avoid single points of failure, it is essential to implement a fault-tolerant mechanism.

Resolving Single Points of Failure:

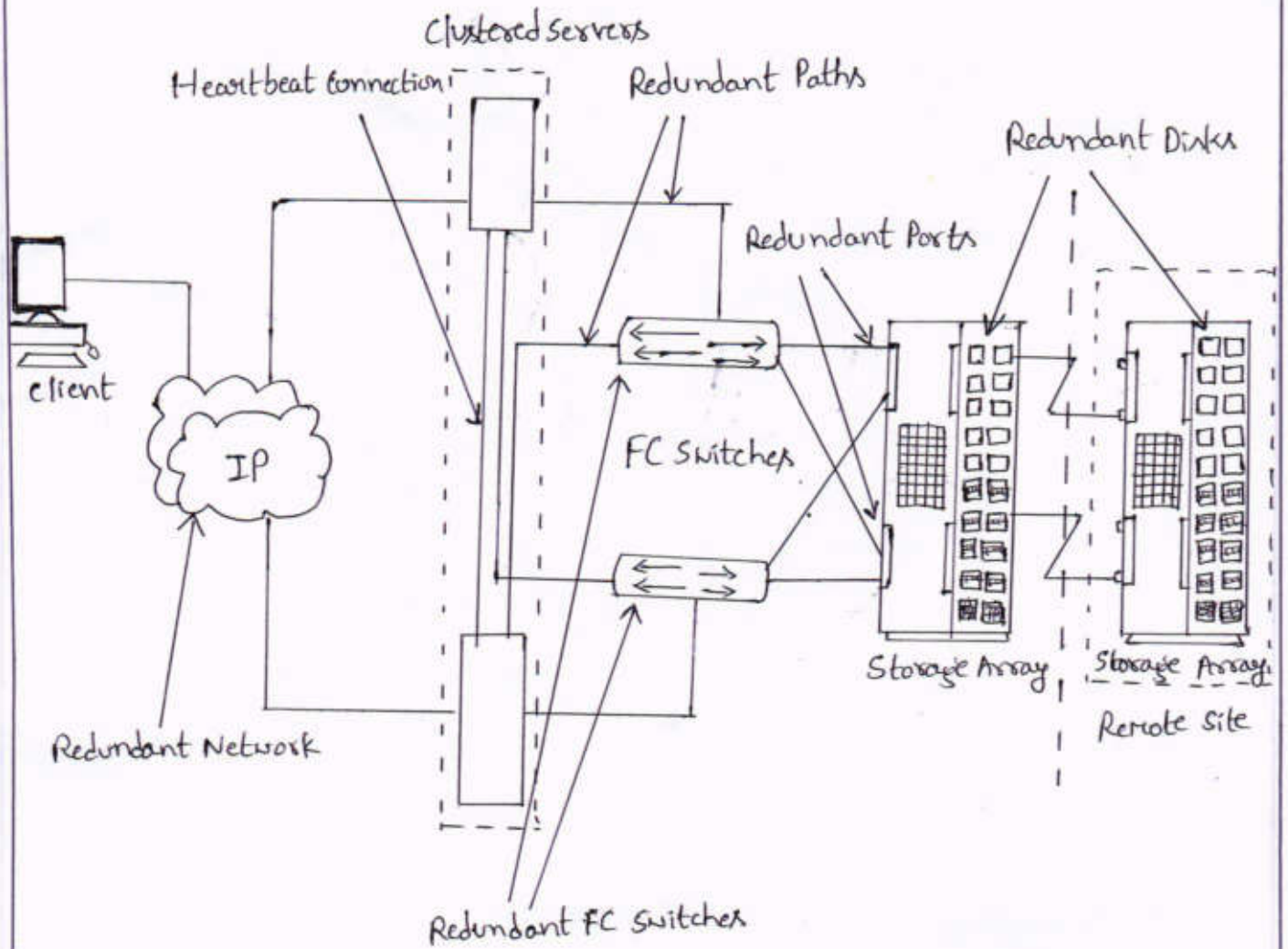
- * To mitigate single points of failure, systems are designed with redundancy, such that the system fails only if all the components in the redundancy group fail.
- * This ensures that the failure of a single component does not affect data availability.
- * Data centers follow stringent guidelines to implement fault tolerance for uninterrupted information availability.
- * Careful analysis is performed to eliminate every single point of failure.
- * Configuration of redundant HBAs at a server to mitigate single HBA failure.

- * Configuration of NIC (Network Interface Card) teaming at a server allows protection against single physical NIC failure. It allows grouping of two or more physical NICs and treating them as a single logical device. With NIC teaming, if one of the underlying physical NICs fails or its cable is unplugged, the traffic is redirected to another physical NIC in the team. Thus, NIC teaming eliminates the single point of failure associated with a single physical NIC.
- * Configuration of redundant switches to account for a switch failure.
- * Configuration of multiple storage array ports to mitigate a port failure.
- * RAID and hot spare configuration to ensure continuous operation in the event of disk failure.
- * Implementation of a redundant storage array at a remote site to mitigate local site failure.

* Implementing server (or compute) clustering, a fault-tolerance mechanism whereby two or more servers in a cluster access the same set of data volumes. Clustered servers if one of the servers or hypervisor fails, the other server or hypervisor can take up the workload.

* Implementing a VM Fault Tolerance mechanism ensures BC in the event of a server failure. This technique creates duplicate copies of each VM on another server so that when a VM failure is detected, the duplicate VM can be used for failover. The two VMs are kept in synchronization with each other in order to perform successful failovers.

Implementation of fault tolerance



SAN

(i) Backup

- A backup is a copy of production data, created and retained for the sole purpose of recovering deleted or corrupted data.

Backup purpose

Backups are performed to serve 3 purposes:

- (i) disaster recovery.
- (ii) operational backup
- (iii) archival

(i) Disaster Recovery.

- Backups can be performed to address disaster recovery needs.
- The backup copies are used for restoring data at an alternate site when the primary site is incapacitated due to a disaster.
- Based on RPO and RTO requirements, organization use different backup strategies for disaster recovery.
- When a tape-based backup method is used as a disaster recovery strategy, the backup tape media is shipped and stored at an offsite location. These tapes can be recalled for restoration at the site.
- Organizations with stringent RPO and RTO requirements use remote replication technology to replicate data to a disaster recovery site.

- This allows organizations to bring up production system online in a relatively short period of time in the event of a disaster.

(ii) Operational Backup.

- Data in the production environment changes with every business transaction and operation.
- Operational backup is a backup of data at a point in time and is used to restore data in the event of data loss or logical corruptions that may occur during routine processing.
- Operational backups are created for the active production information by using incremental or differential backup technologies.
- An example of an operational backup is a backup performed for a production data-base just before a bulk batch update.
- This ensures the availability of a clean copy of the production database if the batch update corrupts the production database.

(iii) Archival

- Backups are also performed to address archival requirements.
- Although CAs has emerged as the primary solution for archive, traditional backups are still used by small and medium enterprises for long-term preservation of transaction records, e-mail messages, and other business records required for regulatory compliance.
- Apart from addressing disaster recovery, archival and operational requirements, backups

serve as a protection against data loss due to physical damage of a storage device, software failures, or virus attacks.

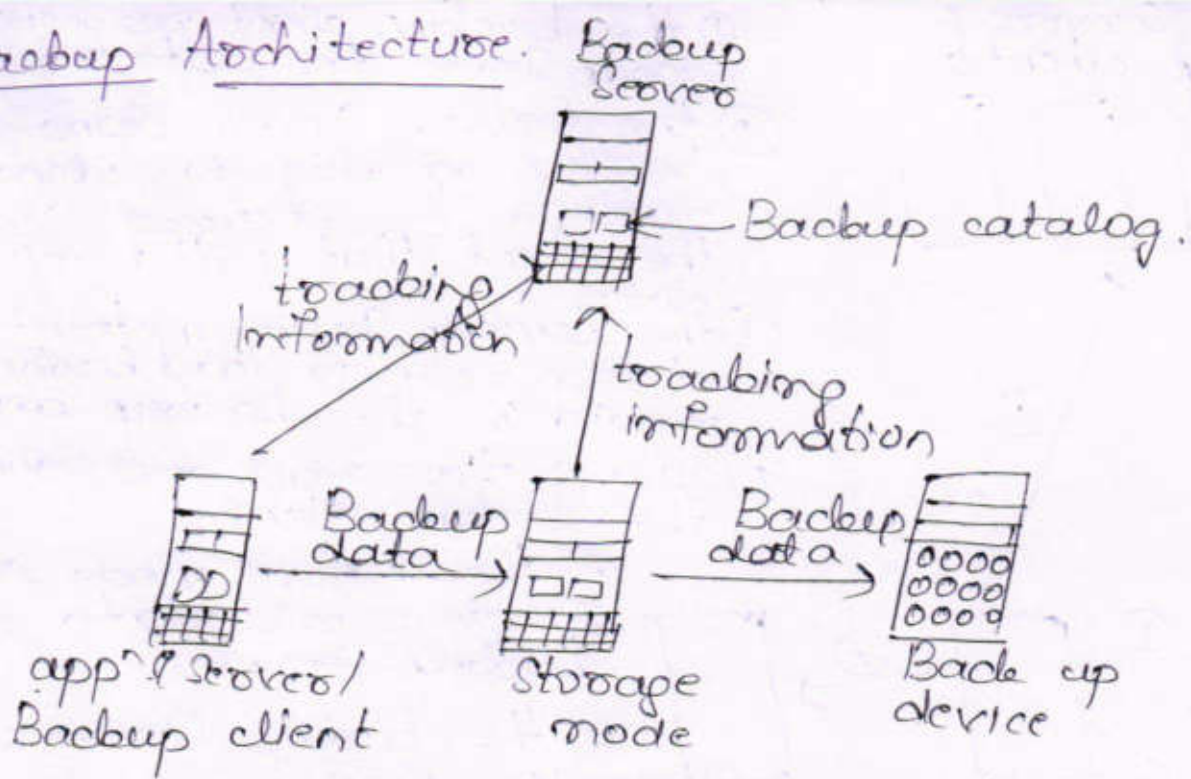
- Backups can also be used to protect against accidents such as a deletion or intentional data destruction.

1. Backup methods.

- Hot Backup and cold backup are the two methods set up for a backup.
- Hot backup, the application is up and running with users accessing their data during the backup process.
- This method of backup is also referred to as an online backup.
- A cold backup requires the appⁿ to be shut down during the backup process.
- Hence, this method is also referred to as an offline backup.
- The HOT backup of online production data is challenging because data is actively used and changed. If a file is open, it is normally not backed up during the backup process.
- In such situations, an open file agent is required to backup the open file.
- These agents interact directly with the OS or appⁿ and enable the creation of consistent copies of open files.
- Disadvantage: Associated with a hot backup is that the agents usually affect the overall application performance.
- Consistent backups of database can also be done by using a cold backup.
- This requires the database to remain inactive during the backup.

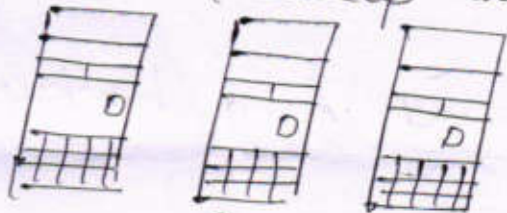
- Disadvantage ∴ cold backup is that the database is inaccessible to users during the backup process.
- A point-in-time (PIT) copy method is deployed in environments in which the impact of downtime from a cold backup or the performance impact resulting from a hot backup is unacceptable.
- The PIT copy is created from the production volume and used as the source for the backup. This reduces the impact on the production volume.
- To ensure consistency, it is not enough to back up only the production data for recovery.
- Certain attributes and properties attached to a file, such as permissions, owner, and other metadata, also need to be backed up.
- These attributes are as important as the data itself and must be backed up for consistency.
- Base-metal recovery (BMR) refers to a backup in which all metadata, system information, and application configurations are appropriately backed up for a full system recovery.
- BMR builds the base system, which includes partitioning, the file system layout, the OS, the app, and the relevant configurations.
- BMR recovers the base system first before starting the recovery of data files.

2. Backup Architecture.



3. Backup and restore operations

app servers/ Backup clients



(1) Backup Servers initiates scheduled backup process

(2) Backup server retrieves backup-related information from backup catalog.

(3a) Backup Servers instructs storage mode to load backup media in backup device.

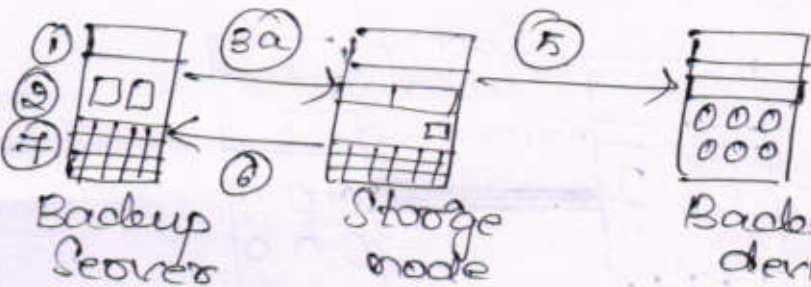
(3b) Backup server instructs backup clients to send data to be backed up to storage mode

(4) Backup clients send data to storage mode and update the backup catalog on the backup server

(5) Storage mode sends data to backup device.

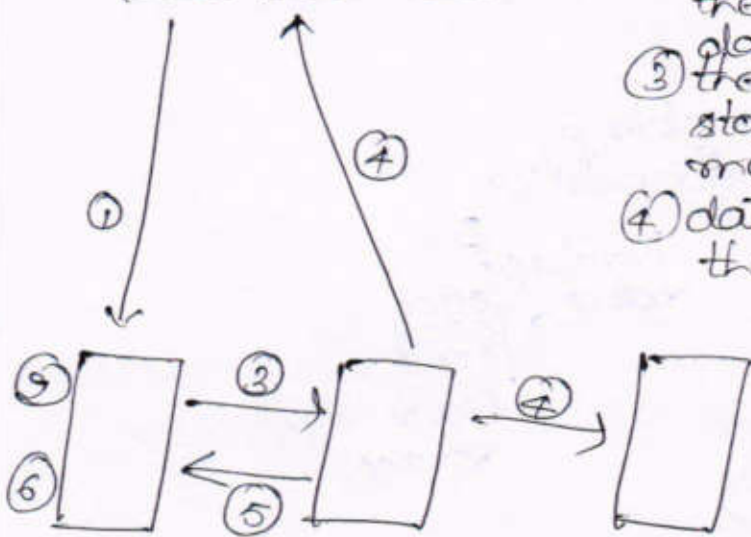
(6) Storage mode sends metadata and media information to backup server

(7) Backup server updates the backup catalog.



app servers / Backup clients

- ① the backup client requests the backup server for data restore.
- ② the backup server scans the backup catalog to identify data to be restored and the client that will receive data.
- ③ the backup server instructs the storage node to load backup media in the backup device.
- ④ data is then read and sent to the backup client.
- ⑤ the storage node sends restore metadata to the backup server.
- ⑥ the backup server updates the backup catalog.



4. Backup Topologies.

- o three basic topologies are used in a backup environment
 1. Direct attached backup
 2. LAN-based backup
 3. SAN-based backup.
 4. Mixed topology is also used by combining LAN-based and SAN-based topologies.

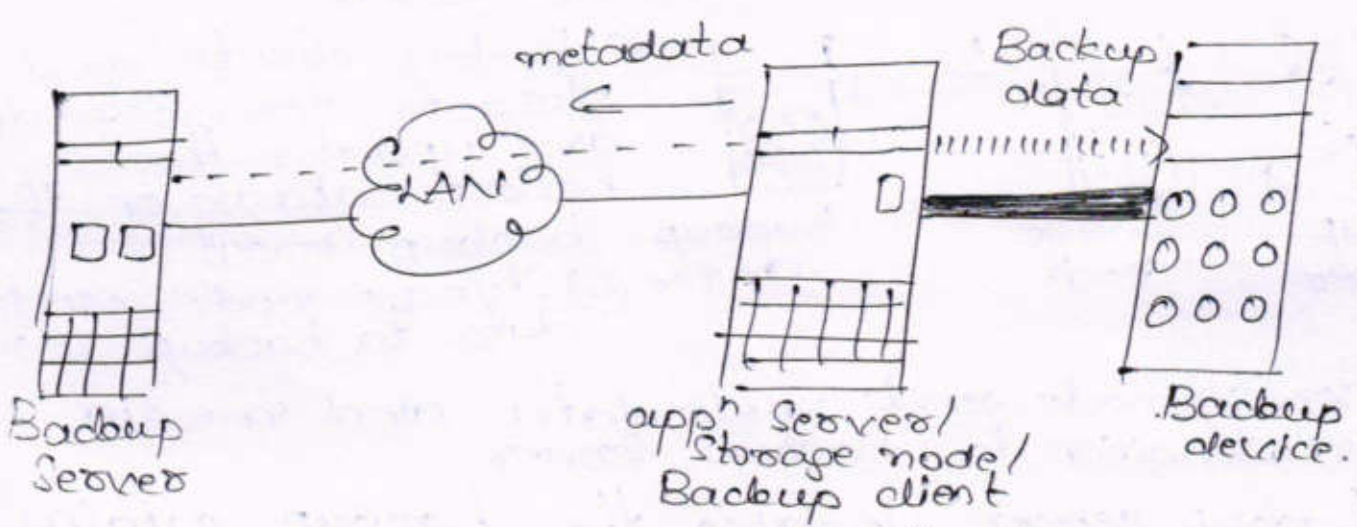


fig:- Direct - attached backup topology.

- As the environment grows, there will be a need for centralized management and sharing of backup devices to optimize costs.
- An appropriate software is required to share the backup devices among multiple servers.

app Servers/Backup client

Backup Servers

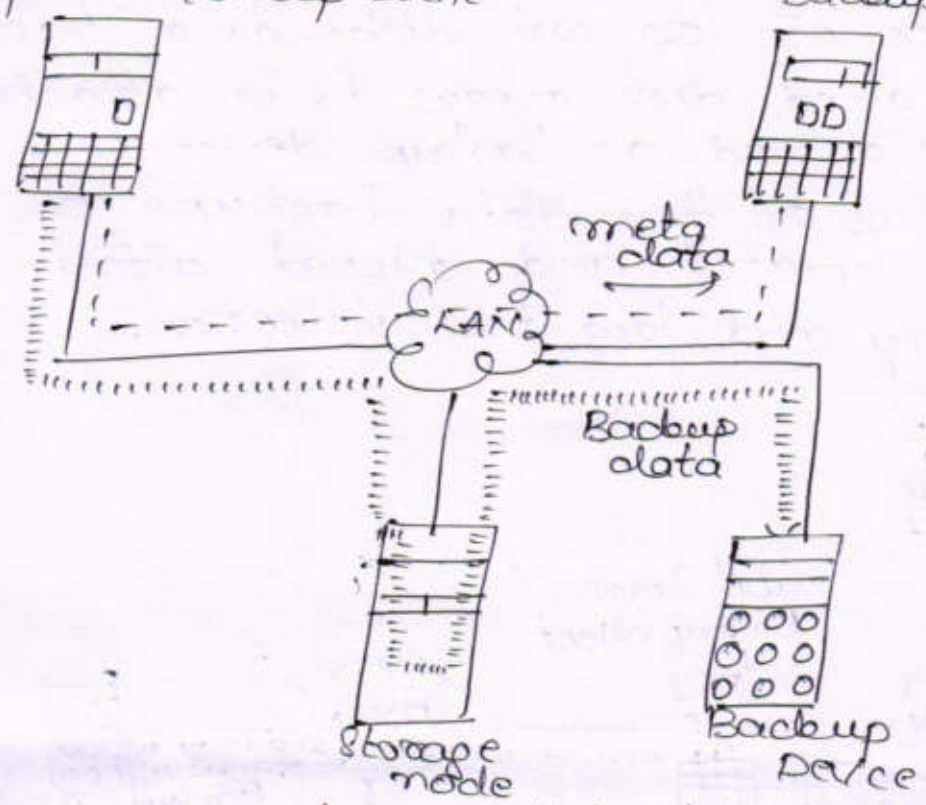


fig:- LAN-based backup technology.

- The data to be backup is transferred from the backup client (source) to the backup device (destination) over the LAN, which might affect network performance.

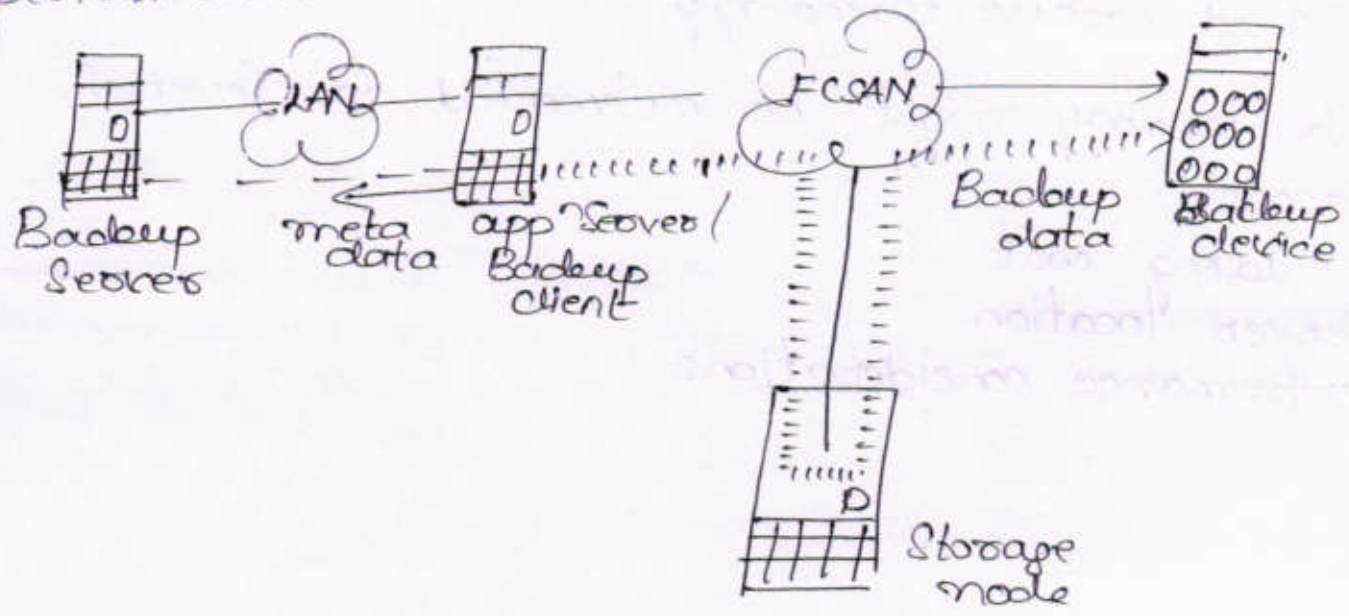


fig:- SAN based backup technology

- o the backup data traffic is restricted to the SAN, and only the backup metadata is transported over the LAN.
- o the volume of metadata is insignificant when compared to the manufacture data; the LAN performance is not degraded in this configuration.
- o The emergence of low cost disks as a backup medium has enabled disk arrays to be attached to the SAN and used as backup devices.
- o A tape backup of these data backups on the disks can be created and shipped offsite for disaster recovery and long term retention.

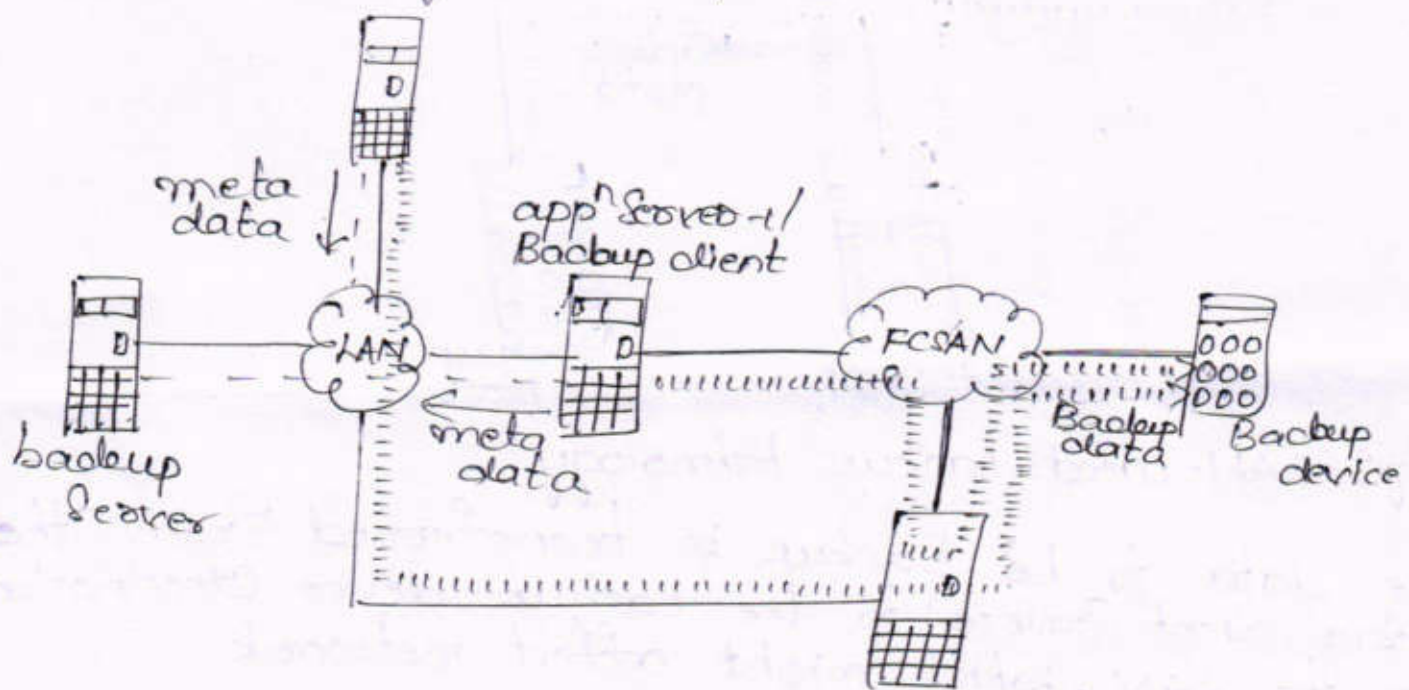


fig:- Mixed backup technology

- o this topology might be implemented for several reasons.
- o Including cost
- o Server location
- o performance considerations.

15. Data deduplication for backup.

- data deduplication is the process of identifying and eliminating redundant data.
- when duplicate data is detected during backup, the data is discarded and only the pointer is created to refer the copy of the data that is already backed up.
- Data deduplication helps to reduce the storage requirement for backup, shorten the backup window, and remove the network burden.
- It also helps to store more backups on the disk and retain the data on the disk for a longer time.

5.1 Data duplication methods:

- There are 2 methods of deduplication:
 - file level
 - subfile level
- File-level deduplication (also called single-instance storage) detects and removes redundant copies of identical files.
- It enables storing only one copy of the file the subsequent copies are replaced with a pointer that points to the original file.
- file level deduplication is simple and fast but does not address the problem of duplicate content inside the files.
- Sub file deduplication, breaks the file into smaller chunks and then uses a specialized algorithm to detect redundant data within and across the file.
- As a result, subfile deduplication eliminates duplicate data across files.

There are two forms of subfile deduplication:

1. Fixed-length block.
2. Variable-length segment.

1. Fixed length block deduplication divides the files into fixed length blocks and uses a hash algorithm to find the duplicate data.

• Although simple in design, fixed length blocks might miss many opportunities to discover redundant data because the block boundary of similar data might be different.

2. In variable length segment deduplication, if there is a change in the segment, the boundary for only that segment is adjusted, leaving the remaining segments unchanged.

• This method vastly improves the ability to find duplicate data segments compared to fixed block.

5.2 data deduplication implementation

deduplication for backup can happen at the data source or the backup target.

• Source-Based data deduplication:

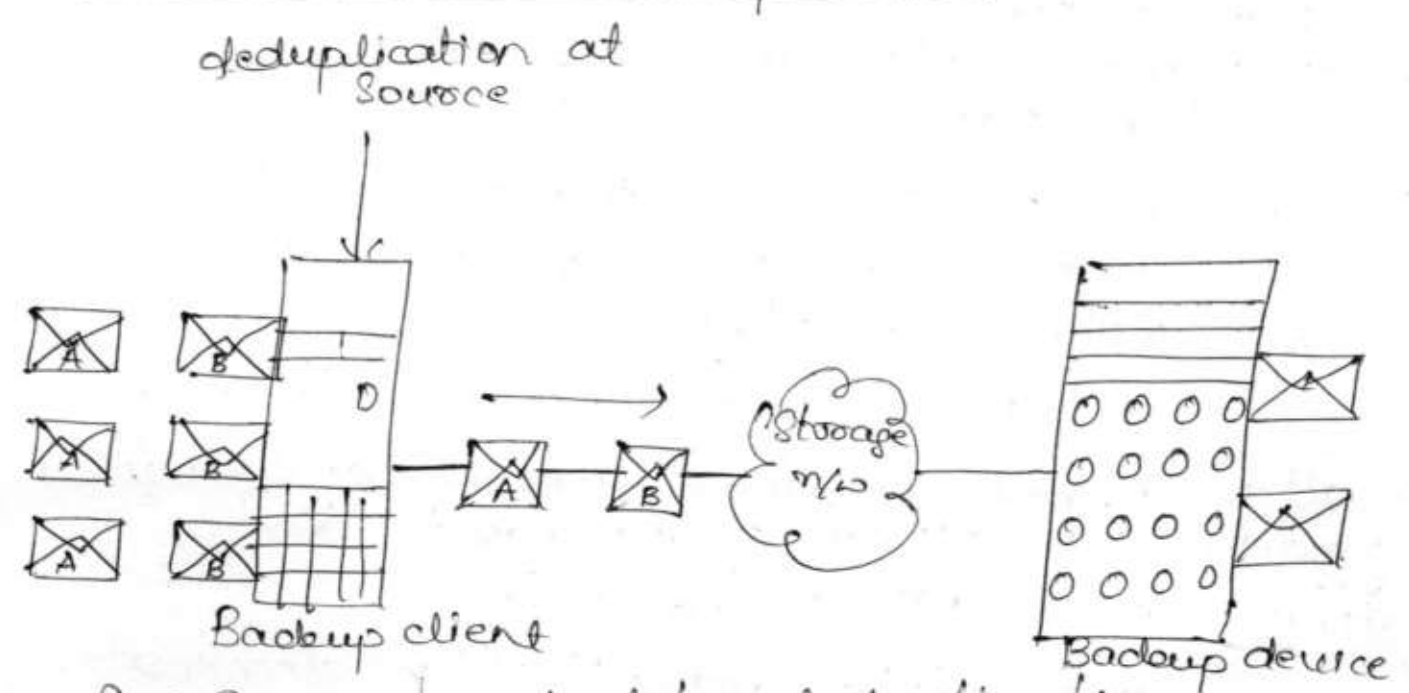


Fig: Source based data deduplication.

6. Target-Based data duplication

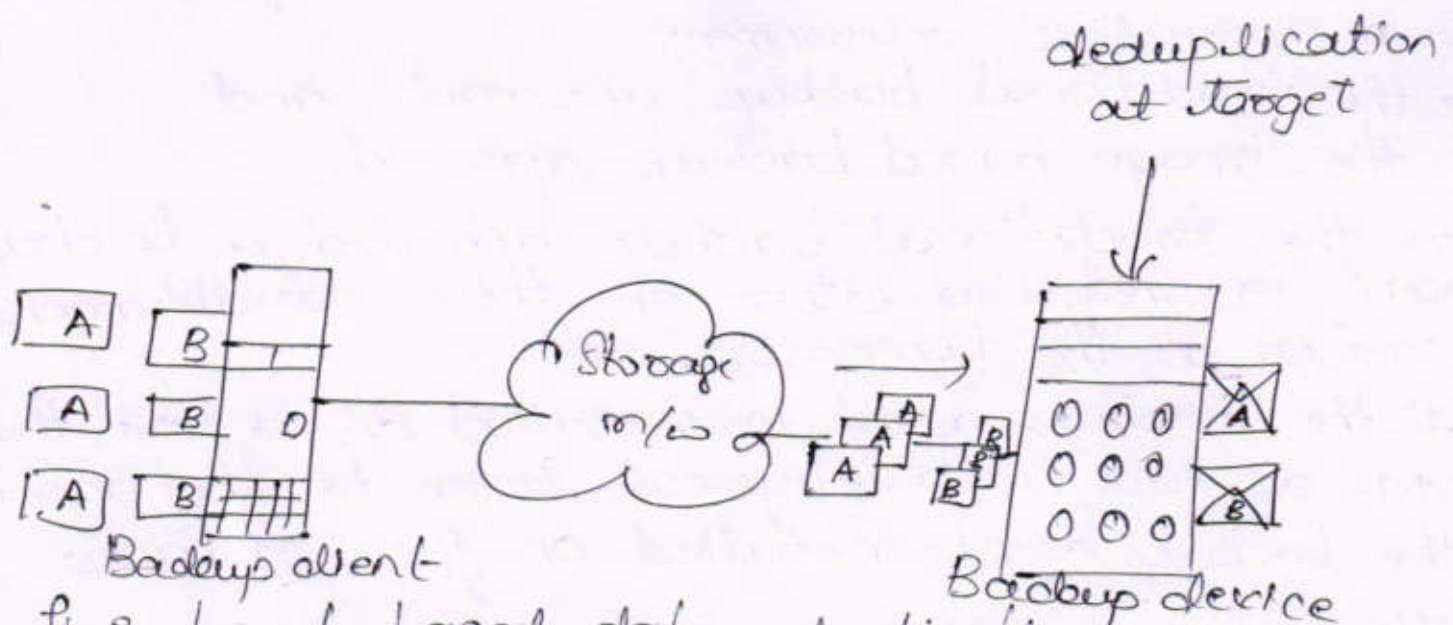
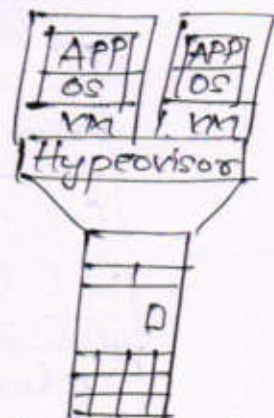


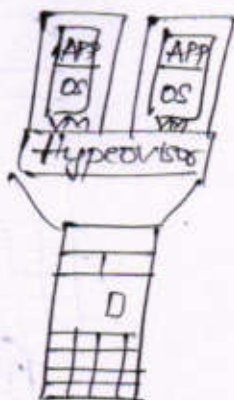
fig:- target-based data duplication

- Inline deduplication performs deduplication on the backup. data before it is stored on the backup device.
- Hence, this method reduces the storage capacity needed for the backup.
- Inline deduplication introduces less time required to identify and remove duplication in the data.
- So, this method is best suited for an environment with a large backup windows.
- Post-process deduplication enables the backup data to be stored or written on the backup device first and then deduplicated later.
- this much is suitable for situations with tighter backup windows.
- However, post process deduplication requires more storage capacity to store the backup images before they are deduplicated.

8. Backup in virtualized environments:
- there are 2 approaches for performing a backup in a virtualized environment:
 - the traditional backup approach and the image-based backup approach.
 - In the traditional backup approach, a backup agent is installed either on the virtual machines (VMs) or on the hypervisor.
 - If the backup agent is installed on a VM, the VM appears as a physical server to the agent. The backup agent installed on the VM backs up devices.
 - The agent does not capture VM files such as the virtual BIOS file, VM swap file logs, and configuration files.
 - Therefore, for a VM restore, a user needs to manually recreate the VM and then restore data onto it.
 - If the backup agent is installed on the hypervisor, the VM appears as a set of files to the agent. So, VM files can be backed up by performing a file system backup from a hypervisor.
 - Disadvantage:
 - the traditional backup method can cause high CPU utilization on the server being backed up.
 - the backup should be performed when the server resources are idle or during a low activity period on the network.



Backup agent runs on each VM



Backup agent runs on Hypervisor

A - Backup agent

Fig: Traditional VM backup

- Image-based backup operates at the hypervisor level and essentially takes a snapshot of the VM.
- It creates a copy of the guest OS and all the data associated with it (snapshot of VM disk files), including the VM state and application configurations.
- The backup is saved as a single file called an "image", and this image is mounted on the separate physical machine-proxy server, which acts as a backup client.
- This effectively loads the backup processing from the hypervisor and transfers the load on the proxy server, thereby reducing the impact to VMs running on the hypervisor.
- Image based backup enables quick restoration of a VM.

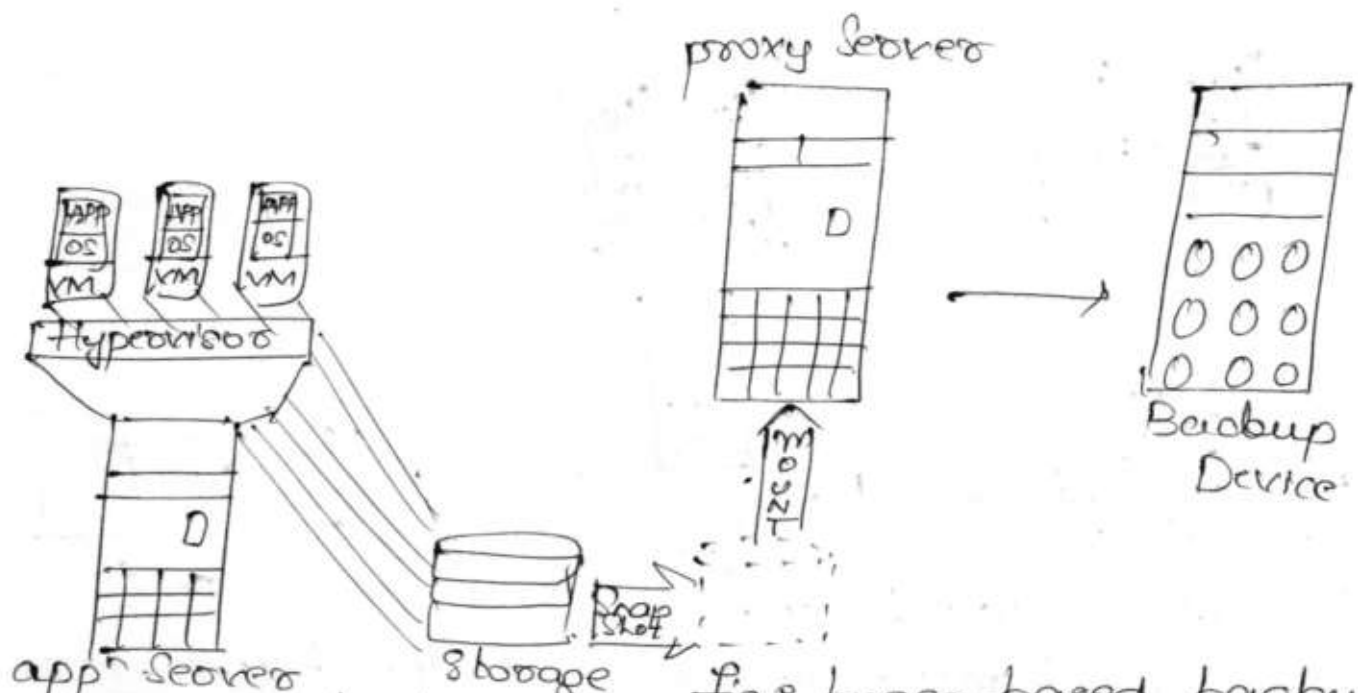
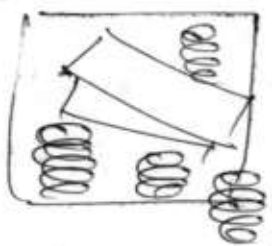


fig:- Image-based backup

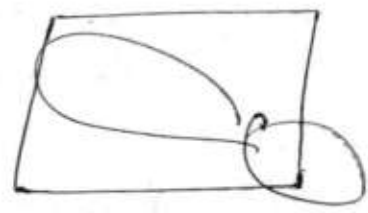
7. Data Archiving

- In the life cycle of information, data is actively created, accessed and changed.
- As data ages, it is less likely to be changed and eventually becomes "fixed" but continues to be accessed by applications and users.
- this data is called fixed content.

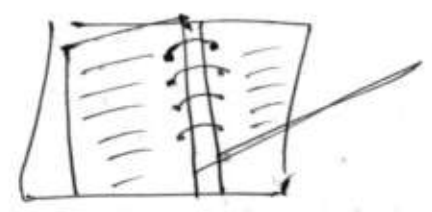
Generate New Revenues



Improve Service levels



Leverage Historical value



Digital assets retained for active reference and value

Electronic documents

Digital records

- Contracts & claims
- E-mail attachments
- Financial spreadsheets
- CAD/CAM designs
- Presentations
- Documents
 - checks & security trade
 - Historical preservation
- Photographs
 - Personal / Professional

Examples of fixed content data

- An archive can be implemented as an online, nearline, or offline solution.
- Online archive: A storage device directly connected to a host that makes the data immediately accessible.
- Nearline archive: A storage device connected to a host, but the device where the data is stored must be mounted or loaded to access the data.
- Offline archive: A storage device that is not ready to use. Manual intervention is required to connect, mount, or load the storage device before data can be accessed.
- Traditionally, optical and tape media were used in archives.
- Optical media are typically write once read many (WORM) devices that protect the original file from being overwritten.
- Some tape devices also provide this functionality by implementing file-locking capabilities.
- Although these devices are inexpensive, they involve operational, management and maintenance overhead.
- Tapes and optical media are also at risk to copy and destroy.
- These requirements have exposed the shortcomings of the traditional tape and optical media archive solutions.
- Content addressed storage (CAS) is disk-based storage that has emerged as an alternative to tape and optical solutions.
- CAS meets the demand to improve data accessibility and to protect, dispose of, and ensure service-level agreements (SLAs) for archive data.

Module – 5:

Securing and Managing Storage Infrastructure

- Domains of storage security along with covering security
- Security threats, and countermeasures in various domains Security solutions for FC-SAN
- IP-SAN and NAS environments, Security in virtualized
- Virtual environments, Information lifecycle management (ILM)
- Storage tiering
- Cloud service management activities

Module – 5:

Securing and Managing Storage Infrastructure

1 Information Security Framework

The basic information security framework is built to achieve four security goals: confidentiality, integrity, and availability (CIA), along with accountability. This framework incorporates all security standards, procedures, and controls, required to mitigate threats in the storage infrastructure environment.

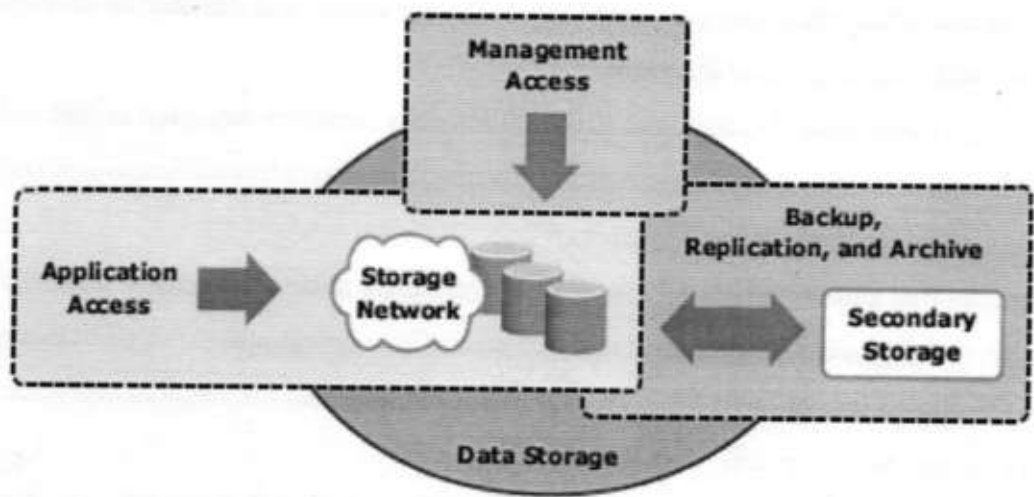
- **Confidentiality:** Provides the required secrecy of information and ensures that only authorized users have access to data. This requires authentication of users who need to access information. Data in transit (data transmitted over cables) and data at rest (data residing on a primary storage, backup media, or in the archives) can be encrypted to maintain its confidentiality. In addition to restricting unauthorized users from accessing information, confidentiality also requires implementing traffic flow protection measures as part of the security protocol. These protection measures generally include hiding source and destination addresses, frequency of data being sent, and amount of data sent.
- **Integrity:** Ensures that the information is unaltered. Ensuring integrity requires detection of and protection against unauthorized alteration or deletion of information. Ensuring integrity stipulates measures such as error detection and correction for both data and systems.
- **Availability:** This ensures that authorized users have reliable and timely access to systems, data, and applications residing on these systems. Availability requires protection against unauthorized deletion of data and denial of service (discussed in section “14.2.2 Threats”). Availability also implies that sufficient resources are available to provide a service.
- **Accountability service:** Refers to accounting for all the events and operations that take place in the data center infrastructure. The accountability service maintains a log of events that can be audited or traced later for the purpose of security.

2 Storage Security Domains

Storage devices connected to a network raise the risk level and are more exposed to security threats via networks. However, with increasing use of networking in storage environments, storage devices are becoming highly exposed to security threats from a variety of sources. Specific controls must be implemented to secure a storage networking environment. This requires a closer look at storage networking security and a clear understanding of the access path leading to storage resources. If a particular path is unauthorized and needs to be prohibited by technical controls:

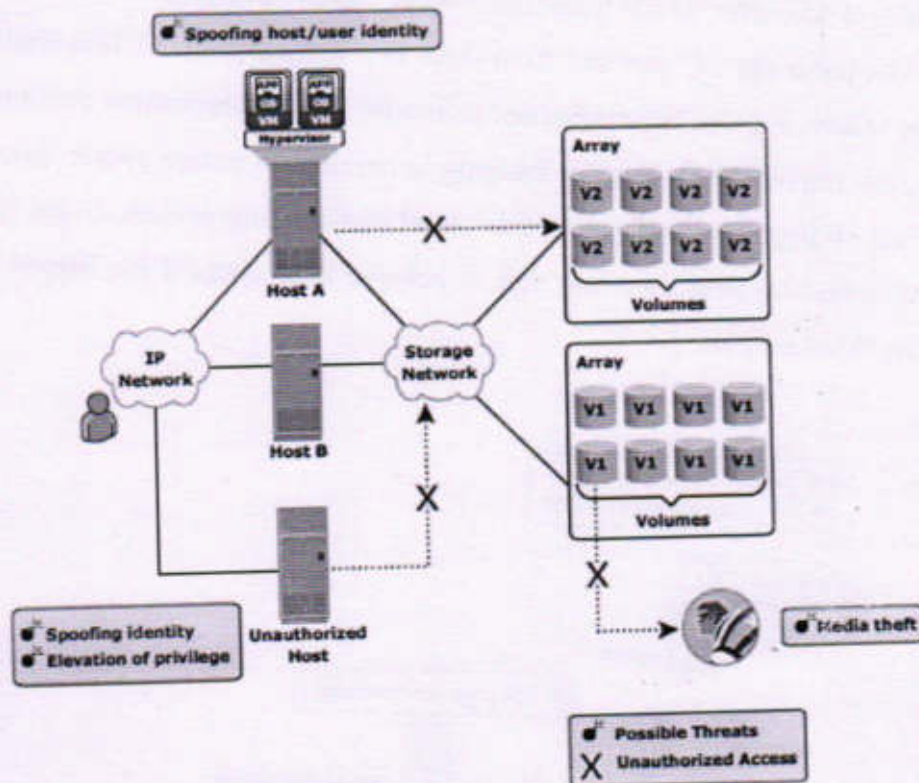
ensure that these controls are not compromised. If each component within the storage network is considered a potential access point, the attack surface of all these access points must be analyzed to identify the associated vulnerabilities. To identify the threats that apply to a storage network, access paths to data storage can be categorized into three security domains: *application access*, *management access*, and *backup, replication, and archive*. Figure 14-1 depicts the three security domains of a storage system environment. The first security domain involves application access to the stored data through the storage network. The second security domain includes management access to storage and interconnect devices and to the data residing on those devices.

This domain is primarily accessed by storage administrators who configure and manage the environment. The third domain consists of backup, replication, and archive access. Along with the access points in this domain, the backup media also needs to be secured. To secure the storage networking environment, identify the existing threats within each of the security domains and classify the threats based on the type of security services — availability, confidentiality, integrity, and accountability. The next step is to select and implement various controls as countermeasures to the threats



The *application access domain* may include only those applications that access the data through the file system or a database interface. An important step to secure the application access domain is to identify the threats in the environment and appropriate controls that should be applied. Implementing physical security is also an important consideration to prevent media theft. Figure 14-2 shows application access in a storage networking environment. Host A can access all V1 volumes; host B can access all V2 volumes. These volumes are classified according to the access level, such as confidential, restricted, and public. Some of the possible threats in this scenario could be host A spoofing the identity or elevating to the privileges of host B to gain access to host B's resources. Another threat could be that an unauthorized host gains access to the network; the attacker on this host may try to spoof the identity of another host and tamper with the data, snoop the network, or execute a DoS attack. Also any form of media theft could also

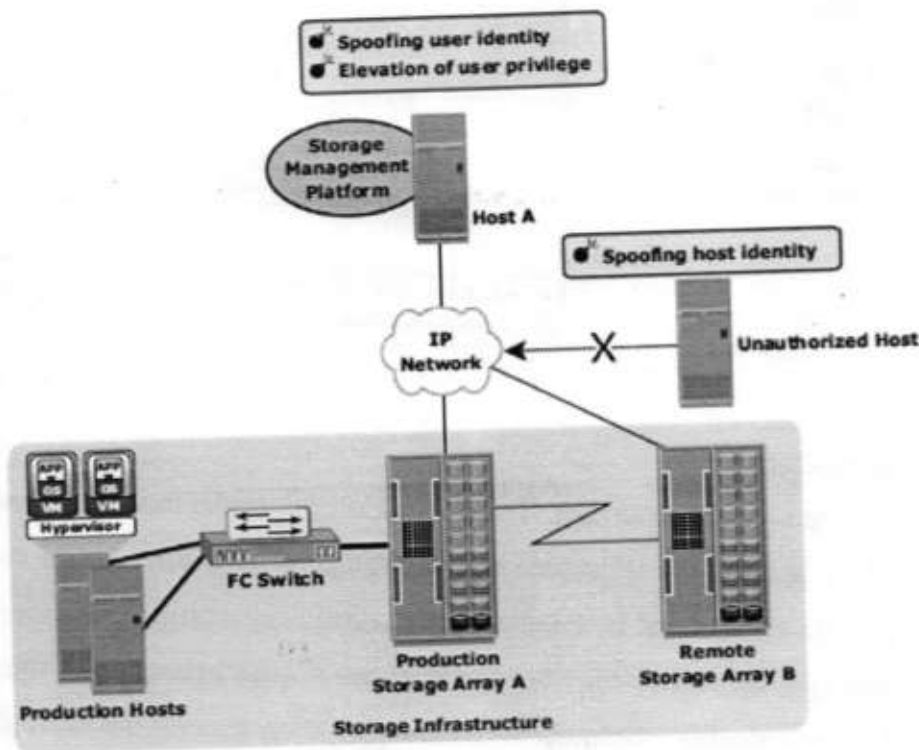
compromise security. These threats can pose several serious challenges to the network security; therefore, they need to be addressed.



Controlling User Access to Data

Access control services regulate user access to data. These services mitigate the threats of spoofing host identity and elevating host privileges. Both these threats affect data integrity and confidentiality. Access control mechanisms used in the application access domain are user and host authentication (technical control) and authorization (administrative control). These mechanisms may lie outside the boundaries of the storage network and require various systems to interconnect with other enterprise identity management and authentication systems, for example, systems that provide strong authentication and authorization to secure user identities against spoofing. NAS devices support the creation of *access control lists* that regulate user access to specific files. The Enterprise Content Management application enforces access to data by using Information Rights Management (IRM) that specifies which users have what rights to a document. Restricting access at the host level starts with authenticating a node when it tries to connect to a network. Different storage networking technologies, such as iSCSI, FC, and IP-based storage, use various authentication mechanisms, such as Challenge-Handshake Authentication Protocol (CHAP), Fibre Channel Security Protocol (FC-SP), and IPSec, respectively, to authenticate host access. After a host has been authenticated, the next

step is to specify security controls for the storage resources, such as ports, volumes, or storage pools, that the host is authorized to access. *Zoning* is a control mechanism on the switches that segments the network into specific paths to be used for data traffic; *LUN masking* determines which hosts can access which storage devices. Some devices support mapping of a host's WWN to a particular FC port and from there to a particular LUN. This binding of the WWN to a physical port is the most secure. Finally, it is important to ensure that administrative controls, such as need security policies and standards, are implemented. Regular auditing is required to ensure proper functioning of administrative controls. This is enabled by logging significant events on all participating devices. Event logs should also be protected from unauthorized access because they may fail to achieve their goals if the logged content is exposed to unauthorized modifications by an attacker



Security Implementations in Storage Networking

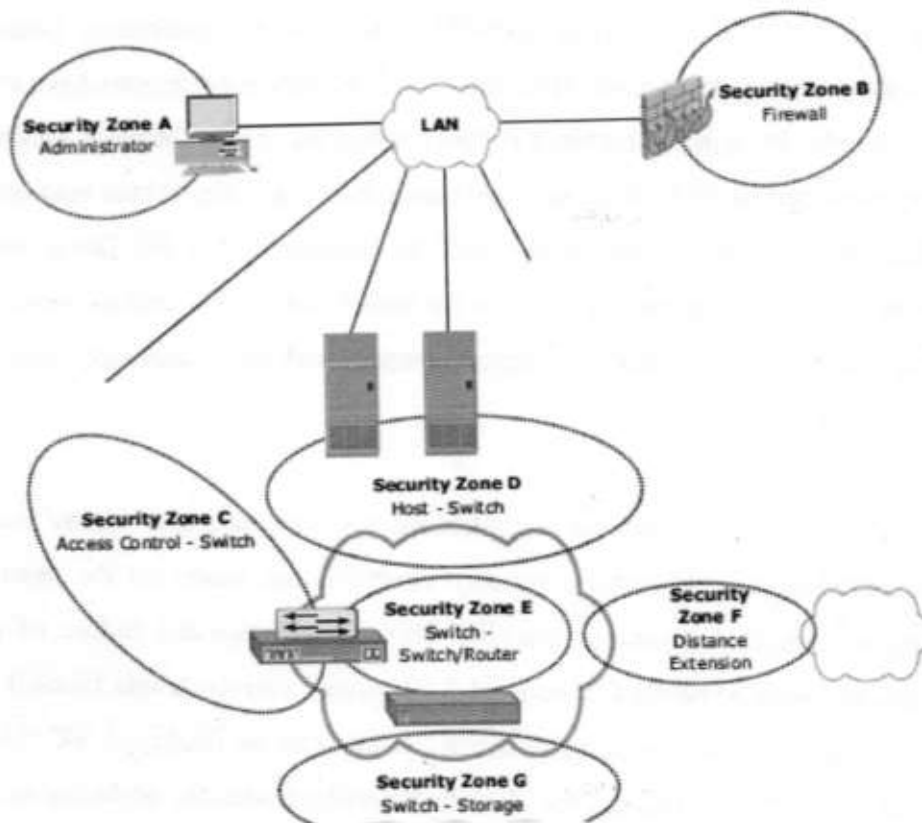
FC SAN

Traditional FC SANs enjoy an inherent security advantage over IP-based networks. An FC SAN is configured as an isolated private environment with fewer nodes than an IP network. Consequently, FC SANs impose fewer security threats. However, this scenario has changed with converged networks and storage consolidation, driving rapid growth

and necessitating designs for large, complex SANs that span multiple sites across the enterprise. Today, no single comprehensive security solution is available for FC SANs. Many FC SAN security mechanisms have evolved from their counterpart in IP networking, thereby bringing in matured security solutions. *Fibre Channel Security Protocol* (FC-SP) standards (T11 standards), published in 2006, align security mechanisms and algorithms between IP and FC interconnects. These standards describe protocols to implement security measures in a FC fabric, among fabric elements and N_Ports within the fabric. They also include guidelines for authenticating FC entities, setting up session keys, negotiating the parameters required to ensure frame-by-frame integrity and confidentiality, and establishing and distributing policies across an FC fabric.

FC SAN Security Architecture

Storage networking environments are a potential target for unauthorized access, theft, and misuse because of the vastness and complexity of these environments. Therefore, security strategies are based on the *defense in depth* concept, which recommends multiple integrated layers of security. This ensures that the failure of one security control will not compromise the assets under protection. Figure 14-5 illustrates various levels (zones) of a storage networking environment that must be secured and the security measures that can be deployed. FC SANs not only suffer from certain risks and vulnerabilities that are unique, but also share common security problems associated with physical security and remote administrative access. In addition to implementing SAN-specific security measures organizations must simultaneously leverage other security implementations in the enterprise. Table 14-1 provides a comprehensive list of protection strategies that must be implemented in various security zones. Some of the security mechanisms listed in Table 14-1 are not specific to SAN but are commonly used data center techniques. For example, two-factor authentication is implemented widely; in a simple implementation it requires the use of a username/password and an additional security component such as a smart card for authentication.



Basic SAN Security Mechanisms

LUN masking and zoning, switch-wide and fabric-wide access control, RBAC, and logical partitioning of a fabric (Virtual SAN) are the most commonly used SAN security methods.

SECURITY ZONES	PROTECTION STRATEGIES
Zone D (Host to switch)	Restrict Fabric access to legitimate hosts by (a) implementing ACLs: Known HBAs can connect on specific switch ports only; and (b) implementing a secure zoning method, such as port zoning (also known as hard zoning).
Zone E (Switch to Switch/Switch to Router)	Protect traffic on fabric by (a) using E_Port authentication; (b) encrypting the traffic in transit; and (c) implementing FC switch controls and port controls.
Zone F (Distance Extension)	Implement encryption for in-flight data (a) FC-SP for long-distance FC extension; and (b) IPSec for SAN extension via FCIP.
Zone G (Switch to Storage)	Protect the storage arrays on your SAN via (a) WWPN-based LUN masking; and (b) S_ID locking: masking based on source FC address.

LUN Masking and Zoning

LUN masking and zoning are the basic SAN security mechanisms used to protect against unauthorized access to storage. LUN masking and zoning are detailed in Chapter 4 and Chapter 5, respectively. The standard implementations of LUN masking on storage arrays mask the LUNs presented to a frontend storage port based on the WWPNs of the source HBAs. A stronger variant of LUN masking may sometimes be offered whereby masking can be done on the basis of source FC addresses. It offers a mechanism to lock down the FC address of a given node port to its WWN. *WWPN zoning* is the preferred choice in security-conscious environments.

Securing Switch Ports

Apart from zoning and LUN masking, additional security mechanisms, such as port binding, port lockdown, port lockout, and persistent port disable, can be implemented on switch ports. *Port binding* limits the number of devices that can attach to a particular switch port and allows only the corresponding switch port to connect to a node for fabric access. Port binding mitigates but does not eliminate WWPN spoofing. *Port lockdown* and *port lockout* restrict a switch port's type of initialization. Typical variants of port lockout ensure that the switch port cannot function as an E_Port and cannot be used to create an ISL, such as a rogue switch. Some variants ensure that the port role is restricted to only FL_Port, F_Port, E_Port, or a combination of these. *Persistent port disable* prevents a switch port from being enabled even after a switch reboot. ***Switch-Wide and Fabric-Wide Access Control*** As organizations grow their SANs locally or over longer distances, there is a greater need to effectively manage SAN security. Network security can be configured on the FC switch by using *access control lists (ACLs)* and on the fabric by using fabric binding. ACLs incorporate the device connection control and switch connection control policies. The device connection control policy specifies which HBAs and storage ports can be a part of the fabric, preventing unauthorized devices from accessing it. Similarly, the switch connection control policy specifies which switches are allowed to be part of the fabric, preventing unauthorized switches from joining it. *Fabric binding* prevents an unauthorized switch from joining any existing switch in the fabric. It ensures that authorized membership data exists on every switch and any attempt to connect any switch in the fabric by using an ISL causes the fabric to segment. Role-based access control provides additional security to a SAN by preventing unauthorized activity on the fabric for management operations. It enables the security administrator to assign roles to users that explicitly specify privileges or access rights after logging into the fabric. For example, the *zone admin* role can modify the zones on the fabric whereas a basic user may view only fabric-related information, such as port types and logged-in nodes.

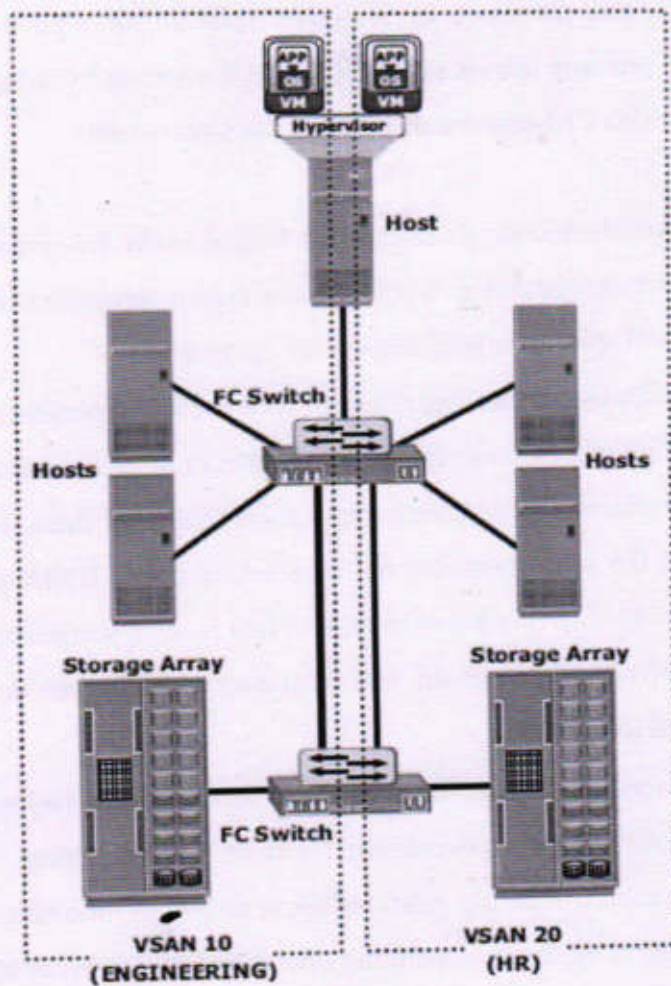
Logical Partitioning of a Fabric: Virtual SAN

VSANs enable the creation of multiple logical SANs over a common physical SAN. They provide the capability to build larger consolidated fabrics and still maintain the required security and isolation between them depicts logical

partitioning in a VSAN. The SAN administrator can create distinct VSANs by populating each of them with switch ports. In the example, the switch ports are distributed over two VSANs: 10 and 20 — for the Engineering and HR divisions, respectively. Although they share physical switching gear with other divisions, they can be managed individually as standalone fabrics. Zoning should be done for each VSAN to secure the entire physical SAN. Each managed VSAN can have only one active zone set at a time. VSANs minimize the impact of fabricwide disruptive events because management and control traffic on the SAN — which may include RSCNs, zone set activation events, and more — does not traverse VSAN boundaries. Therefore, VSANs are a cost-effective alternative for building isolated physical fabrics. They contribute to information availability and security by isolating fabric events and providing authorization control within a single fabric.

NAS

NAS is open to multiple exploits, including viruses, worms, unauthorized access, snooping, and data tampering. Various security mechanisms are implemented in NAS to secure data and the storage networking infrastructure. Permissions and ACLs form the first level of protection to NAS resources by restricting accessibility and sharing. These permissions are deployed over and above the default behaviors and attributes associated with files and folders. In addition, various other authentication and authorization mechanisms, such as Kerberos and directory services, are implemented to verify the identity of network users and define their privileges. Similarly, firewalls protect the storage infrastructure from unauthorized access and malicious attacks.



NAS File Sharing: Windows ACLs

Windows supports two types of ACLs: *discretionary access control lists* (DACLS) and *system access control lists* (SACLs). The DACL, commonly referred to as the ACL, that determines access control. The SACL determines what accesses need to be audited if auditing is enabled. In addition to these ACLs, Windows also supports the concept of object ownership. The owner of an object has hard-coded rights to that object, and these rights do not need to be explicitly granted in the SACL. The owner, SACL, and DACL are all statically held as attributes of each object. Windows also offers the functionality to inherit permissions, which allows the child objects existing within a parent object to automatically inherit the ACLs of the parent object.

ACLs are also applied to directory objects known as security identifiers (SIDs). These are automatically generated by a Windows server or domain when a user or group is created, and they are abstracted from the user. In this way, though a user may identify his login ID as "User1," it is simply a textual representation of the true SID, which is used

ing system. Internal processes in Windows refer to an account's SID rather than the group name while granting access to an object. ACLs are set by using the standard Windows

Example: can also be configured with CLI commands or other third-party tools.

NAS File Sharing: UNIX Permissions

For the UNIX operating system, a *user* is an abstraction that denotes a logical entity for assignment of ownership and operation privileges for the system. A user can be either a person or a system operation. A UNIX system is only aware of the privileges of the user to perform specific operations on the system and

identifies each user by a user ID (UID) and a username, regardless of whether it is a person, a system operation, or a device. In UNIX, users can be organized into one or more groups. The concept of group serves the purpose to assign sets of privileges for a given resource and sharing them among many users that need them. For example, a group of people working on one project may need the same permissions for a set of files. UNIX permissions specify the operations that can be performed by any ownership relation with respect to a file. In simpler terms, these permissions specify what the owner can do, what the owner group can do, and what everyone else can do with the file. For any given ownership relation, three bits are used to specify

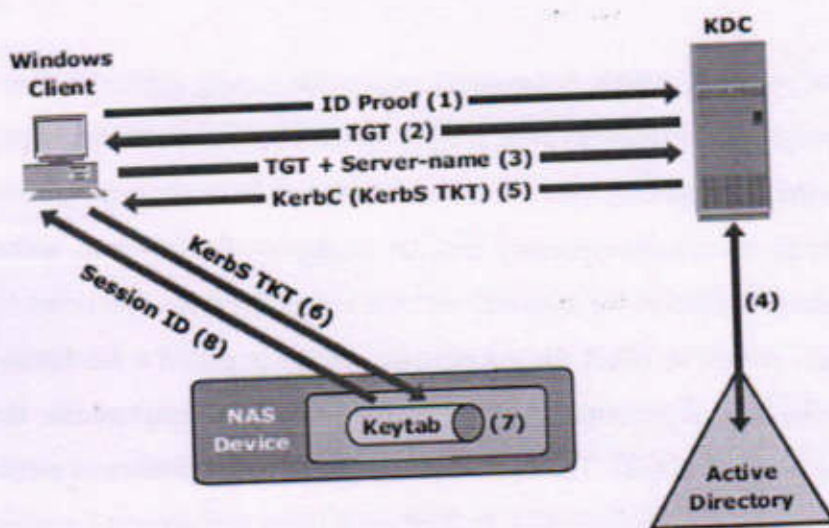
access permissions. The first bit denotes read (r) access, the second bit denotes write (w) access, and the third bit denotes execute (x) access. Because UNIX defines three ownership relations (Owner, Group, and All), a triplet (defining the access permission) is required for each ownership relationship, resulting in nine bits. Each bit can be either set or clear. When displayed, a set bit is marked by its corresponding operation letter (r, w, or x), a clear bit is denoted by a dash (-), and all are put in a row, such as rwxr-xr-x. In this example, the owner can do anything with the file, but group owners and the rest of the world can read or execute only. When displayed, a character denoting the mode of the file may precede this nine-bit pattern. For example, if the file is a directory, it is denoted as "d"; and if it is a link, it is denoted as "l."

NAS File Sharing: Authentication and Authorization

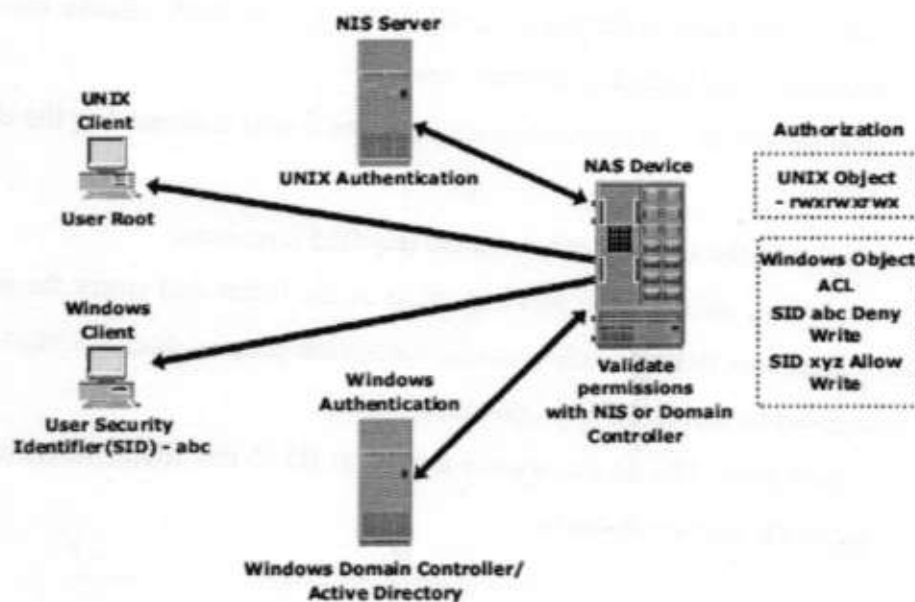
In a file-sharing environment, NAS devices use standard file-sharing protocols, NFS and CIFS. Therefore authentication and authorization are implemented and supported on NAS devices in the same way as in a UNIX or Windows file-sharing environment. Authentication requires verifying the identity of a network user and therefore involves a login credential lookup on a Network Information System (NIS) server in a UNIX environment. Similarly, a Windows client is authenticated by a Windows domain controller that houses the Active Directory. The Active Directory uses LDAP to access information about network objects in the directory and Kerberos for network security. NAS devices use the same authentication techniques to validate network user credentials. Figure 14-7 depicts the authentication process in a NAS environment. Authorization defines user privileges in a network. The authorization

stored in the KDC database are known as *principals*. In a NAS environment, Kerberos is primarily used when authenticating against a Microsoft Active Directory domain, although it can be used to execute security functions in UNIX environments. The Kerberos authentication process shown in Figure 14-8 includes the following steps:

1. The user logs on to the workstation in the Active Directory domain (or forest) using an ID and a password. The client computer sends a request to the AS running on the KDC for a Kerberos ticket. The KDC verifies the user's login information from Active Directory. (This step is not explicitly shown in Figure 14-8.)
2. The KDC responds with an encrypted Ticket Granting Ticket (TGT) and an encrypted session key. TGT has limited validity period. TGT can be decrypted only by the KDC, and the client can decrypt only the session key.
3. When the client requests a service from a server, it sends a request, consisting of the previously generated TGT encrypted with the session key and the resource information to the KDC. 4. The KDC checks the permissions in Active Directory and ensures that the user is authorized to use that service.
5. The KDC returns a service ticket to the client. This service ticket contains fields addressed to the client and to the server hosting the service.
6. The client then sends the service ticket to the server that houses the required resources.
7. The server, in this case the NAS device, decrypts the server portion of the ticket and stores the information in a keytab file. As long as the client's Kerberos ticket is valid, this authorization process does not need to be repeated. The server automatically allows the client to access the appropriate resources.
8. A client-server session is now established. The server returns a session ID to the client, which tracks the client activity, such as file locking, as long as the session is active.



techniques for UNIX users and Windows users are quite different. UNIX files use mode bits to define access rights granted to owners, groups, and other users, whereas Windows uses an ACL to allow or deny specific rights to a particular user for a particular file. Although NAS devices support both of these methodologies for UNIX and Windows users, complexities arise when UNIX and Windows users access and share the same data. If the NAS device supports multiple protocols, the integrity of both permission methodologies must be maintained. NAS device vendors provide a method of mapping UNIX permissions to Windows and vice versa, so a multiprotocol environment can be supported. However, consider these complexities of multiprotocol support when designing a NAS solution. At the same time, validate the domain controller and NIS server connectivity and bandwidth. If multiprotocol access is required, specific vendor access policy implementations need to be considered.

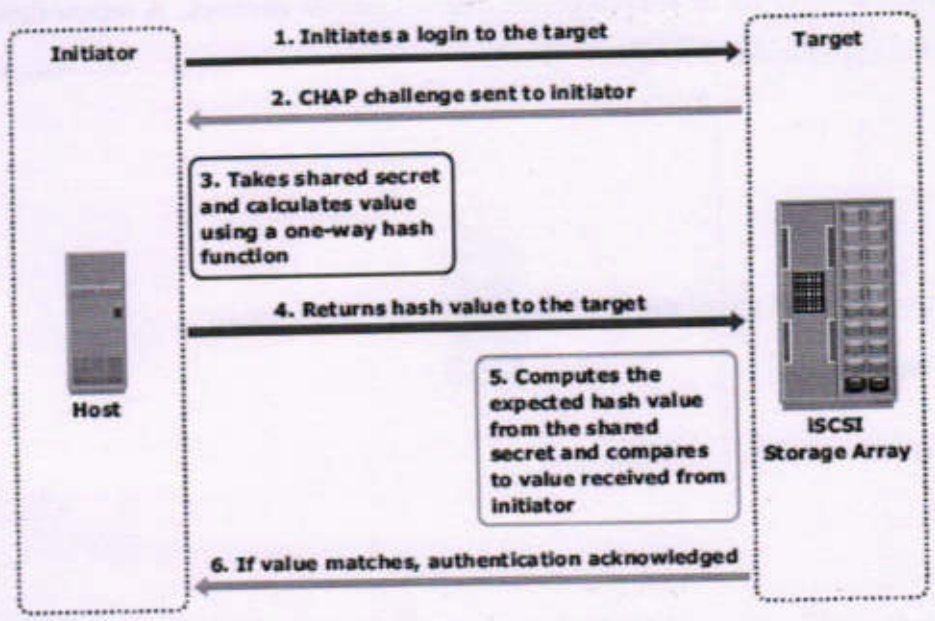


Kerberos

Kerberos is a network authentication protocol, which is designed to provide strong authentication for client/server applications by using secret-key cryptography. It uses cryptography so that a client and server can prove their identity to each other across an insecure network connection. After the client and server have proven their identities, they can choose to encrypt all their communications to ensure privacy and data integrity. In Kerberos, authentications occur between clients and servers. The client gets a ticket for a service and the server decrypts this ticket by using its secret key. Any entity, user, or host that gets a service ticket for a Kerberos service is called a *Kerberos client*. The term *Kerberos server* generally refers to the Key Distribution Center (KDC). The KDC implements the Authentication Service (AS) and the Ticket Granting Service (TGS). The KDC has a copy of every password associated with every principal, so it is absolutely vital that the KDC remain secure. In Kerberos, users and servers for which a secret key is

exchanged directly over the communication channel; rather, a one-way hash function converts it into a hash value, which is then exchanged. A hash function, using the MD5 algorithm, transforms data in such a way that the result is unique and cannot be changed back to its original form. Figure 14-10 depicts the CHAP authentication process. If the initiator requires reverse CHAP authentication, the initiator authenticates the target by using the same procedure. The CHAP secret must be configured on the initiator and the target. A CHAP entry, composed of the name of a node and the secret associated with the node, is maintained by the target and the initiator. The same steps are executed in a two-way CHAP authentication scenario. After these steps are completed, the initiator authenticates the target. If both authentication steps succeed, then data access is allowed. CHAP is often used because it is a fairly simple protocol to implement and can be implemented across a number of disparate systems.

iSNS discovery domains function in the same way as FC zones. Discovery domains provide functional groupings of devices in an IP-SAN. For devices to communicate with one another, they must be configured in the same discovery domain. State change notifications (SCNs) inform the iSNS server when devices are added or removed from discovery domain. Figure 14-11 depicts the discovery domains in iSNS.



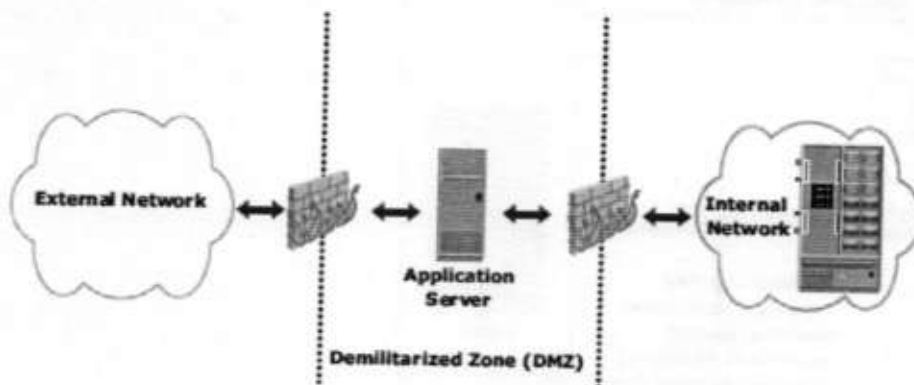
Securing Storage Infrastructure in Virtualized and Cloud Environments

Security Concerns

Organizations are rapidly adopting virtualization and cloud computing, however they have some security concerns. These key security concerns are multitenancy, velocity of attack, information assurance, and data privacy.

Network-Layer Firewalls

Because NAS devices utilize the IP protocol stack, they are vulnerable to various attacks initiated through the public IP network. Network layer firewalls are implemented in NAS environments to protect the NAS devices from these security threats. These network-layer firewalls can examine network packets and compare them to a set of configured security rules. Packets that are not authorized by a security rule are dropped and not allowed to continue to the destination. Rules can be established based on a source address (network or host), a destination address (network or host), a port, or a combination of those factors (source IP, destination IP, and port number). The effectiveness of a firewall depends on how robust and extensive the security rules are. A loosely defined rule set can increase the probability of a security breach. Figure 14-9 depicts a typical firewall implementation. A demilitarized zone (DMZ) is commonly used in networking environments. A DMZ provides a means to secure internal assets while allowing Internet-based access to various resources. In a DMZ environment, servers that need to be accessed through the Internet are placed between two sets of firewalls. Application-specific ports, such as HTTP or FTP, are allowed through the firewall to the DMZ servers. However, no Internet-based traffic is allowed to penetrate the second set of firewalls and gain access to the internal network. The servers in the DMZ may or may not be allowed to communicate with internal resources. In such a setup, the server in the DMZ is an Internet-facing web application accessing data stored on a NAS device, which may be located on the internal private network. A secure design would serve only data to internal and external applications through the DMZ.



IP SAN

This section describes some of the basic security mechanisms used in IP SAN environments. The *Challenge Handshake Authentication Protocol* (CHAP) is a basic authentication mechanism that has been widely adopted by network devices and hosts. CHAP provides a method for initiators and targets to authenticate each other by utilizing a secret code or password. CHAP secrets are usually random secrets of 12 to 128 characters. The secret is never

secure a hypervisor and VMs, virtualized and cloud environments also require further measures on the guest OS and application levels.

Security at the Network Level

The key security measures that minimize vulnerabilities at the network layer are firewall, intrusion detection, demilitarized zone (DMZ), and encryption of data-in-flight. A *firewall* protects networks from unauthorized access while permitting only legitimate communications. In a virtualized and cloud environment, a firewall can also protect hypervisors and VMs. For example, if remote administration is enabled on a hypervisor, access to all the remote administration interfaces should be restricted by a firewall. A firewall also secures VM-to-VM traffic. This firewall service can be provided using a *Virtual Firewall (VF)*. A VF is a firewall service running entirely on the hypervisor. A VF provides packet filtering and monitoring of the VM-to-VM traffic. A VF gives visibility and control over the VM traffic and enforces policies at the VM level. *Intrusion Detection (ID)* is the process to detect events that can compromise the confidentiality, integrity, or availability of a resource. An ID System (IDS) automatically analyzes events to check whether an event or a sequence of events match a known pattern for anomalous activity, or whether it is (statistically) different from most of the other events in the system. It generates an alert if an irregularity is detected. DMZ and data encryption are also deployed as security measures in the virtualized and cloud environments. However, these deployments work in the same way as in the traditional data center.

Security at the Storage Level

Major threats to storage systems in virtualized and cloud environments arise due to compromises at compute, network, and physical security levels. This is because access to storage systems is through compute and network infrastructure. Therefore, adequate security measures should be in place at the compute and network levels to ensure storage security. Common security mechanisms that protect storage include the following:

- Access control methods to regulate which users and processes access the data on the storage systems
- Zoning and LUN masking
- Encryption of data-at-rest (on the storage system) and data-in-transit. Data encryption should also include encrypting backups and storing encryption keys separately from the data. Data shredding that removes the traces of the deleted data

Apart from these mechanisms, isolation of different types of traffic using VSANs further enhances the security of storage systems. In the case of storage utilized by hypervisors, additional security steps are required to protect the storage. Storage for hypervisors using clustered file systems supporting multiple VMs may require separate LUNs for VM components and VM data.

Multitenancy, by virtue of virtualization, enables multiple independent tenants to be serviced using the same set of storage resources. In spite of the benefits offered by multitenancy, it is still a key security concern for users and service providers. Colocation of multiple VMs in a single server and sharing the same resources increase the attack surface. It may happen that business critical data of one tenant is accessed by other competing tenants who run applications using the same resources. *Velocity-of-attack* refers to a situation in which any existing security threat in the cloud spreads more rapidly and has a larger impact than that in the traditional data center environments. *Information assurance* for users ensures confidentiality, integrity, and availability of data in the cloud. Also the cloud user needs assurance that all the users operating on the cloud are genuine and access the data only with legitimate rights and scope. *Data privacy* is also a major concern in a virtualized and cloud environment. A CSP needs to ensure that Personally Identifiable Information (PII) about its clients is legally protected from any unauthorized disclosure.

Security Measures

Security measures can be implemented at the compute, network, and storage levels. These security measures implemented at three layers mitigate the risks in virtualized and cloud environments.

Security at the Compute Level

Securing a compute infrastructure includes enforcing the security of the physical server, hypervisor, VM, and guest OS (OS running within a virtual machine).

Physical server security involves implementing user authentication and authorization mechanisms. These mechanisms identify users and provide access privileges on the server. To minimize the attack surface on the server, unused hardware components, such as NICs, USB ports, or drives, should be removed or disabled.

A *hypervisor* is a single point of security failure for all the VMs running on it. Rootkits and malware installed on a hypervisor make detection difficult for the antivirus software installed on the guest OS. To protect against security-critical hypervisor updates should be installed regularly. Further, the hypervisor management system must also be protected. Malicious attacks and infiltration to the management system can impact all the existing VMs and allow attackers to create new VMs. Access to the management system should be restricted to authorized administrators. Furthermore, there must be a separate firewall installed between the management system and the rest of the network.

VM isolation and *hardening* are some of the common security mechanisms to effectively safeguard a VM from an attack. VM isolation helps to prevent a compromised guest OS from impacting other guest OSs. VM isolation is implemented at the hypervisor level. Apart from isolation, VMs should be hardened against security threats. Hardening is a process to change the default configuration to achieve greater security. Apart from the measures to

Storage tiering:

Storage tiering, or the act of moving data between different types of storage to match the I/O characteristics of the data to that of the storage, is a technique used by administrators for many years (decades even). The simplest and most common motivation for this is data aging. That is, as sections of data age, its access patterns change, most likely being accessed less: the data cools and eventually becomes cold. Cold data has very different I/O patterns, and hence storage needs, from warm and hot data. Even hot data may have different storage needs ² data for an OLTP database workload will have a predominantly random I/O pattern and be well suited to performance solid-state drive (SSD) while the sequential workloads of decision support or analytics workloads will be better suited to capacity SSD or even hard disk drive (HDD). Over the years, the tools and techniques available to administrators have varied, but one thing always has been the same: there is a range of storage media with different characteristics (price, performance etc.). The introduction of Oracle Database 12c and the Oracle FS1-2 storage system adds new techniques Tables, Tablespaces, and Storage It is necessary to understand the layers in the storage stack before delving into the various techniques for moving data within a database between storage tiers. Figure 2 shows these layers. They start with physical storage within the Oracle FS1-2 flash storage system (capacity hard drives, performance hard drives, capacity SSDs and/or performance SSDs) and move up to the volumes (virtual disks or LUNs) presented by Oracle FS1-2. Oracle Automatic Storage Management, a feature of Oracle DataBase, performs the function of a logical volume manager; it consumes the volumes presented by Oracle FS1-2 and presents them to the database as disk groups and each has one or more datafiles that map to the underlying operating system. Database tables exist within tablespaces, and each tablespace is mapped to a specific Oracle Automatic Storage Management disk group

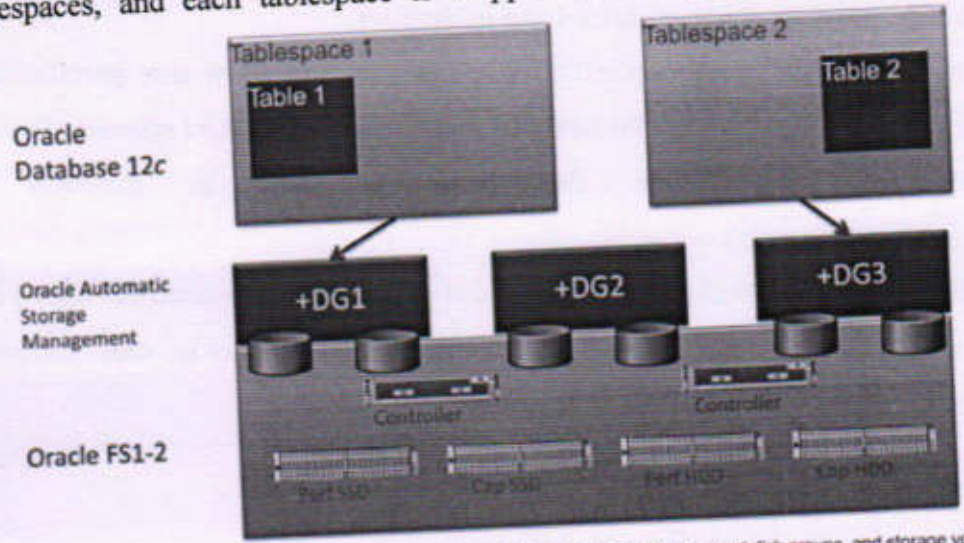


Figure 2: The relationship among tables, tablespaces, Oracle Automatic Storage Management disk groups, and storage volumes

Information lifecycle management (ILM)

Information Life Cycle Management (ILM) is a complex Data Life Management Cycle (DLM) subset and Records and Information Management (RIM) practice used for data storage, system administration and management. The strategic ILM approach is used to determine how data is moved, deleted, destroyed or archived and is based on automated storage procedures, manual data organization formats (paper, microfilm, photographs, negatives and audio/video recordings) and early data storage management, like hierarchical storage management (HSM).

ILM is effective in modern computing where data management is critical, due to compliance issues from legislation like the Health Insurance Portability and Accountability Act (HIPAA) and the Sarbanes-Oxley Act. Both are used to regulate particular types of data management. ILM uses more complex criteria than data file type, age and access frequency. ILM products automate data management by organizing data and automating data migration into tiers categorized by policy criteria. ILM is based on three storage strategies, as follows:

- **Policy:** Determined by business goals and drivers. Storage and information policies are shaped by executive and managerial determinations of IT governance and management, service level agreements (SLA), change control processes and system availability and recovery time requirements in the event of unexpected events such as accidents or disasters.
- **Operational:** Includes data backup and recovery, like data restoration and system restarts; archiving (long term data retention) and other daily processes and procedures for storage management.
- **Infrastructure:** Includes logical and physical architectures, such as simulated and physical hard drive partitions; applications and corresponding storage platforms related to required production, testing and development; data storage security and data center capacities and limitations.

ILM's path management feature is used to facilitate stored application data retrieval and allow user specification of policies that define data values according to different times, rates and lifespan. For example, ILM systems allow users to search for various types of stored data file instances, such as customer IDs.

Unforeseen circumstances occur outside of normal business operations and cannot be automated. An example is a legal hold, also known as a litigation hold or legal freeze, which requires data administrators to cease normal ILM data flow continuation.

Next is the concept of partitioning. While in Figure 2 it may be that all the data within a table becomes cold and hence, the DBA wants to move the whole table to a different storage tier, much more likely is that just parts of the data become cold. This is where partitioning plays an important role. Partitioning allows a table, index, or index-organized table to be subdivided into smaller pieces, and each piece of such a database object is called a partition. Each partition has its own name, and may optionally have its own storage characteristics. From the perspective of a database administrator, a partitioned object has multiple pieces that can be managed either collectively or individually. This gives the administrator considerable flexibility in managing partitioned objects. However, from the perspective of the application, a partitioned table is identical to a nonpartitioned table; no modifications are necessary when accessing a partitioned table using SQL queries and DML statements. Each row in a partitioned table is unambiguously assigned to a single partition. The partitioning key is comprised of one or more columns that determine the partition in which each row will be stored. Oracle Database 12c automatically directs insert, update, and delete operations to the appropriate partition through the use of the partitioning key. The administrator has considerable flexibility about the choice of partitioning key(s), but, as Figure 3 demonstrates, by far the most common choice is based on date and/or time. For example, in an order entry . Figure 4 is a variant of Figure 1 in which a partitioned table is added. Table 3 has four partitions, two of which are in Tablespace 1 in disk group +DG1 and two of which have been moved to Tablespace 2 in disk group +DG3.

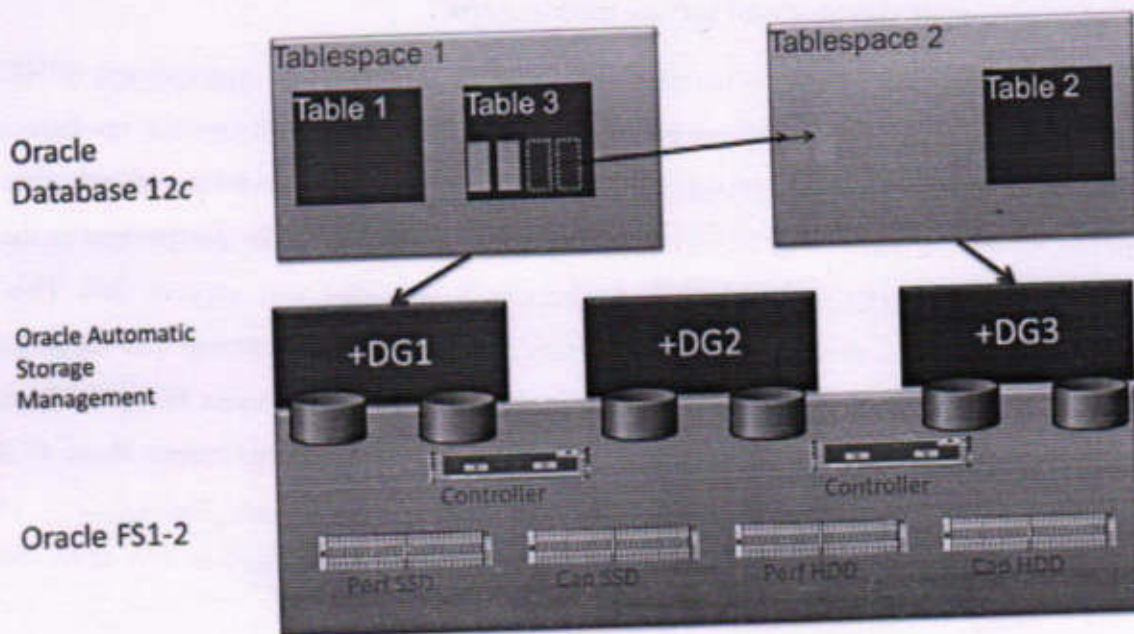


Figure 4: The relationship among tables, partitions, tablespaces, and storage

Cloud service management activities

The delivery of dynamic, cloud-based infrastructure, platform and application services doesn't occur in a vacuum. In addition to best practices for effective administration of all the elements associated with cloud service delivery, cloud service management and cloud monitoring tools enable providers to keep up with the continually shifting capacity demands of a highly elastic environment. Cloud monitoring and cloud service management tools allow cloud providers to ensure optimal performance, continuity and efficiency in virtualized, on-demand environments. These tools -- software that manages and monitors networks, systems and applications -- enable cloud providers not just to guarantee performance, but also to better orchestrate and automate provisioning of resources. Cloud monitoring tools specifically, enable cloud providers to track the performance, continuity and security of all of the components that support service delivery: the hardware, software and services in the data center and throughout the network infrastructure.

Through successful cloud service management and monitoring, cloud providers can use service quality to differentiate themselves in what remains a crowded and noisy marketplace. Effective cloud service management also helps lower the risk of frequent cloud outages that can jeopardize security systems. Using these tools also support greater operational efficiency, helping cloud providers minimize costs and maximize profit margins. However, achieving these goals can be difficult in a complex virtual delivery environment where visibility and control are limited.

What are the key processes associated with cloud service management?

Cloud service management shares some basic principles with traditional IT service management (ITSM). Cloud management tools help providers administrate the systems and applications that facilitate the on-demand service delivery model. The goal of these practices is to improve the efficiency of the cloud environment and achieve a high level of customer satisfaction. Essentially, cloud service management takes the customer perspective as the measure of service assurance and manages all the individual IT resources in a way that will support that. This involves adjusting the operations and policies, as necessary, of all the assets in the virtual environment that support and affect the on-demand service delivery model. Such assets include servers, software and services that provide access and connectivity to these cloud services. The core elements of cloud service management mirror those of traditional ITSM -- including cloud service-level agreement (SLA) management, cloud capacity management, availability management and billing -- and are applied to administrate a cloud delivery environment in a systemic way. These

processes are supported with tools that track provisioning and change management, configuration management, release management, incident management, performance management and service continuity. Customers are supported directly and indirectly through a help desk function. Cloud service management is complemented by monitoring software that tracks operational information and feeds that data to the appropriate management resource. Given the elastic, highly virtualized nature of cloud environments, there are some key differences in approaches to cloud service management and conventional IT service management. The two disciplines have different objectives, requiring tools that emphasize their individual requirements. Whereas the goals of traditional ITSM are effective SLA management, improved performance and streamlined billing, the goal of cloud service management is to orchestrate resources for fast provisioning, effective capacity management and ongoing service stability. Automation is vital to ensure efficiency and reduce costs.

Cloud service management platforms: Build or buy?

Although vendors have developed many cloud service management and monitoring tools for enterprises that build and manage their own private clouds, there are far fewer tools that meet the scale, security and performance requirements of cloud providers. Beyond that, there are even fewer solutions that provide the comprehensive capabilities associated with the entire ITSM process for cloud providers, namely orchestration.

CBCS SCHEME

17CS754

Seventh Semester B.E. Degree Examination, Jan./Feb. 2021

Storage Area Networks

Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. Explain the key characteristics of Data centre with neat diagram. (08 Marks)
b. Explain the core elements of Data center. (04 Marks)
c. Discuss the process of mapping user files to disk storage with neat diagram. (08 Marks)

OR

- 2 a. Describe the concept of Mirroring and Parity. (04 Marks)
b. What is RAID? Explain the RAID levels with reference to nested RAID, RAID 3 and RAID 5 with neat diagram. (08 Marks)
c. Discuss the components of an intelligent storage system with neat diagram. (08 Marks)

Module-2

- 3 a. List and explain different Fibre channel connectivity options with neat diagram. (08 Marks)
b. Define FCOE. Explain components of an FCOE network. (08 Marks)
c. Define Zoning. Explain types of Zoning. (04 Marks)

OR

- 4 a. Discuss components of NAS with neat diagram. (06 Marks)
b. List and explain benefits of NAS. (06 Marks)
c. Explain object storage and Retrieval in OSD with diagram. (08 Marks)

Module-3

- 5 a. Define Business Continuity. Explain BC terminology in detail. (06 Marks)
b. Discuss different Backup Topologies. (08 Marks)
c. Explain the concept of Backup in virtualized Environments. (06 Marks)

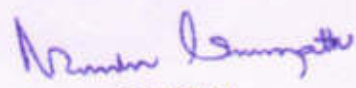
OR

- 6 a. Explain local Replication technology using Host based methods. (08 Marks)
b. Discuss synchronous + Asynchronous and Synchronous + Disk Buffered of three site replication. (06 Marks)
c. Explain the concept of Remote replication and migration in a Virtualized Environment. (06 Marks)

Module-4

- 7 a. Define Cloud Computing. List and explain the essential characteristics of cloud computing. (08 Marks)
b. List the cloud service models and discuss any two of them. (08 Marks)
c. List and explain benefits of cloud computing. (04 Marks)

1 of 2


PRINCIPAL
SIET, TUMAKURU.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and/or equations written eg. 42+8 = 50, will be treated as malpractice.

OR

- 8 a. Explain Cloud Deployment models in detail. (10 Marks)
- b. Explain Cloud Computing infrastructure in detail. (10 Marks)

Module-5

- 9 a. List and explain the different types of security threats. (06 Marks)
- b. Discuss IPSAN CHAP protocol with neat diagram. (06 Marks)
- c. Discuss security solutions for FC-SAN and NAS. (08 Marks)

OR

- 10 a. List and describe storage infrastructure management activities. (04 Marks)
- b. Explain Information lifecycle management with proper example. (08 Marks)
- Discuss two methods of storage tiering. (08 Marks)



Manjunath Kumar

PRINCIPAL
SIET, TUMAKURU.



CBCS SCHEME

15CS754

--	--	--	--	--	--	--	--	--	--

Seventh Semester B.E. Degree Examination, June/July 2019 Storage Area Networks

Time: 3 hrs.

Max. Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. What is a data center? List the core components of data center. Explain the characteristics of data center. (08 Marks)
- b. Discuss volume manager and compute virtualization in detail. (08 Marks)

OR

- 2 a. Differentiate between software and hardware RAID. Illustrate how parity method is used for RAID levels. (08 Marks)
- b. With a neat diagram explain ISS. Explain in detail the cache component of ISS. (08 Marks)

Module-2

- 3 a. List and explain different FC connectivity options with a neat diagram. (08 Marks)
- b. With diagram explain ISCSI implementation. (08 Marks)

OR

- 4 a. What is NAS? Explain NAS implementation in detail. (08 Marks)
- b. List the key features of Content Addressed Storage (CAS). Illustrate with a neat block diagram the unified storage for CAS system. (08 Marks)

Module-3

- 5 a. Explain with a neat diagram BC planning lifecycle. (08 Marks)
- b. Mention backup topologies. List various backup forget solution and explain any one with a neat diagram. (08 Marks)

OR

- 6 a. List various uses of local replication. Explain storage array based local replication with a neat diagram. (08 Marks)
- b. Differentiate between Synchronous and Asynchronous based remote replication model. (08 Marks)

Module-4

- 7 a. List various cloud computing characteristics. Explain the cloud computing infrastructure components with a neat diagram. (08 Marks)
- b. With diagram explain different cloud deployment models. (08 Marks)

OR

- 8 Explain in detail in band and out of band virtualization appliances with a neat diagram. (16 Marks)

Module-5

- 9 a. What are the different rules tried for information security? Explain in detail FCSAN based security implementation. (08 Marks)
- b. List and explain different storage infrastructure management activities in detail. (08 Marks)

OR

- 10 a. Explain different storage management activities. (08 Marks)
- b. What is ILM? List and explain various benefits of ILM. (08 Marks)

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and/or equations written eg. 42+8 = 50, will be treated as malpractice.

Principal

PRINCIPAL
SIET, TUMAKURU.

CBCS SCHEME

USN

--	--	--	--	--	--	--	--	--	--

15CS754

Seventh Semester B.E. Degree Examination, Dec.2018/Jan.2019 Storage Area Networks

Time: 3 hrs.

Max. Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. What is a data center? Explain key characteristics of data center elements with diagram. (08 Marks)
- b. What is a file system? Explain the process of mapping user files to the disk storage. (08 Marks)

OR

- 2 a. What is RAID? Explain the RAID levels with reference to nested RAID, RAID3, RAID5 with neat diagram. (08 Marks)
- b. With neat diagram, explain the structure of read and write operations with cache. (08 Marks)

Module-2

- 3 a. Explain FC connectivity options with relevant diagram. (08 Marks)
- b. Explain block-level storage virtualization with neat diagram. Explain VSAN in brief. (08 Marks)

OR

- 4 a. What is FCoE? Explain the components of FCoE with neat diagram. (08 Marks)
- b. What is NAS? Explain the benefits of NAS. (08 Marks)

Module-3

- 5 a. What is business continuity? Explain the BC Terminology in detail. (08 Marks)
- b. Explain Backup and Restore operations with neat diagram. (08 Marks)

OR

- 6 a. What is data deduplication? Explain the implementation of data deduplication. (08 Marks)
- b. Explain Synchronous + Asynchronous and Synchronous + Disk Buffered methods of three-site replication with neat diagram. (08 Marks)

Module-4

- 7 a. What is cloud computing? Explain the characteristics and benefits of cloud computing? (08 Marks)
- b. Explain the various cloud service models available. (08 Marks)

OR

- 8 a. Explain the public cloud and private cloud deployment models in cloud computing. (08 Marks)
- b. Explain the cloud computing infrastructure in detail. (08 Marks)

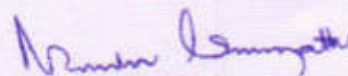
Module-5

- 9 a. Explain FC SAN security architecture with neat diagram. (08 Marks)
- b. Explain the concept of Kerberos with neat diagram. (08 Marks)

OR

- 10 a. Explain the storage management activities in detail. (08 Marks)
- b. Explain Information Lifecycle Management (ILM) in detail with challenges. (08 Marks)

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.



PRINCIPAL
SIET, TUMAKURU.



CBCS SCHEME

15CS754

Seventh Semester B.E. Degree Examination, Dec.2019/Jan.2020 Storage Area Networks

Time: 3 hrs.

Max. Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. Explain with neat diagram the Evolution of storage Architecture. (06 Marks)
b. Discuss core Elements of Data center and key characteristics of Data center. (10 Marks)

OR

- 2 a. Describe with neat block diagram the components of Intelligent storage system. (08 Marks)
b. With diagram explain different RAID Techniques. (08 Marks)

Module-2

- 3 a. Explain with neat diagram the components of Fiber Channels (FC) storage Area Networks. (08 Marks)
b. What is zoning? Explain its types. (08 Marks)

OR

- 4 a. Discuss different iSCSI Topologies with neat diagrams. (08 Marks)
b. Write short notes on Fiber Channel Over Ethernet (FCOE). (08 Marks)

Module-3

- 5 a. Discuss different back up Topologies. (08 Marks)
b. What is data deduplication? Explain its implementation methods. (08 Marks)

OR

- 6 a. Explain local Replication technology using Host based methods. (06 Marks)
b. Write a short notes on the following ; (10 Marks)
i) Three site Replications ii) Network based Remote Replication.

Module-4

- 7 a. Explain the characteristics of clouds computing. (04 Marks)
b. Discuss cloud Deployment models. (06 Marks)
c. Explain Cloud computing Infrastructure. (06 Marks)

OR

- 8 a. Discuss the steps involved in transitioning from classic data center to cloud computing Environment service. (08 Marks)
b. Write a short notes on the following : (08 Marks)
i) Business drives for cloud computing
ii) Cloud migration considerations.

Module-5

- 9 a. Explain the different types of security threats. (06 Marks)
b. Discuss security solutions for FC - SAN and IP-SAN. (10 Marks)

OR

- 10 a. Explain the various information infrastructure components in classic and virtual Environments. (08 Marks)
b. Write a short notes on the following : (08 Marks)
i) Information Life Cycle Management (ILM). ii) Storage Tiering.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and/or equations written eg. 42+8 = 50, will be treated as malpractice.

Principal
PRINCIPAL
SIET, TUMAKURU.

CBCS SCHEME

USN

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

15CS754

Seventh Semester B.E. Degree Examination, Jan./Feb.2021

Storage Area Networks

Time: 3 hrs.

Max. Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. Explain evolution of storage architecture with neat diagram. (08 Marks)
b. Discuss the key characteristics of a data centre, with neat diagram. (08 Marks)

OR

- 2 a. What is protocol? Explain the popular interface protocols used for host to storage communications. (08 Marks)
b. Explain two different types of intelligent storage systems. (08 Marks)

Module-2

- 3 a. Explain Fibre channel protocol stack with neat diagram. (08 Marks)
b. What is zoning? Explain its types. (08 Marks)

OR

- 4 a. What is FCIP? Explain FCIP protocol stack. (08 Marks)
b. What is NAS? Explain its components with neat sketch. (08 Marks)

Module-3

- 5 a. Explain any two backup topologies with neat diagram. (08 Marks)
b. Discuss Data Deduplication implementations. (08 Marks)

OR

- 6 a. Give different uses of local replicas. (08 Marks)
b. Explain remote replication technologies. (08 Marks)

Module-4

- 7 a. What is cloud computing? Give its characteristics. (08 Marks)
b. Explain different cloud service models. (08 Marks)

OR

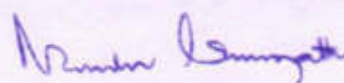
- 8 a. Explain various cloud deployment models. (08 Marks)
b. Discuss cloud challenges. (08 Marks)

Module-5

- 9 a. Explain storage security domains. (08 Marks)
b. Explain various security measures of cloud environment. (08 Marks)

OR

- 10 a. Explain various storage infrastructure management activities. (08 Marks)
b. Explain information lifecycle management. (08 Marks)



Seventh Semester B.E. Degree Examination, June/July 2016

Storage Area Networks

Time: 3 hrs.

Max. Marks:100

Note: Answer FIVE full questions, selecting at least TWO questions from each part.

PART - A

- 1 a. What is a data center? Explain key characteristics of data center elements (10 Marks)
b. Explain the various components of disk drive. (06 Marks)
c. Consider a disk I/O system in which an I/O request arrives at the rate of 80IOPS. The disk service time is 6 ms. Compute the following:
i) Utilization ii) Response time
iii) Average queue size iv) Time spent by a request in a queue. (04 Marks)
- 2 a. Explain the various techniques on the basis of which RAID levels are defined. (09 Marks)
b. An application has 1000 heavy users at a peak of 2 IOPS each and 2000 typical users at a peak of 1 IOPS each, with a read/write ratio of 2:1. It is estimated that the application also experiences an overhead of 10 percent for other workloads. Calculate the IOPS requirement for RAID1, RAID5 and RAID6. (06 Marks)
c. With a neat diagram, differentiate between write through and write back cache. (05 Marks)
- 3 a. Explain fibre channel with respect to protocol stack, zoning and login types. (10 Marks)
b. Write a note on SCSI-3 architecture. (10 Marks)
- 4 a. What is NAS? Explain the benefits of NAS. (10 Marks)
b. Differentiate between Native and Bridged iSCSI connectivity. (06 Marks)
c. Write a note on iSCSI PDU. (04 Marks)

PART - B

- 5 a. With neat diagrams, explain the concepts of object storage and retrieval in CAS systems. (10 Marks)
b. What is storage virtualization? Differentiate between block level and file level virtualization with neat diagrams. (10 Marks)
- 6 a. What is business continuity? Explain BC planning life cycle with a neat diagram. (10 Marks)
b. Explain the reasons for which backup is performed. (10 Marks)
- 7 a. Describe the various storage array based local replication technologies. (10 Marks)
b. What is storage array based remote replication? Differentiate between synchronous and asynchronous replication mode in it. (10 Marks)
- 8 a. Explain the four security attributes which are under threat. (04 Marks)
b. Write a note on risk triad. (06 Marks)
c. Describe the categories on the basis of which storage management is classified. (05 Marks)
d. Write a note on accessibility monitoring. (05 Marks)

Principals Signature

PRINCIPAL
SIET, TUMAKURU

Seventh Semester B.E. Degree Examination, Aug./Sept.2020
Storage Area Networks

Time: 3 hrs.

Max. Marks: 80

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. Explain the key characteristics of a data center, with a neat diagram. (08 Marks)
 b. With a neat diagram, explain the structure of read and write operations in cache. (08 Marks)

OR

- 2 a. List the different RAID levels where parity technique has been adopted. Explain any three. (10 Marks)
 b. Compare virtual and traditional storage provisioning. (06 Marks)

Module-2

- 3 a. What is zoning? What are the advantages of zoning? Explain the various types of zoning. (08 Marks)
 b. Write a note on iSCSI. (08 Marks)

OR

- 4 a. Explain the fibre channel Protocol stack with neat figure. (08 Marks)
 b. Explain I/O consolidation using FCoE. (08 Marks)

Module-3

- 5 a. Define the following terminologies:
 i) MTBF ii) RPO iii) MTTR iv) RFO. (08 Marks)
 b. Describe the failure analysis in BC. Mention some important BC technology solutions. (08 Marks)

OR

- 6 a. Explain backup and restore operations with neat diagram. (08 Marks)
 b. Explain Backup in virtualized environments. (08 Marks)

Module-4

- 7 a. Define cloud computing. List and explain the essential characteristics of cloud computing. (06 Marks)
 b. Classify the deployment models in cloud computing. Explain any two. (10 Marks)

OR

- 8 a. List and explain the challenges facing in cloud-computing. (06 Marks)
 b. Explain cloud infrastructure layers, with diagram. (10 Marks)

Module-5

- 9 a. Write a note on: i) Risk triad ii) Threats (08 Marks)
 b. Explain the concept of Kerberos with neat diagram. (08 Marks)

OR

- 10 a. Discuss the IPSAN security implementation in storage networking. (06 Marks)
 b. Explain the storage infrastructure management activities in detail with example. (10 Marks)



Seventh Semester B.E. Degree Examination, Jan./Feb. 2021
Storage Area Networks

Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

- 1 a. Explain the key characteristics of Data centre with neat diagram. (08 Marks)
- b. Explain the core elements of Data center. (04 Marks)
- c. Discuss the process of mapping user files to disk storage with neat diagram. (08 Marks)

OR

- 2 a. Describe the concept of Mirroring and Parity. (04 Marks)
- b. What is RAID? Explain the RAID levels with reference to nested RAID, RAID 3 and RAID 5 with neat diagram. (08 Marks)
- c. Discuss the components of an intelligent storage system with neat diagram. (08 Marks)

Module-2

- 3 a. List and explain different Fibre channel connectivity options with neat diagram. (08 Marks)
- b. Define FCOE. Explain components of an FCOE network. (08 Marks)
- c. Define Zoning. Explain types of Zoning. (04 Marks)

OR

- 4 a. Discuss components of NAS with neat diagram. (06 Marks)
- b. List and explain benefits of NAS. (06 Marks)
- c. Explain object storage and Retrieval in OSD with diagram. (08 Marks)

Module-3

- 5 a. Define Business Continuity. Explain BC terminology in detail. (06 Marks)
- b. Discuss different Backup Topologies. (08 Marks)
- c. Explain the concept of Backup in virtualized Environments. (06 Marks)

OR

- 6 a. Explain local Replication technology using Host based methods. (08 Marks)
- b. Discuss synchronous + Asynchronous and Synchronous + Disk Buffered of three site replication. (06 Marks)
- c. Explain the concept of Remote replication and migration in a Virtualized Environment. (06 Marks)

Module-4

- 7 a. Define Cloud Computing. List and explain the essential characteristics of cloud computing. (08 Marks)
- b. List the cloud service models and discuss any two of them. (08 Marks)
- c. List and explain benefits of cloud computing. (04 Marks)

USN

--	--	--	--	--	--	--	--	--	--

10CS/IS765

Seventh Semester B.E. Degree Examination, Dec.2015/Jan.2016
Storage Area Networks

Time: 3 hrs.

Max. Marks:100

**Note: Answer FIVE full questions, selecting
at least TWO questions from each part.**

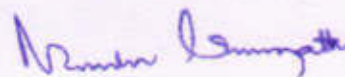
PART - A

- 1 a. A hospital uses an application that stores patient X-ray data in the form of large binary objects in an oracle database. The application is hosted on a UNIX server, and the hospital staff accesses the X-ray records through a GB Ethernet backbone. An EMC CLARiiion storage array provides storage to the UNIX server, which has 6 TB of usable capacity. Explain the core elements of the data center, and key requirements for data center elements. What are the typical challenges the storage management team may face in meeting the service-level demands of the hospital staff? (12 Marks)
- b. Consider a disk I/O system in which an I/O request arrives at the rate of 80 IOPS. The disk service time is 6ms.
 - i) Compute the following: utilization of I/O controller, total response time, average queue size and total time spent by a request in a queue.
 - ii) Compute the preceding parameter if the service time is halved. (08 Marks)
- 2 a. An application has 1000 heavy users at a peak of 2 IOPS each and 2000 typical users at a peak of 1 IOPS each, with a read/write ratio of 2:1. It is estimated that the application also experiences an overhead of 20% for other workloads. Calculate the IOPS required for RAID 1, RAID 3, RAID 5 and RAID 6. Also compute the number of drives required to support the application in different RAID environments if 10K rpm drives with a rating of 130 IOPS per drive were used. (10 Marks)
- b. Categories and explain intelligent storage systems with diagram. (10 Marks)
- 3 a. If three hard disk drives are connected in a daisy chain and communicate over SCSI, explain SCSI-3 standard architecture and SCSI communication model. (10 Marks)
- b. What is zoning? Discuss a scenario, where soft zoning is preferred and where hard zoning is preferred. (05 Marks)
- c. Differentiate between full and partial mesh topology. (05 Marks)
- 4 a. What are the factors affecting NAS performance? (04 Marks)
- b. Draw and explain the components, the topologies and the protocol stack of iSCSI. (16 Marks)

PART - B

- 5 a. Explain the data object storage process and process of data retrieval from CAS system with diagram. (10 Marks)
- b. Illustrate a NAS environment before and after the implementation of file level virtualization. (10 Marks)
- 6 a. Draw and explain BC planning life cycle. (12 Marks)
- b. What are different back-up topologies? Explain. (08 Marks)

1 of 2



PRINCIPAL
SIET, TUMAKURU.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg. 42+8 = 50, will be treated as malpractice.

- 7 a. A host generates 8000 I/Os at peak utilization with an average I/O size of 32 KB. The response time is currently measured at an average of 12 ms during peak utilizations. When synchronous replication is implemented with a fibre channel link to a remote site, what is the response time experienced by the host if the network latency is 6 ms per I/O? (04 Marks)
- b. What is the importance of recoverability and consistency in local replication? (04 Marks)
- c. Discuss the effects of a bunker failure in a three-site replication for the following implementations: (12 Marks)
- i) Multihop synchronous and disk buffered
 - ii) Multihop synchronous and asynchronous
 - iii) Multitarget
- 8 a. What are monitoring parameters and components monitored for storage infrastructure? Explain in details. (10 Marks)
- b. Explain storage infrastructure management activities in detail. (10 Marks)



10CS/IS765

Seventh Semester B.E. Degree Examination, June/July 2016

Storage Area Networks

Time: 3 hrs.

Max. Marks:100

Note: Answer FIVE full questions, selecting at least TWO questions from each part.

PART - A

1. a. What is a data center? Explain key characteristics of data center elements. (10 Marks)
b. Explain the various components of disk drive. (06 Marks)
c. Consider a disk I/O system in which an I/O request arrives at the rate of 80 IOPS. The disk service time is 6 ms. Compute the following:
i) Utilization
ii) Average queue size
iii) Response time
iv) Time spent by a request in a queue. (04 Marks)
2. a. Explain the various techniques on the basis of which RAID levels are defined. (09 Marks)
b. An application has 1000 heavy users at a peak of 2 IOPS each and 2000 typical users at a peak of 1 IOPS each, with a read/write ratio of 2:1. It is estimated that the application also experiences an overhead of 10 percent for other workloads. Calculate the IOPS requirement for RAID1, RAID5 and RAID6. (06 Marks)
c. With a neat diagram, differentiate between write through and write back cache. (05 Marks)
3. a. Explain fibre channel with respect to protocol stack, zoning and login types. (10 Marks)
b. Write a note on SCSI-3 architecture. (10 Marks)
4. a. What is NAS? Explain the benefits of NAS. (10 Marks)
b. Differentiate between Native and Bridged iSCSI connectivity. (06 Marks)
c. Write a note on iSCSI PDU. (04 Marks)

PART - B

5. a. With neat diagrams, explain the concept of object storage and retrieval in CAS systems. (10 Marks)
b. What is storage virtualization? Differentiate between block level and file level virtualization with neat diagrams. (10 Marks)
6. a. What is business continuity? Explain BC planning life cycle with a neat diagram. (10 Marks)
b. Explain the reasons for which backup is performed. (10 Marks)
7. a. Describe the various storage array based local replication technologies. (10 Marks)
b. What is storage array based remote replication? Differentiate between synchronous and asynchronous replication mode in it. (10 Marks)
8. a. Explain the four security attributes which are under threat. (04 Marks)
b. Write a note on risk triad. (06 Marks)
c. Describe the categories on the basis of which storage management is classified. (05 Marks)
d. Write a note on accessibility monitoring. (05 Marks)

Principal

PRINCIPAL
SIET, TUMAKURU.



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: I
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:20/05/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

1. a. Explain the evolution of storage technology with neat diagram. 8M|CO1|
b. What is connectivity? List and explain the physical components of connectivity 8M |CO5|
c. Differentiate b/w journaling and non - journaling file system. 4M|CO1|

OR

2. a. With neat diagram, explain the components of ISS. 10M|CO1|
b. What is a data center? Explain core elements, key characteristics and key management activities of data center. 10M|CO5|
3. a. Explain the following with appropriate diagram. i)Nested RAID ii)RAID 6 iii)RAID 0 10M|CO1|
b. Explain compute virtualization 5M|CO1|
c. Explain the following interface protocols (i)IDE/ATA & Serial ATA (ii)SCSI & Serial SCSI 5M|CO1|

OR

4. a Explain any 2 techniques on the basis of which RAID levels are define. 10M|CO1|
b Explain virtualization &cloud computing. 5M|CO1|
c. Explain any 2 software components that are essential parts of a host system. 5M|CO1|



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: I
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:20/05/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

- 1.a. Explain the evolution of storage technology with neat diagram. 8M|CO1|
b. What is connectivity? List and explain the physical components of connectivity 8M |CO5|
c. Differentiate b/w journaling and non - journaling file system. 4M|CO1|

OR

2. a. With neat diagram, explain the components of ISS. 10M|CO1|
b. What is a data center? Explain core elements, key characteristics and key management activities of data center. 10M|CO5|
3. a. Explain the following with appropriate diagram. i)Nested RAID ii)RAID 6 iii)RAID 0 10M|CO1|
b. Explain compute virtualization 5M|CO1|
c. Explain the following interface protocols (i)IDE/ATA & Serial ATA (ii)SCSI & Serial SCSI 5M|CO1|

OR

4. a Explain any 2 techniques on the basis of which RAID levels are define. 10M|CO1|
b Explain virtualization &cloud computing. 5M|CO1|
c. Explain any 2 software components that are essential parts of a host system. 5M|CO1|


PRINCIPAL
SIET., TUMAKURU.

Internal Assessment Test-1

Scheme

1. a. Diagram 4M
Explanation 4M

b. Connectivity def 2M
List 2M
Explanation 4M

c. Any 4 points Each carry one marks

2. a. Diagram 3M
Explanation 7M

b. Data Center def 2M
Core elements 3M
Key Characteristics 3M
Key management 2M

3. a. (i) Diagram 2M
~~Explanation~~ Explanation 2M

(iii) & (ii) Diagram 1.5M
Explanation 1.5M

b. Diagram 2M
Explanation 3M

4. a. Explain any 2 Techniques Each Carry 5M.

Diagram 2M
Explanation 3M

b. Virtualization 2.5M

Cloud Computing 2.5M

c. Explain any 2 S/W Components Each
Carry 2.5M.

1. a. Explain the evolution of storage technology with neat diagram. [D4+E4]

➤ Historically, organizations had centralized computers (mainframe) and information storage devices (tape reels and disk packs) in their data center.

➤ The evolution of open systems and the affordability and ease of deployment that they offer made it possible for business units/departments to have their own servers and storage.

➤ In earlier implementations of open systems, the storage was typically internal to the server. This approach is referred to as server-centric storage architecture (see Fig 1.4 [a]).

➤ In this server-centric storage architecture, each server has a limited number of storage devices, and any administrative tasks, such as maintenance of the server or increasing storage capacity, might result in unavailability of information.

➤ The rapid increase in the number of departmental servers in an enterprise resulted in unprotected, unmanaged, fragmented islands of information and increased capital and operating expenses.

➤ To overcome these challenges, storage evolved from server-centric to information-centric architecture.

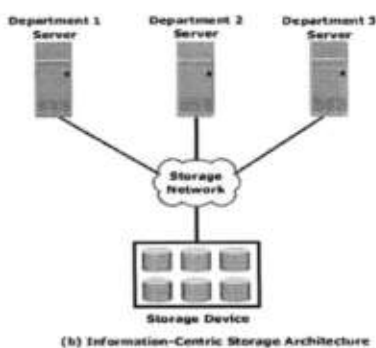
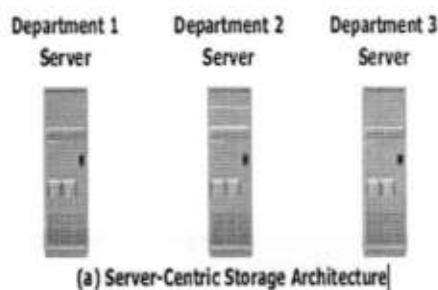


Fig: Evolution of storage architecture

➤ In information-centric architecture, storage devices are managed centrally and independent of servers.

➤ These centrally-managed storage devices are shared with multiple servers.

➤ When a new server is deployed in the environment, storage is assigned from the same shared storage devices to that server.

➤ The capacity of shared storage can be increased dynamically by adding more storage devices without impacting information availability.

➤ In this architecture, information management is easier and cost-effective. ➤ Storage technology and architecture continues to evolve, which enables organizations to consolidate, protect, optimize, and leverage their data to achieve the highest return on information assets.

b. What is connectivity? List and explain the physical components of connectivity [2+6]

- Connectivity refers to the interconnection between hosts or between a host and peripheral devices, such as printers or storage devices.
- Connectivity and communication between host and storage are enabled using physical components and interface protocols.

➤ **Physical Components of Connectivity:**

- ✓ the host interface device
- ✓ port
- ✓ cable.

➤ A host interface device or host adapter connects a host to other hosts and storage devices.

✓ E.g.: host bus adapter (HBA) and network interface card (NIC).

✓ HBA is an application-specific integrated circuit (ASIC) board that performs I/O interface functions between the host and storage, relieving the CPU from additional I/O processing workload.

✓ A host typically contains multiple HBAs.

➤ A port is a specialized outlet that enables connectivity between the host and external devices. An HBA may contain one or more ports to connect the host.

➤ Cables connect hosts to internal or external devices using copper or fiber optic media.

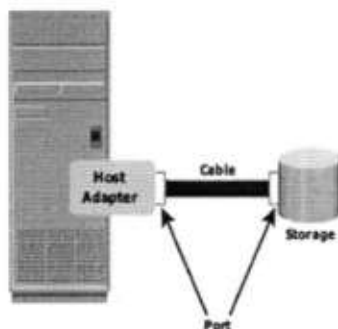


Figure 2-4: Physical components of connectivity

c. Differentiate b/w journaling and non - journaling file system. [2+2]

Non-Journaling file System:

Non journaling file systems cause a potential loss of files because they use separate writes to update their data and metadata.

- If the system crashes during the write process, the metadata or data might be lost or corrupted.
- system reboots, the file system attempts to update the metadata structures by examining and repairing them. This operation takes a long time on large file systems.
- If there is insufficient information to re-create the wanted or original structure, the files might be misplaced or lost, resulting in corrupted file systems.

Journaling file system:

- A journaling file system uses a separate area called a log or journal.
- This journal might contain all the data to be written (physical journal) or just the metadata to be updated (logical journal). Before changes are made to the file system, they are written to this separate area.

- After the journal has been updated, the operation on the file system can be performed.
- Journaling File systems is that they are slower than other file systems.
- This slowdown is the result of the extra operations that have to be performed on the journal each time the file system is changed.

OR

2. a. With neat diagram, explain the components of ISS. {D3 + E7}

Intelligent Storage Systems are feature-rich RAID arrays that provide highly optimized I/O processing capabilities.

➤ These storage systems are configured with a large amount of memory (called cache) and multiple I/O paths and use sophisticated algorithms to meet the requirements of performance-sensitive applications.

➤ An intelligent storage system consists of four key components Components of an Intelligent Storage System Front End.

➤ The front end provides the interface between the storage system and the host.

➤ It consists of two components: i. Front-End Ports ii. Front-End Controllers.

✓ Front End

✓ Cache

✓ Back end

✓ Physical disks.

➤ An I/O request received from the host at the front-end port is processed through cache and the back end, to enable storage and retrieval of data from the physical disk.

➤ A read request can be serviced directly from cache if the requested data is found in cache.

➤ In modern intelligent storage systems, front end, cache, and back end are typically integrated on a single board (referred to as a storage processor or storage controller).

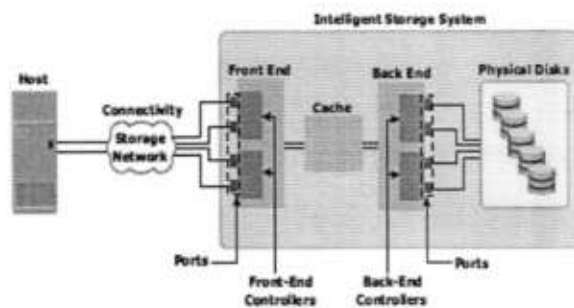


Fig:Components of an Intelligent Storage System.

1.Front End:

- The front end provides the interface between the storage system and the host.
- It consists of two components: front-end ports and front-end controllers.
- Typically, a front end has controllers for high availability, and each controller contains multiple ports that enable large numbers of hosts to connect to the intelligent storage system.
- Front-end controllers route data to and from cache via the internal data bus.
- When the cache receives the write data, the controller sends an acknowledgment message back to the host.

2.Cache:

- *Cache is semiconductor memory where data is placed temporarily to reduce the time required to service I/O requests from the host.*
- *Cache improves storage system performance by isolating hosts from the usual delays associated with rotating disks or hard disk drives (HDD).*
- *Rotating disks are the slowest component of an intelligent storage system.*
- *Data access on rotating disks usually takes several milliseconds because of seek time.*
- *Accessing data from cache is fast and typically takes less than a millisecond.*
- *On intelligent arrays, write data is first placed in cache and then written to disk.*

3.Back End:

- *The back end provides an interface between cache and the physical disks.*
- *It consists of two components: back-end ports and back-end controllers.*
- *The back-end controls data transfers between cache and the physical disks.*
- *From cache, data is sent to the back end and then routed to the destination disk.*
- *Physical disks are connected to **ports** on the back end.*
- *The back-end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage.*

4.Physical Disk:

- *Physical disks are connected to the back-end storage controller and provide stable data storage.*
- *Modern intelligent storage systems provide support to a variety of disk drives with different speeds and types, such as FC and flash drives.*

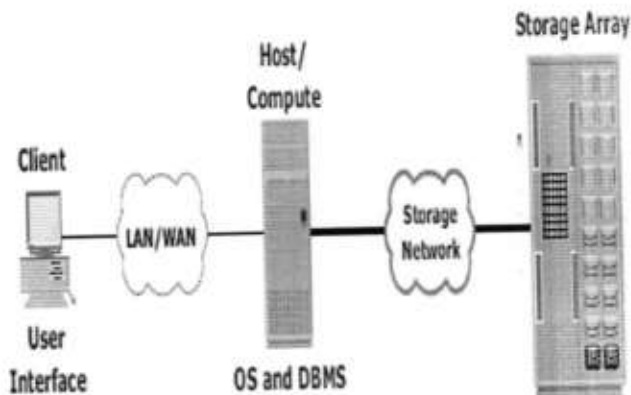
b. What is a data center? Explain core elements, key characteristics and key management activities of data center. $[2 + 3 + 3 + 2]$

The data center infrastructure includes hardware components, such as computers, storage systems, network devices, and power backups; and software components, such as applications, operating systems, and management software.

Core Elements of a Data Center:

Five core elements are essential for the functionality of a data center:

- **Application:** A computer program that provides the logic for computing operations
- **Database management system (DBMS):** Provides a structured way to store data in logically organized tables that are interrelated
- **Host or compute:** A computing platform (hardware, firmware, and software) that runs applications and databases.
- **Network:** A data path that facilitates communication among various networked devices.
- **Storage:** A device that stores data persistently for subsequent use.



Key Characteristics of a Data Center:

- **Availability:** A data center should ensure the availability of information when required.
- **Security:** Data centers must establish policies, procedures, and core element integration to prevent unauthorized access to information.
- **Scalability:** Business growth often requires deploying more servers, new applications, and additional databases. Data center resources should scale based on requirements, without interrupting business operations.
- **Performance:** All the elements of the data center should provide best performance based on the required service levels.
- **Data integrity:** Data integrity refers to mechanisms, such as error correction codes or which ensure that data is stored and retrieved exactly as it was received.
- **Capacity:** Data center operations require adequate resources to store and process large amounts of data, efficiently. When capacity requirements increase, the data center must provide additional capacity without interrupting availability or with minimal disruption. Capacity may be managed by reallocating the existing resources or by adding new resources.
- **Manageability:** A data center should provide easy and integrated management of all its elements. Manageability can be achieved through automation and reduction of human (manual) intervention in common tasks.



The key management activities include the following:

- **Monitoring:**
 - It is a continuous process of gathering information on various elements and services running in a data center.
 - The aspects of a data center that are monitored include security, performance, availability, and capacity.

➤ **Reporting:**

- ❑ It is done periodically on resource performance, capacity, and utilization.
- ❑ Reporting tasks help to establish business justification and chargeback of costs associated with data center operations.

➤ **Provisioning:**

- ❑ It is a process of providing the hardware, software, and other resources required to run a data center.
- ❑ Provisioning activities mainly include resources management to meet capacity, availability, performance, and security requirements.

3.a. Explain the following with appropriate diagram. i) Nested RAID ii) RAID 6 iii) RAID 0 [4 + 3 + 3]

i) Nested RAID:

- Most data centers require data redundancy and performance from their RAID arrays.
- RAID 1+0 and RAID 0+1 combine the performance benefits of RAID 0 with the redundancy benefits of RAID 1.
- They use striping and mirroring techniques and combine their benefits.
- These types of RAID require an even number of disks, the minimum being four as show in the following Figure.

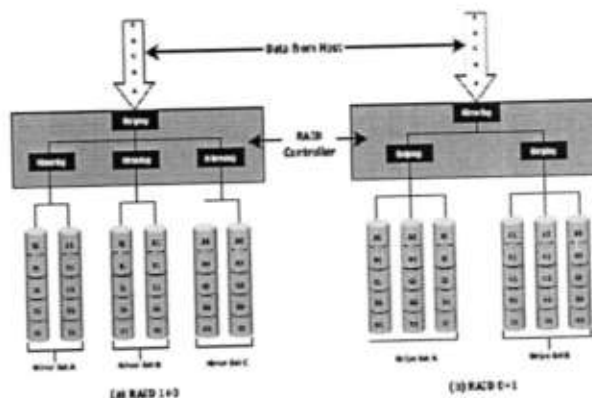


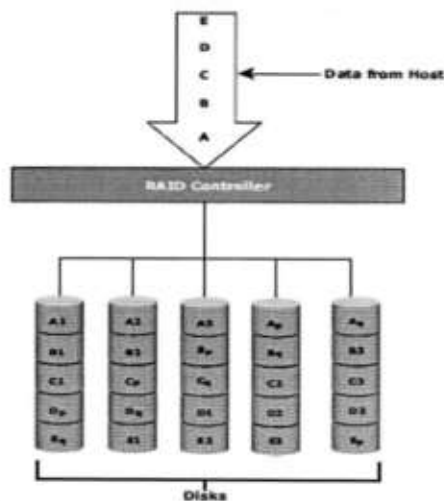
Figure 2-7: Nested RAID

- RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. Similarly, RAID 0+1 is also known as RAID 01 or RAID 0/1.
- Some applications that benefit from RAID 1+0 include the following:
 - ❖ High transaction rate Online Transaction Processing (OLTP)
 - ❖ Large messaging installations.
 - ❖ Database applications with random access workloads.
- ❖ RAID 1+0 is also called striped mirror. The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of the data are striped across multiple disk drive pairs in a RAID set.
- ❖ When replacing a failed drive, only the mirror is rebuilt.

- RAID 1+0, consider an example of six disks
- forming a RAID 1+0 (RAID 1 first and then RAID 0) set. These six disks are paired into three sets of two disks, where each set acts as a RAID 1 set (mirrored pair of disks).
- Data is then striped across all the three mirrored sets to form RAID 0.
- ✓ Following are the steps performed in RAID 1+0 Drives 1+2 = RAID 1 (Mirror Set A)
- ✓ Drives 3+4 = RAID 1 (Mirror Set B)
- ✓ Drives 5+6 = RAID 1 (Mirror Set C)
- ✓ RAID 0+1 is also called a mirrored stripe. The basic element of RAID 0+1 is a stripe.
- ✓ This means that the process of striping data across disk drives is performed initially, and then the entire stripe is mirrored.
- ✓ In this configuration if one drive fails, then the entire stripe is faulted.
- Consider the same example of six disks to understand the working of RAID 0+1.
- Here, six disks are paired into two sets of three disks each.
- Each of these sets, in turn, act as a RAID 0 set that contains three disks and then these two sets are mirrored to form RAID 1.
- ✓ Drives 1 + 2 + 3 = RAID 0 (Stripe Set A)
- ✓ Drives 4 + 5 + 6 = RAID 0 (Stripe Set B)

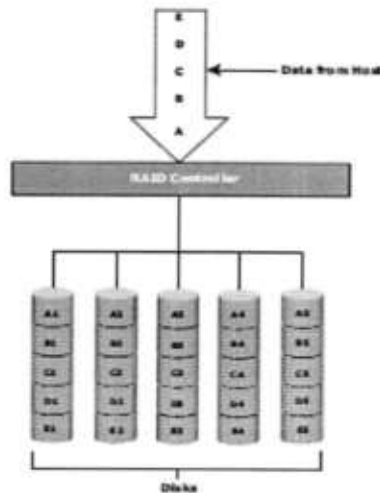
ii) RAID 6

- RAID 6 works the same way as RAID 5, except that RAID 6 includes a second parity element to enable survival if two disk failures occur in a RAID set.
- Advantage: The write penalty in RAID 6 is more than that in RAID 5 therefore, RAID 5 writes perform better than RAID 6.
- Disadvantage: The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.



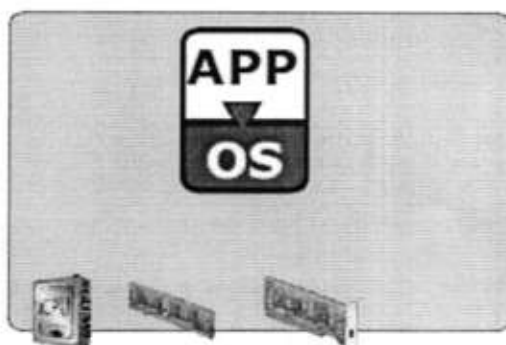
iii) RAID 0

- RAID 0 configuration uses data striping techniques, where data is striped across all the disks within a RAID set.
- Following Fig shows RAID 0 in an array in which data is striped across five disks.
- When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously.
- RAID 0 is a good option for applications that need high I/O throughput.
- If these applications require high availability during drive failures, RAID 0 does not provide data protection and availability.



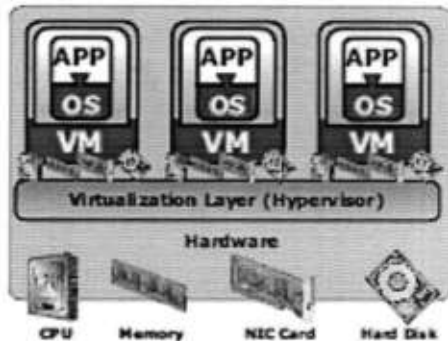
b. Explain compute virtualization. {5M}

- It enables multiple operating systems to run concurrently on single or clustered physical machines.
- This technique enables creating portable virtual compute systems called *virtual machines (VMs)*
- Each VM runs an operating system and application instance in an inaccessible manner.
- Compute virtualization is achieved by a virtualization layer that resides between the hardware and virtual machines. This layer is also called the *hypervisor*.
- The hypervisor provides hardware resources, such as CPU, memory, and network to all the virtual machines.
- Within a physical server, a large number of virtual machines can be created depending on the hardware capabilities of the physical server.
- Physical server often faces resource-conflict issues when two or more applications running on the server have conflicting requirements.
- These issues are further compounded with an application's high-availability requirements. As a result, the servers are limited to serve only one application at a time, as shown in the following Figure.



Before compute virtualization

- On the other hand, many applications do not take full advantage of the hardware capabilities available to them.
- Consequently, resources such as processors, memory, and storage remain underutilized. Compute virtualization enables users to overcome these challenges as show in the following Figure by allowing multiple operating systems and applications to run on a single physical machine.
- This technique significantly improves server utilization and provides server consolidation.



(b) After Compute Virtualization

c. Explain the following interface protocols (i) IDE/ATA & Serial ATA (ii) SCSI & Serial SCSI. [2.5 + 2.5]

(i) IDE/ATA & Serial ATA:

- IDE/ATA is a popular interface protocol standard used for connecting storage devices, such as disk drives and CD-ROM drives.
- This protocol supports parallel transmission and therefore is also known as Parallel ATA (PATA) or simply ATA.
- IDE/ATA has a variety of standards and names.
- The Ultra DMA/133 version of ATA supports a throughput of 133 MB per second.
- The serial version of this protocol supports single bit serial transmission and is known as Serial ATA (SATA).

(ii) SCSI & Serial SCSI:

- SCSI has emerged as a preferred connectivity protocol in high-end computers.
- This protocol supports parallel transmission and offers improved performance, scalability, and compatibility compared to ATA.
- SCSI supports up to 16 devices on a single bus and provides data transfer rates up to 640 MB/s.
- A newer version of serial SCSI (SAS 2.0) supports a data transfer rate up to 6 Gb/s.

OR

4. a. Explain any 2 techniques on the basis of which RAID levels are defined. [5 + 5]

RAID techniques:

1. striping
2. mirroring
3. parity

1 Striping:

- Striping is a technique to spread data across multiple drives (more than one) to use the drives in parallel.
- All the read-write heads work simultaneously, allowing more data to be processed in a shorter time and increasing performance, compared to reading and writing from a single disk.
- Within each disk in a RAID set, a predefined number of contiguously addressable disk blocks are defined as a *strip*.
- The set of aligned strips that spans across all the disks within the RAID set is called a *stripe*

- Strip size (also called stripe depth) describes the number of blocks in a strip and is the maximum amount of data that can be written to or read from a single disk in the set assuming that the accessed data starts at the beginning of the strip.
- All strips in a stripe have the same number of blocks.
- Having a smaller strip size means that data is broken into smaller pieces while spread across the disks.

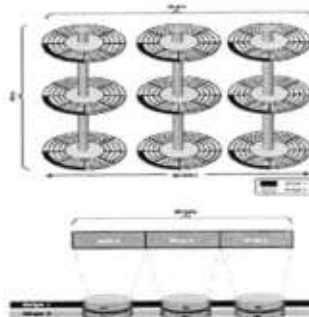


Figure 3-3: Striped RAID 0

2. Mirroring:

- *Mirroring is a technique whereby the same data is stored on two different disk drives, yielding two copies of the data. If one disk drive failure occurs, the data is intact on the surviving disk drive.*
- When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair.
- This activity is clear to the host.
- In addition to providing complete data redundancy, mirroring enables fast recovery from disk failure.
- ❖ Limitation:
- Mirroring involves duplication of data — the amount of storage capacity needed is twice the amount of data being stored. Therefore, mirroring is considered expensive
- Mirroring improves read performance because read requests can be serviced by both disks.
- However, write performance is slightly lower than that in a single disk because each write request manifests as two writes on the disk drives. Mirroring does not deliver the same levels of write performance as a striped RAID.

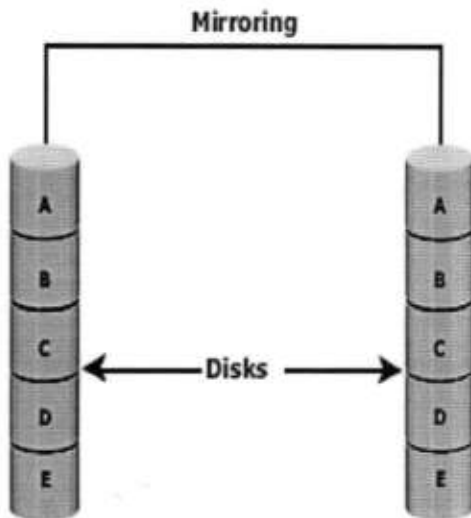


Figure 3-3: Mirrored disks in an array

3. Parity:

- Parity is a redundancy technique that ensures protection of data without maintaining a full set of duplicate data.
- Calculation of parity is a function of the RAID controller.
- Calculation of parity is a function of the RAID controller.

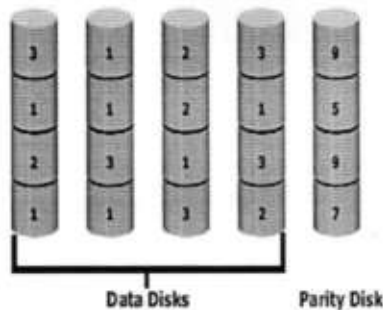


Figure 3-4: Parity RAID

b Explain virtualization & cloud computing. [2.5+2.5]

- Virtualization is a technique of abstracting physical resources, such as compute, storage, and network, and making them appear as logical resources.
- Common examples of virtualization are virtual memory used on compute systems and partitioning of raw disks.
- Virtualization enables group of physical resources and providing an combined view of the material resource capabilities.
- Virtual resources can be created and provisioned from the shared physical resources.

- In today's fast-paced and competitive environment, organizations must be agile and flexible to meet changing market requirements. This leads to rapid expansion and upgrade of resources while meeting budgets.
- **Cloud computing**, addresses these challenges efficiently.
- Cloud computing enables individuals or businesses to use IT resources as a service over the network.
- It provides highly scalable and flexible computing that enables provisioning of resources on demand.
- Users can scale up or scale down the demand of computing resources, including storage capacity, with minimal management effort or service provider interaction.
- Cloud computing enables consumption-based metering; therefore, consumers pay only for the resources they use, such as CPU hours used, amount of data transferred, and gigabytes of data stored.

c. Explain any 2 software components that are essential parts of a host system. [2.5+2.5].

Operating System

- In a traditional computing environment, an operating system controls all aspects of computing.
- It works between the application and the physical components of a compute system.
- In a virtualized compute environment, the virtualization layer works between the operating system and the hardware resources.

Functions of OS

- data access
- monitors and responds to user actions and the environment
- organizes and controls hardware components
- manages the allocation of hardware resources
- It provides basic security for the access and usage of all managed resources
- performs basic storage management tasks
- manages the file system, volume manager, and device drivers.

Memory Virtualization

- Memory has been, and continues to be, an expensive component of a host.
- It determines both the size and number of applications that can run on a host.
- Memory virtualization is an operating system feature that virtualizes the physical memory(RAM) of a host.
- It creates virtual memory with an address space larger than the physical memory space present in the compute system.
- The operating system utility that manages the virtual memory is known as the virtual memory manager (VMM).
- The space used by the VMM on the disk is known as a swap space.

> A swap space (also known as page file or swap file) is a portion of the disk drive that appears to be physical memory to the operating system.

> In a virtual memory implementation, the memory of a system is divided into contiguous blocks of fixed-size pages.

> A process known as paging moves inactive physical memory pages onto the swap file and brings them back to the physical memory when required.

Device Drivers

> A device driver is special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a disk drive.

Volume Manager

> In the early days, disk drives appeared to the operating system as a number of continuous disk blocks. The entire disk drive would be allocated to the file system or other data entity used by the operating system or application.

Disadvantages:

✓ lack of flexibility.

✓ When a disk drive ran out of space, there was no easy way to extend the file system's size.

✓ as the storage capacity of the disk drive increased, allocating the entire disk drive for the file system often resulted in underutilization of storage capacity
Solution: evolution of Logical Volume Managers (LVMs)

> LVM enabled dynamic extension of file system capacity and efficient storage management.

INTERNAL ASSESMENT TEST: II
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:10/06/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

1. a. Explain FC protocol stack with neat diagram. 7M|CO4
b. What is zoning? Explain the types of zoning with neat diagram. 7M |CO4
c. .Explain MTBF and MTTR. 6M|CO2

OR

2. a. Explain BC planning life cycle. 8M|CO5
b. single point of failure. Explain the idea to resolving single point of failure with neat diagram 9M|CO5
c. Define i) Accessibility ii) Reliability iii) Time liness 3M|CO5

3. a. With neat diagram explain Backup and Restore operations. 10M|CO2
b. List and explain Backup Topologies 10M|CO2

OR

4. a List and explain the methods of deduplication. 10M|CO2
b With neat diagram explain Traditional VM and Image-based backup. 10M|CO2

INTERNAL ASSESMENT TEST: II
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:10/06/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

- 1a. Explain FC protocol stack with neat diagram. 7M|CO4
b. What is zoning? Explain the types of zoning with neat diagram. 7M |CO4
c. .Explain MTBF and MTTR. 6M|CO2

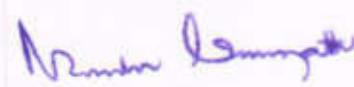
OR

2. a. Explain BC planning life cycle. 8M|CO5
b. single point of failure. Explain the idea to resolving single point of failure with neat diagram 9M|CO5
c. Define i) Accessibility ii) Reliability iii) Time liness 3M|CO5

3. a. With neat diagram explain Backup and Restore operations. 10M|CO2
b. List and explain Backup Topologies 10M|CO2

OR

4. a List and explain the methods of deduplication. 10M|CO2
b With neat diagram explain Traditional VM and Image-based backup. 10M|CO2


PRINCIPAL
SIET, TUMAKURU.

Internal Assessment Test - 2

Scheme

1. a. Diagram 2M
Explanation 5M

b. Zoning Def 2M
Types 5M
├─ Diagram 2M
└─ Explanation 3M

c. Explanation about MTBF and MTT_R, Each carry 3M

2. a. Explanation about five stages
First and Second Stage carry one mark
Remaining 3 stage carry 2M

b. Single point of failure Def 2M
Explanation about resolving 7M
├─ Diagram 3M
└─ Explanation 4M

c. Defin Each carry one mark

3. a. Diagram 5M
Explanation 5M

b. List 2M

Explanation 8M

↳ $4 \times 2 = 8M$ Diagram 2M
Explanation 2M

4. a. List 2M

Explanation 8M

b. Diagram 4M

Explanation 6M [3+3].

1. a. Explain FC protocol stack with neat diagram. {D 2 + E 5}

Fibre Channel Protocol Stack:

- FCP defines the communication protocol in five layers:
- FC-0 through FC-4 (except FC-3 layer, which is not implemented).
- In a layered communication model, the peer layers on each node talk to each other through defined protocols.

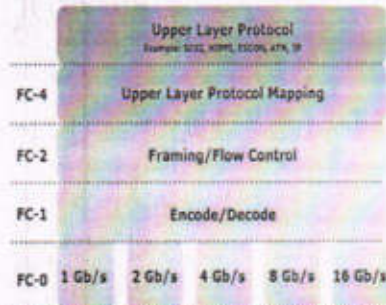


Figure 5-12: Fibre Channel protocol stack

- **FC-4** is the uppermost layer in the FCP stack.
- This layer defines the application interfaces and the way Upper Layer Protocols (ULPs) are mapped to the lower FC layers.
- The FC standard defines several protocols that can operate on the FC-4 layer.
- Some of the protocols include High Performance Parallel Interface (HIPPI) Framing Protocol, Enterprise Storage Connectivity (ESCON), Asynchronous Transfer Mode (ATM),
- **FC-2 Layer**
- The FC-2 layer provides Fibre Channel addressing, structure, and organization of data (frames, sequences, and exchanges).
- It also defines fabric services, classes of service, flow control, and routing.
- **FC-1 Layer**
- The FC-1 layer defines how data is encoded prior to transmission and decoded upon receipt.
- At the transmitter node, an 8-bit character is encoded into a 10-bit transmissions character. This character is then transmitted to the receiver node.
- At the receiver node, the 10-bit character is passed to the FC-1 layer, which decodes the 10-bit character into the original 8-bit character.
- FC links with speeds of 10 Gbps and above use 64-bit to 66-bit encoding algorithms.
- **FC-0 Layer**
- FC-0 is the lowest layer in the FCP stack. This layer defines the physical interface, media, and transmission of bits.
- The FC-0 specification includes cables, connect- tors, and optical and electrical parameters for a variety of data rates.

b. What is zoning? Explain the types of zoning with neat diagram. {D 2 + E 3}

Zoning is an FC switch function that enables node ports within the fabric to be logically segmented into groups and to communicate with each other within the group.

Types of zoning

Zoning can be categorized into three types:

1. Port zoning:

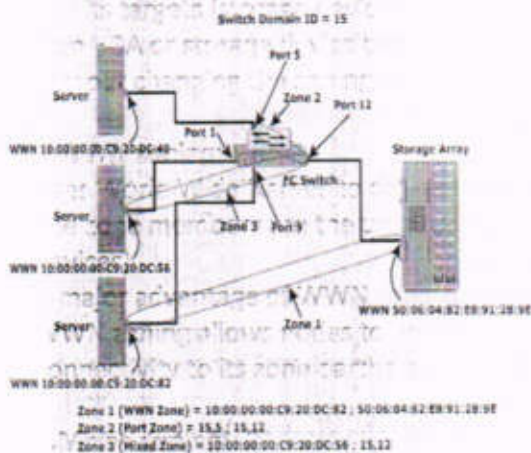
- Uses the physical address of switch ports to define zones.
- In port zoning, access to node is determined by the physical switch port to which a node is connected.
- The zone members are the port identifier (switch domain ID and port number) to which HBA and its targets (storage devices) are connected.
- If an HBA or storage device port fails, an administrator just has to replace the failed device without changing the zoning configuration.

2. WWN zoning:

- Uses World Wide Names to define zones.
- The zone members are the unique WWN addresses of the HBA and its targets (storage devices).
- A major advantage of WWN zoning is its flexibility.
- WWN zoning allows nodes to be moved to another switch port in the fabric and maintain connectivity to its zone partners without having to modify the zone configuration.

3. Mixed zoning:

- Combines the qualities of both WWN zoning and port zoning.
- Using mixed zoning enables a specific node port to be tied to the WWN of another node.



c. Explain MTBF and MTTR. [3+3]

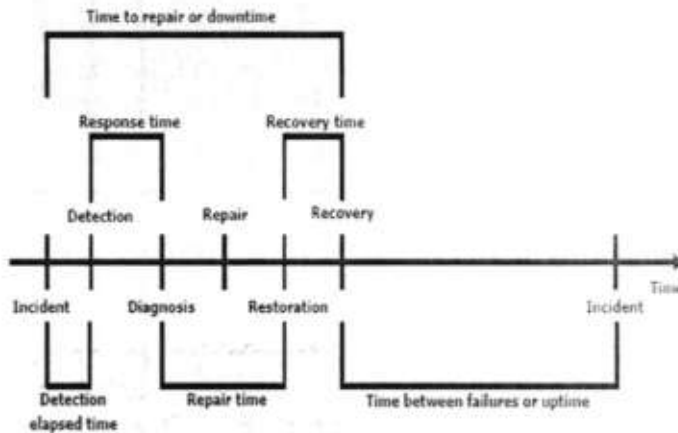
Mean Time Between Failure (MTBF):

- ❖ It is the average time available for a system or component to perform its normal operations between failures.
- ❖ It is the measure of system or component reliability and is usually expressed in hours.

Mean Time To Repair (MTTR):

- It is the average time required to repair a failed component.
- While calculating MTTR, it is assumed that the fault responsible for the failure is correctly identified and the required spares and personnel are available.

- A fault is a physical defect at the component level, which may result in information unavailability.
- MTTR includes the total time required to do the following activities:
- ❖ Detect the fault, the repairs team, diagnose the fault, obtain the spare parts, test, and restore the data.



- IA is the time period during which a system is in a condition to perform its intended function upon demand.
- It can be expressed in terms of system uptime and downtime and measured as the amount or percentage of system uptime:
- ❖ $IA = \frac{\text{system uptime}}{\text{system uptime} + \text{system downtime}}$
- Where
- system uptime is the period of time during which the system is in an accessible state;*
- when it is not accessible, it is termed as system downtime.*
- In terms of MTBF and MTTR, IA could also be expressed as
- $IA = \frac{MTBF}{MTBF + MTTR}$

OR

2. a. Explain BC planning life cycle. [8M]

- The conceptualization to the realization of the BC plan, a life cycle of activities can be defined for the BC process.
- The BC planning life cycle includes five stages
 1. Establishing objectives
 2. Analyzing
 3. Designing and developing
 4. Implementing
 5. Training, testing, assessing, and maintaining

1. Establish objectives:

- Determine BC requirements.
- Estimate the scope and budget to achieve requirements.
- Select a BC team that includes subject matter experts from all areas of the business, whether internal or external.

Create BC policies.

2. Analysis:

- Collect information on data profiles, business processes, infrastructure support, dependencies, and frequency of using business infrastructure.
- Conduct a Business Impact Analysis (BIA).
- Identify critical business processes and assign recovery priorities.
- Perform risk analysis for critical functions and create mitigation strategies.
- Perform cost benefit analysis for available solutions based on the mitigation strategy.
- Evaluate options.

3. Design and develop:

- Define the team structure and assign individual roles and responsibilities.
- Design data protection strategies and develop infrastructure.
- Develop emergency solutions.
- Develop emergency response procedures.
- Detail recovery and restart procedures.

4. Implement:

- Implement risk management and mitigation procedures that include backup, replication, and management of resources.
- Prepare the disaster recovery sites that can be utilized if a disaster affects the primary data center.
- Implement redundancy for every resource in a data center to avoid single points of failure.

5. Train, test, assess, and maintain:

- Train the employees who are responsible for backup and replication of business-critical data on a regular basis or whenever there is a modification in the BC plan.
- Train employees on emergency response procedures when disasters are declared.
- Train the recovery team on recovery procedures based on contingency scenarios.
- Perform damage-assessment processes and review recovery plans.
- Test the BC plan regularly to evaluate its performance and identify its limitations.
- Assess the performance reports and identify limitations.
- Update the BC plans and recovery/restart procedures to reflect regular changes within the data center.

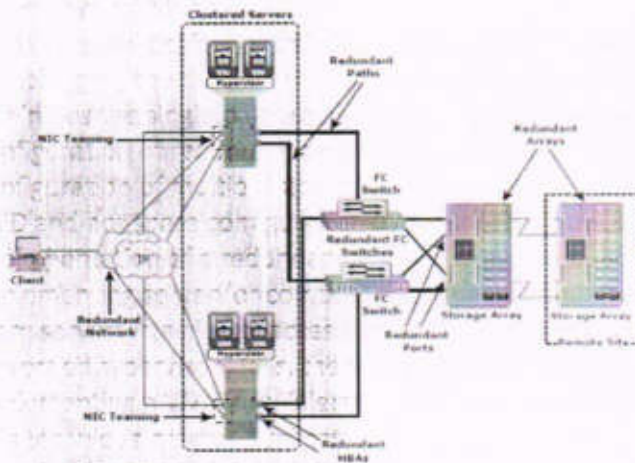
b. single point of failure. Explain the idea to resolving single point of failure with neat diagram. [2M + 7M]

A single point of failure refers to the failure of a component that can terminate the availability of the entire system or IT service.

Resolving Single Points of Failure:

- To mitigate single points of failure, systems are designed with redundancy, such that the system fails only if all the components in the redundancy group fail.
- This ensures that the failure of a single component does not affect data availability.
- Data centers follow stringent guidelines to implement fault tolerance for uninterrupted information availability.
- Careful analysis is performed to eliminate every single point of failure.

- Configuration of redundant HBAs at a server to mitigate single HBA failure.
- Configuration of NIC (Network Interface Card) teaming at a server allows protection against single physical NIC failure. It allows grouping of two or more physical NICs and treating them as a single logical device. With NIC teaming, if one of the underlying physical NICs fails or its cable is unplugged, the traffic is redirected to another physical NIC in the team. Thus, NIC teaming eliminates the single point of failure associated with a single physical NIC
- Configuration of redundant switches to account for a switch failure.
- Configuration of multiple storage array ports to mitigate a port failure.
- RAID and hot spare configuration to ensure continuous operation in the event of disk failure.
- Implementation of a redundant storage array at a remote site to mitigate local site failure.
- Implementing server (or compute) clustering, a fault-tolerance mechanism whereby two or more servers in a cluster access the same set of data volumes. Clustered servers If one of the servers or hypervisors fails, the other server or hypervisor can take up the workload.
- Implementing a VM Fault Tolerance mechanism ensures BC in the event of a server failure. This technique creates duplicate copies of each VM on another server so that when a VM failure is detected, the duplicate VM can be used for failover. The two VMs are kept in synchronization with each other in order to perform successful failover.



c. Define i) Accessibility ii) Reliability iii) Time liness [3M] [3x1]

i) Accessibility

Information should be accessible at the right place, to the right user.

ii) Reliability

Information should be reliable and correct in all aspects. It is "the same" as what was stored, and there is no alteration or corruption to the information.

iii) Time liness

Defines the exact moment or the time window (a particular time of the day, week, month, and year as specified) during which information must be accessible.

3. a. With neat diagram explain Backup and Restore operations. [5+5]

- Backup is an additional copy of production data, created and retained for the individual purpose of recovering lost or corrupted data.
- Hot backup and cold backup are the two methods set up for a backup.

- *Hot backup, the application is up-and-running, with users accessing their data during the backup process.*
- *This method of backup is also referred to as an **online backup**.*
- *A **cold backup** requires the application to be shut down during the backup process.*
- *Hence, this method is also referred to as an **offline backup**.*
- *The hot backup of online production data is challenging because data is actively used and changed. If a file is open, it is normally not backed up during the backup process.*
- *In such situations, an **open file agent** is required to back up the open file.*
- *These agents interact directly with the operating system or application and enable the creation of consistent copies of open files.*
- **Disadvantage:** Associated with a hot backup is that the agents usually affect the overall application performance.
- *Consistent backups of databases can also be done by using a cold backup.*
- *This requires the database to remain inactive during the backup.*
- **Disadvantage :** cold backup is that the database is inaccessible to users during the backup process.
- **A point-in-time (PIT) copy** method is deployed in environments in which the impact of downtime from a cold backup or the performance impact resulting from a hot backup is unacceptable.
- *The PIT copy is created from the production volume and used as the source for the backup. This reduces the impact on the production volume.*
- *To ensure consistency, it is not enough to back up only the production data for recovery.*
- *Certain attributes and properties attached to a file, such as permissions, owner, and other metadata, also need to be backed up.*
- *These attributes are as important as the data itself and must be backed up for consistency.*
- **Bare-metal recovery (BMR)** refers to a backup in which all metadata, system information, and application configurations are appropriately backed up for a full system recovery
- *BMR builds the base system, which includes partitioning, the file system layout, the operating system, the applications, and all the relevant configurations.*
- *BMR recovers the base system first before starting the recovery of data files.*

Backup and Restore Operations:

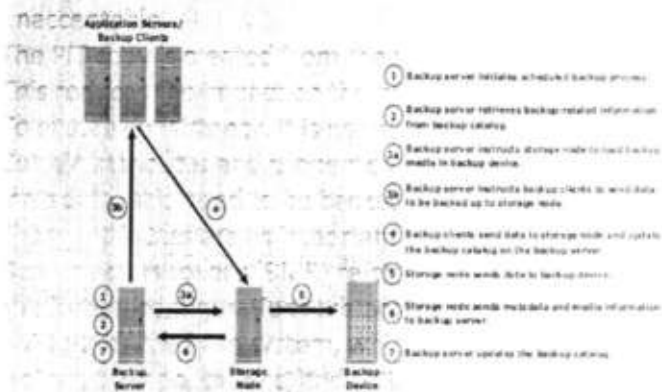


Figure 10-5: Backup operation

b. List and explain Backup Topologies. [48 2M + EXP 8M]

Three basic topologies are used in a backup environment

- 1 Direct-attached backup
- 2 LAN-based backup
- 3 SAN-based backup

4. Mixed topology is also used by combining LAN-based and SAN-based topologies.

1 Direct-attached backup

- As the environment grows, there will be a need for centralized management and sharing of backup devices to optimize costs.
- An appropriate solution is required to share the backup devices among multiple servers.

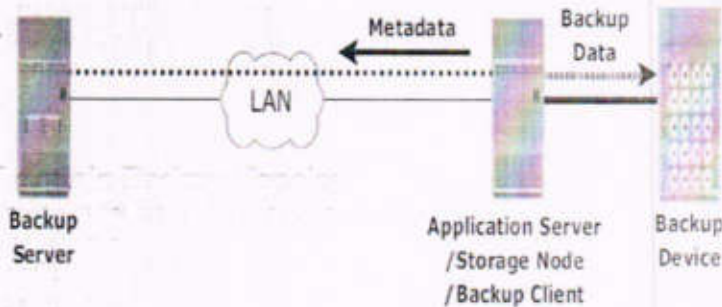


Figure 10-7: Direct-attached backup topology

2 LAN-based backup

- The data to be backed up is transferred from the backup client (source) to the backup device (destination) over the LAN, which might affect network performance.

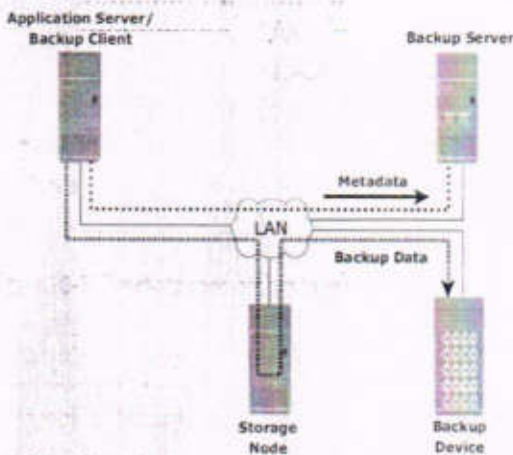


Figure 10-8: LAN-based backup topology

3 SAN-based backup

- The backup data traffic is restricted to the SAN, and only the backup metadata is transported over the LAN.
- The volume of metadata is insignificant when compared to the manufacture data; the LAN performance is not degraded in this configuration.
- The emergence of low-cost disks as a backup medium has enabled disk arrays to be attached to the SAN and used as backup devices.
- A tape backup of these data backups on the disks can be created and shipped offsite for disaster recovery and long-term retention.

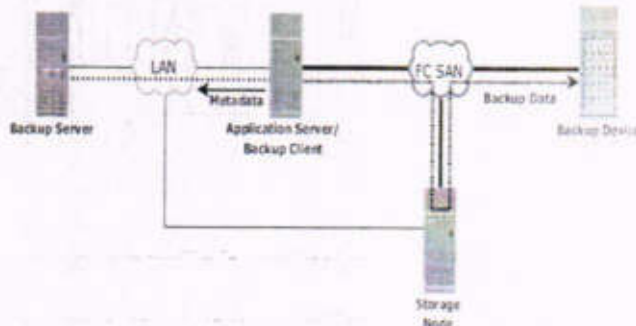


Figure 10-9: SAN-based backup topology

4. Mixed topology is also used by combining LAN-based and SAN-based topologies.

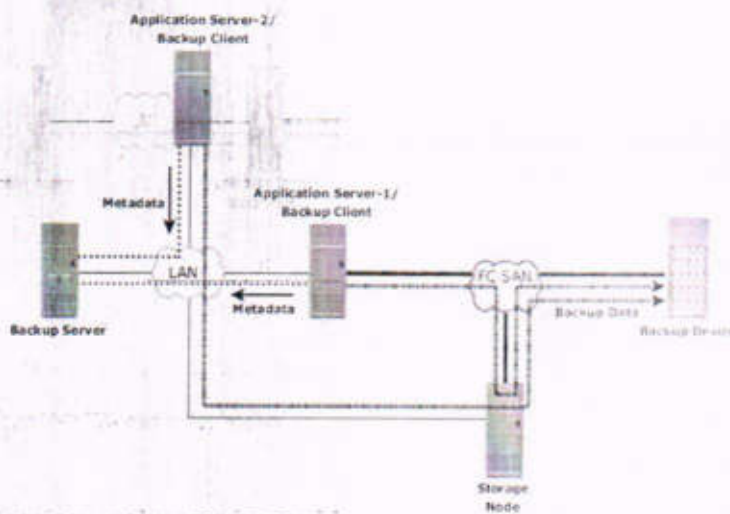


Figure 10-10: Mixed backup topology

- This topology might be implemented for several reasons,
- Including cost,
- Server location,
- Performance considerations.

OR

4.a List and explain the methods of deduplication. [List 2M + Exp 8M]

Data Deduplication Methods:

- There are two methods of deduplication:
- File level.
- Subfile level.

File level.

- File-level deduplication (also called single-instance storage) detects and removes redundant copies of identical files.
- It enables storing only one copy of the file the subsequent copies are replaced with a pointer that points to the original file.
- File-level deduplication is simple and fast but does not address the problem of duplicate content inside the files.

Subfile level.

- Subfile deduplication breaks the file into smaller chunks and then uses a specialized algorithm to detect redundant data within and across the file.
- As a result, subfile deduplication eliminates duplicate data across files.
- There are two forms of subfile deduplication:
 - 1 Fixed-length block
 - 2 Variable-length segments.

1. The *fixed-length block deduplication* divides the files into fixed length blocks and uses a hash algorithm to find the duplicate data.

- Although simple in design, fixed-length blocks might miss many opportunities to discover redundant data because the block boundary of similar data might be different.

2. In *variable-length segment deduplication*, if there is a change in the segment, the boundary for only that segment is adjusted, leaving the remaining segments unchanged.

- This method vastly improves the ability to find duplicate data segments compared to fixed-block.

b With neat diagram explain Traditional VM and Image-based backup. [D4 + EXP 6]

- There are two approaches for performing a backup in a virtualized environment:
- ✓ The traditional backup approach and
- ✓ The image-based backup approach.

➤ In the *traditional backup approach*, a backup agent is installed either on the virtual machine (VM) or on the hypervisor.

- If the backup agent is installed on a VM, the VM appears as a physical server to the agent.
- The backup agent installed on the VM backs up devices.

- The agent does not capture VM files, such as the virtual BIOS file, VM swap file, logs, and configuration files.
- Therefore, for a VM restore, a user needs to manually re-create the VM and then restore data onto it.
- If the backup agent is installed on the hypervisor, the VMs appear as a set of files to the agent.
- So, VM files can be backed up by performing a file system backup from a hypervisor.

Disadvantage:

- The traditional backup method can cause high CPU utilization on the server being backed up.
- the backup should be performed when the server resources are idle or during a low activity period on the network.



Figure 10-21: Traditional VM backup

- *Image-based backup operates at the hypervisor level and essentially takes a snapshot of the VM.*
- It creates a copy of the guest OS and all the data associated with it (snapshot of VM disk files), including the VM state and application configurations.
- The backup is saved as a single file called an "image," and this image is mounted on the separate physical machine—proxy server, which acts as a backup client.
- This effectively loads the backup processing from the hypervisor and transfers the load on the proxy server, thereby reducing the impact to VMs running on the hypervisor.
- Image-based backup enables quick restoration of a VM.

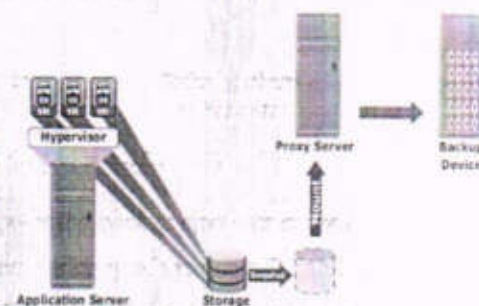


Figure 10-22: image-based backup



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: III
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:01/07/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

1. a. With a neat diagram explain iSCSI Topologies . 8M[CO3]
b. With a neat diagram explain FCIP protocol attack and Encapsulation. 8M [CO3]
c. .Define NAS. Explain its components. 4M[CO3>
- OR
2. a. With a neat diagram explain iSCSI protocol stack . 8M[CO3]
b. Define FCOE. Explain its key components. 5M[CO3]
c. Explain i)CIFS ii)NFS 7M[CO3>
3. a. With neat diagram explain storage security domains and threats in an application access domain. 10M[CO6]
b. With neat diagram explain FC SAN security architecture and its protection strategies. 10M[CO6>
- OR
4. a List and explain local replication Technologies. 10M[CO6]
b Explain modes of Remote Replication. 10M[CO6]



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: III
SUB : Storage Area Network (18CS822)
SEM : VIII

DATE:01/07/2022
MAX MARKS :30
TIME : 75 min

Note: Answer Two full Questions

1. a. With a neat diagram explain iSCSI Topologies . 8M[CO3]
b. With a neat diagram explain FCIP protocol attack and Encapsulation. 8M [CO3]
c. .Define NAS. Explain its components. 4M[CO3>
- OR
2. a. With a neat diagram explain iSCSI protocol stack . 8M[CO3]
b. Define FCOE. Explain its key components. 5M[CO3]
c. Explain i)CIFS ii)NFS 7M[CO3>
3. a. With neat diagram explain storage security domains and threats in an application access domain. 10M[CO6]
b. With neat diagram explain FC SAN security architecture and its protection strategies. 10M[CO6>
- OR
4. a List and explain local replication Technologies. 10M[CO6]
b Explain modes of Remote Replication. 10M[CO6]


PRINCIPAL
SIET., TUMAKURU.

Internal Assessment Test-3

Scheme

1. a. Diagram 4M

Explanation 4M

b. Explanation about FCIP protocol 4M

Diagram 2M

Explanation 2M

and Encapsulation
Each carry

c. Det 1M

Explanation 3M

└ Diagram 1M

└ Explanation 2M

2. a. Diagram 4M

Explanation 4M

b. Det 1M

Explanation 4M

c. Explanation CIFS & NFS Each carry 3.5M.

3. a. Explanation about Storage Security domains
and threats in an application Each carry 5M

└ Diagram 2.5M

└ Explanation 2.5M

b. Explanation about FC SAN and Protection Strategies Each Carry 5M

- Diagram 2.5M
- Explanation 2.5M

4. a. List 2M

Diagram 4M

Explanation 4M

b. Explanation about Synchronous and Asynchronous each Carry 5M

- Diagram 2.5M
- Explanation 2.5M

1. a. With a neat diagram explain iSCSI Topologies .

iSCSI Topologies

- Two topologies of iSCSI implementations are native and bridged.
- Native topology does not have FC components.
- The initiators may be either directly attached to targets or connected through the IP network.
- Bridged topology enables the coexistence of FC with IP by providing iSCSI-to-FC bridging functionality. For example, the initiators can exist in an IP environment while the storage remains in an FC environment.

Native iSCSI Connectivity

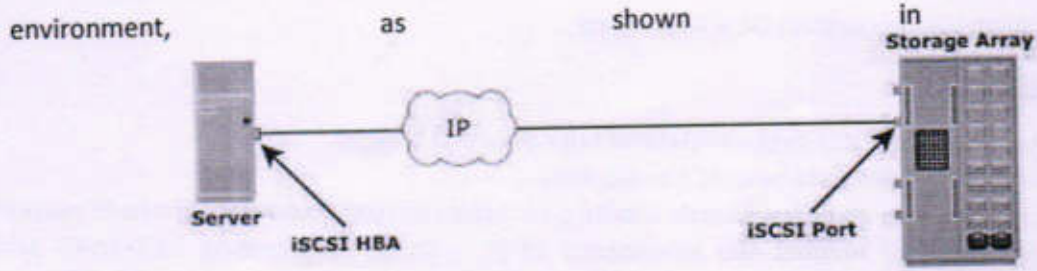
- FC components are not required for iSCSI connectivity if an iSCSI-enabled array is deployed.
- In Fig (a), the array has one or more iSCSI ports configured with an IP address and is connected to a standard Ethernet switch.
- After an initiator is logged on to the network, it can access the available LUNs on the storage array.
- A single array port can service multiple hosts or initiators as long as the array port can handle the amount of storage traffic that the hosts generate.

Bridged iSCSI Connectivity

- A bridged iSCSI implementation includes FC components in its configuration.
- Fig (b), illustrates iSCSI host connectivity to an FC storage array. In this case, the array does not have any iSCSI ports. Therefore, an external device, called a gateway or a multiprotocol router, must be used to facilitate the communication between the iSCSI host and FC storage.
- The gateway converts IP packets to FC frames and vice versa.
- The bridge devices contain both FC and Ethernet ports to facilitate the communication between the FC and IP environments.
- In a bridged iSCSI implementation, the iSCSI initiator is configured with the gateway's IP address as its target destination.
- On the other side, the gateway is configured as an FC initiator to the storage array.

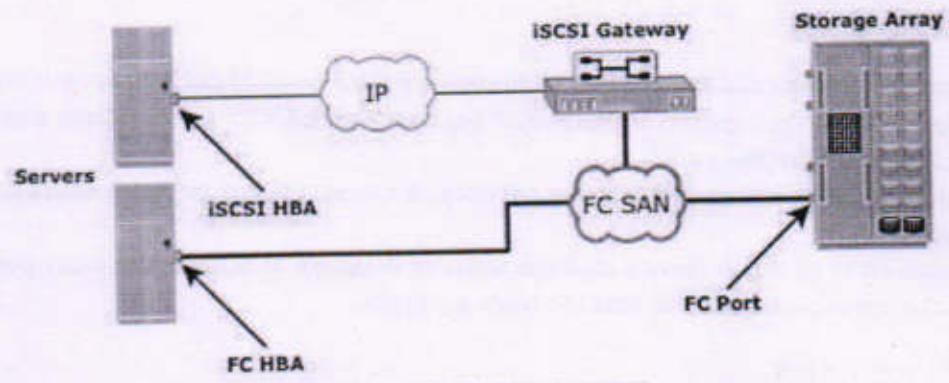
Combining FC and Native iSCSI Connectivity:

- The most common topology is a combination of FC and native iSCSI. Typically, a storage array comes with both FC and iSCSI ports that enable iSCSI and FC connectivity in the same

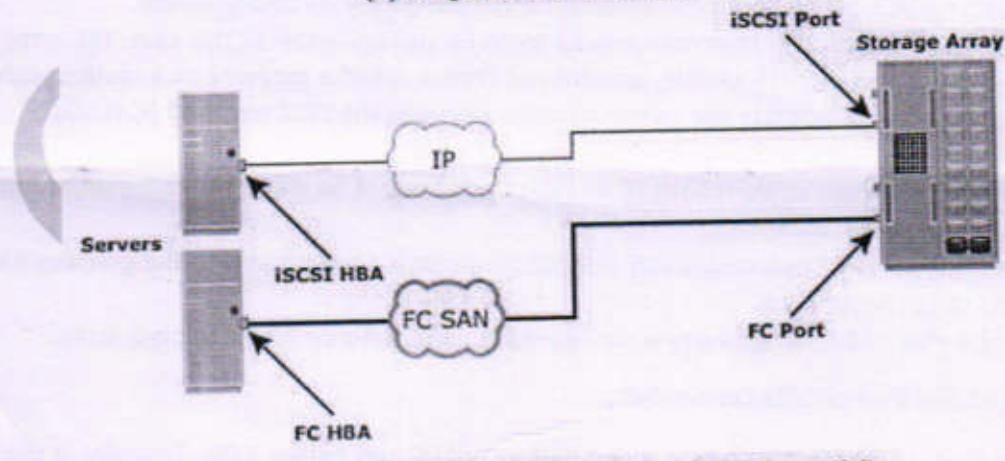


Fig

(a) Native iSCSI Connectivity



(b) Bridged iSCSI Connectivity



(c) Combining FC and Native iSCSI Connectivity

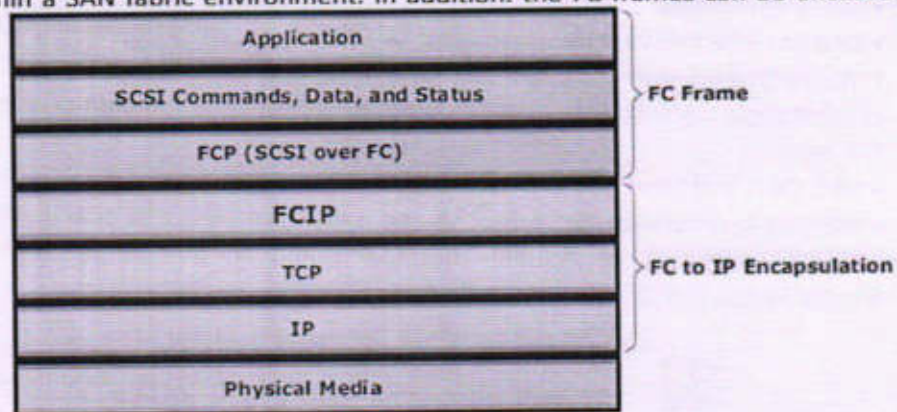
(c).

b. With a neat diagram explain FCIP protocol attack and Encapsulation.

FCIP Protocol Stack

- The FCIP protocol stack is shown in Fig below. Applications generate SCSI commands and data, which are processed by various layers of the protocol stack.
- The upper layer protocol SCSI includes the SCSI driver program that executes the read-andwrite commands.
- Below the SCSI layer is the Fibre Channel Protocol (FCP) layer, which is simply a Fibre Channel frame whose payload is SCSI.

- The FCP layer rides on top of the Fibre Channel transport layer. This enables the FC frames to run natively within a SAN fabric environment. In addition, the FC frames can be encapsulated into the IP packet



- The FCIP layer encapsulates the Fibre Channel frames onto the IP payload and passes them to the TCP layer (see Fig). TCP and IP are used for transporting the encapsulated information across Ethernet, wireless, or other media that support the TCP/IP traffic.
- Encapsulation of FC frame into an IP packet could cause the IP packet to be fragmented when the data link cannot support the maximum transmission unit (MTU) size of an IP packet.
- When an IP packet is fragmented, the required parts of the header must be copied by all fragments.
- When a TCP packet is segmented, normal TCP operations are responsible for receiving and resequencing the data prior to passing it on to the FC processing portion of the device.

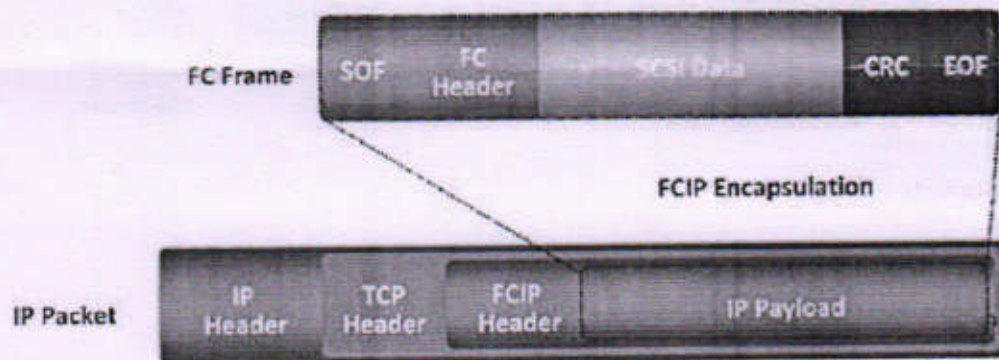


Fig FCIP encapsulation

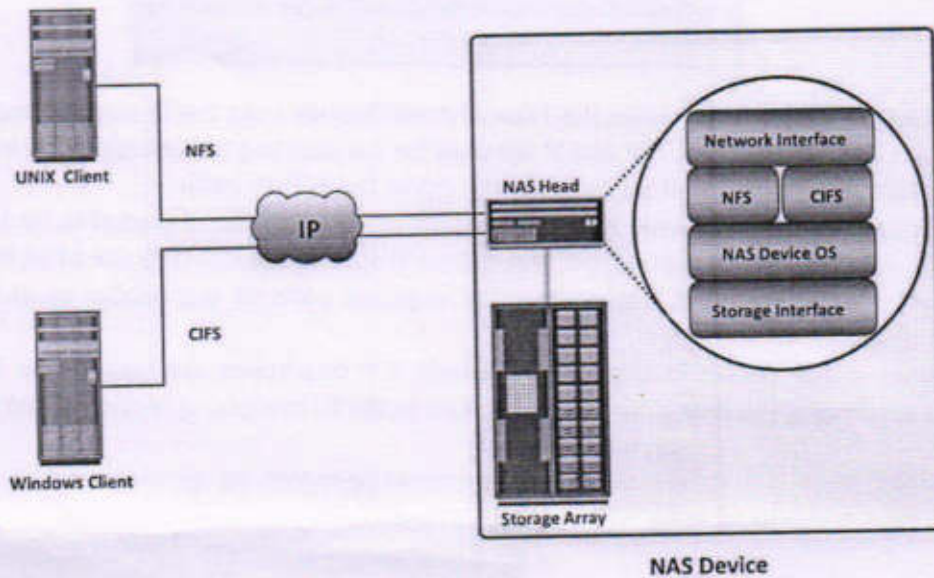
c. Define NAS. Explain its components.

- NAS is an IP based dedicated, high-performance file sharing and storage device.
- Enables NAS clients to share files over an IP network. Uses network and file-sharing protocols to provide access to the file data.
- Ex: Common Internet File System (CIFS) and Network File System (NFS).

Components of NAS

- NAS device has two key components (as shown in Fig 2.33): NAS head and storage.
- In some NAS implementations, the storage could be external to the NAS device and shared with other hosts.

- NAS head includes the following components
 - CPU and memory
 - One or more network interface cards (NICs), which provide connectivity to the client network.
 - An optimized operating system for managing the NAS functionality. It translates filelevel requests into block-storage requests and further converts the data supplied at the block level to file data
 - NFS, CIFS, and other protocols for file sharing.
 - Industry-standard storage protocols and ports to connect and manage physical disk resources
- The NAS environment includes clients accessing a NAS device over an IP network using filesharing protocols.



Components of NAS

OR

2. a. With a neat diagram explain iSCSI protocol stack .

iSCSI Protocol Stack

Fig 2.23 displays a model of the iSCSI protocol layers and depicts the encapsulation order of the SCSI commands for their delivery through a physical carrier.

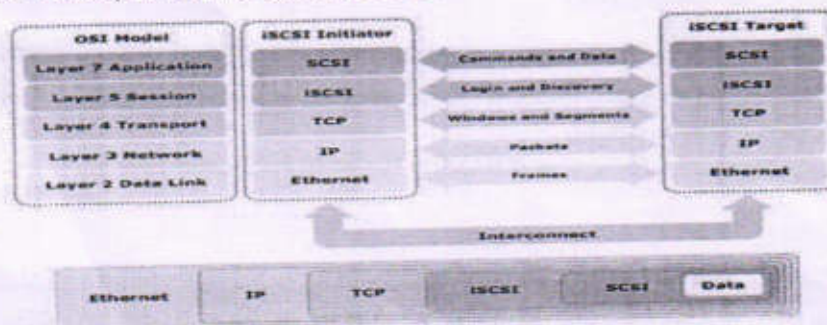


Fig 2.23: iSCSI protocol stack

- SCSI is the command protocol that works at the application layer of the Open System Interconnection (OSI) model.
- The initiators and targets use SCSI commands and responses to talk to each other.
- The SCSI command descriptor blocks, data, and status messages are encapsulated into TCP/IP and transmitted across the network between the initiators and targets.
- iSCSI is the session-layer protocol that initiates a reliable session between devices that recognize SCSI commands and TCP/IP.
- The iSCSI session-layer interface is responsible for handling login, authentication, target discovery, and session management.
- TCP is used with iSCSI at the transport layer to provide reliable transmission.
- TCP controls message flow, windowing, error recovery, and retransmission.
- It relies upon the network layer of the OSI model to provide global addressing and connectivity.
- The Layer 2 protocols at the data link layer of this model enable node-to-node communication through a physical network.

b. Define FCOE. Explain its key components.

FCoE (Fibre Channel over Ethernet)

- Data centers typically have multiple networks to handle various types of I/O traffic — foreexample, an Ethernet network for TCP/IP communication and an FC network for FC communication.
- TCP/IP is typically used for client-server communication, data backup, infrastructure management communication, and so on.
- FC is typically used for moving block-level data between storage and servers.

The key components of FCOE are :

1. Converged Network Adaptors(CNA)
2. Cables
3. FCOE Switches

Converged Network Adaptors(CNA)

- A CNA provides the functionality of both a standard NIC and an FC HBA in a single adapter and consolidates both types of traffic. CNA eliminates the need to deploy separate adapters and cables for FC and Ethernet communications, thereby reducing the required number of server slots and switch ports.
- As shown in Fig below, a CNA contains separate modules for 10 Gigabit Ethernet, Fibre Channel, and FCoE Application Specific Integrated Circuits (ASICs). The FCoE ASIC encapsulates FC frames into Ethernet frames. One end of this ASIC is connected to 10GbE and FC ASICs for server connectivity, while the other end provides a 10GbE interface to connect to an FCoE switch.

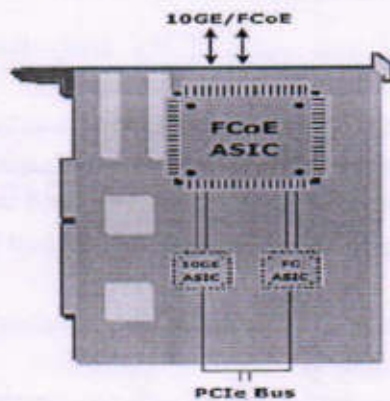


Fig : Converged Network Adapter

Cables

There are two options available for FCoE cabling:

1. Copper based Twinax.
2. standard fiber optical cables.
 - A Twinax cable is composed of two pairs of copper cables covered with a shielded casing. The Twinax cable can transmit data at the speed of 10 Gbps over shorter distances up to 10 meters. Twinax cables require less power and are less expensive than fiber optic cables.
 - The Small Form Factor Pluggable Plus (SFP+) connector is the primary connector used for FCoE links and can be used with both optical and copper cables.

FCoE Switches

An FCoE switch has both Ethernet switch and Fibre Channel switch functionalities.

As shown in Fig below, FCoE switch consists of:

1. Fibre Channel Forwarder (FCF),
2. Ethernet Bridge,
3. set of Ethernet ports
4. optional FC ports
 - The function of the FCF is to encapsulate the FC frames, received from the FC port, into the FCoE frames and also to de-encapsulate the FCoE frames, received from the Ethernet Bridge, to the FC frames.
 - Upon receiving the incoming traffic, the FCoE switch inspects the Ether type (used to indicate which protocol is encapsulated in the payload of an Ethernet frame) of the incoming frames and uses that to determine the destination.

- If the Ether type of the frame is FCoE, the switch recognizes that the frame contains an FC payload and forwards it to the FCF. From there, the FC is extracted from the FCoE frame and transmitted to FC SAN over the FC ports.
- If the Ether type is not FCoE, the switch handles the traffic as usual Ethernet traffic and forwards it over the Ethernet ports.

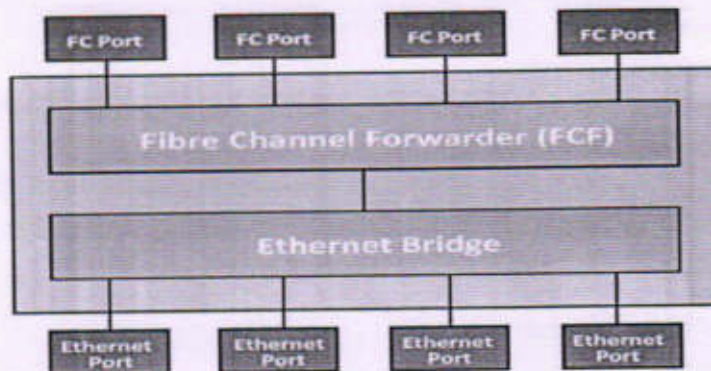


Fig: FCoE switch generic architecture

c. Explain i)CIFS ii)NFS

i)Common Internet File System (CIFS)

- CIFS is a client-server application protocol
- It enables clients to access files and services on remote computers over TCP/IP.
- It is a public, or open, variation of Server Message Block (SMB) protocol.
- It provides following features to ensure data integrity:
- It uses file and record locking to prevent users from overwriting the work of another user on a file or a record.
- It supports fault tolerance and can automatically restore connections and reopen files that were open prior to an interruption. This feature depends on whether an application is written to take advantage of this.
- CIFS is a stateful protocol because the CIFS server maintains connection information regarding every connected client. If a network failure or CIFS server failure occurs, the client receives a disconnection notification. User disruption is minimized if the application has the embedded intelligence to restore the connection. However, if the embedded intelligence is missing, the user must take steps to reestablish the CIFS connection.
- Users refer to remote file systems with an easy-to-use file-naming scheme:
- Eg: \\server\share or \\servername.domain.suffix\share.

ii) Network File System (NFS)

- NFS is a client-server protocol for file sharing that is commonly used on UNIX systems.
- NFS was originally based on the connectionless User Datagram Protocol (UDP).

- It uses Remote Procedure Call (RPC) as a method of inter-process communication between two computers.
- The NFS protocol provides a set of RPCs to access a remote file system for the following operations:
 - Searching files and directories
 - Opening, reading, writing to, and closing a file
 - Changing file attributes
 - Modifying file links and directories
- NFS creates a connection between the client and the remote system to transfer data.
- NFSv3 and earlier is a stateless protocol .
- It does not maintain any kind of table to store information about open files and associated pointers. Each call provides a full set of arguments - a file handle, a particular position to read or write, and the versions of NFS - to access files on the server .

Currently, three versions of NFS are in use:

1. NFS version 2 (NFSv2): Uses UDP to provide a stateless network connection between a client and a server. Features, such as locking, are handled outside the protocol.

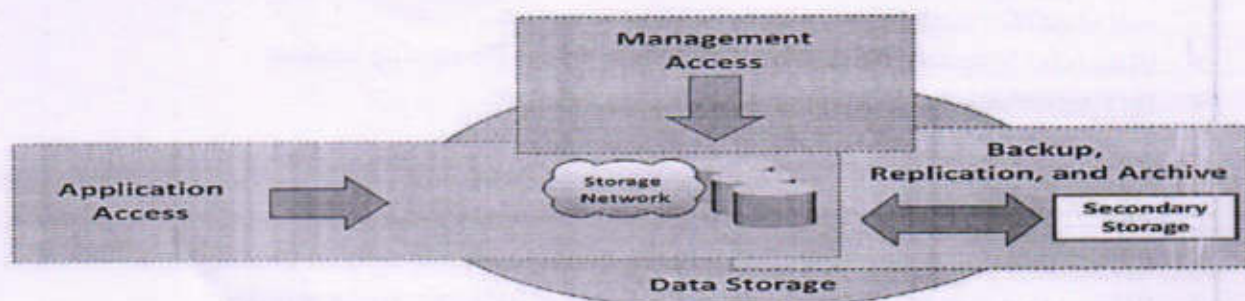
2. NFS version 3 (NFSv3): Uses UDP or TCP, and is based on the stateless protocol design. It includes some new features, such as a 64-bit file size, asynchronous writes, and additional file attributes to reduce refetching.

3. NFS version 4 (NFSv4): Uses TCP and is based on a stateful protocol design. It offers enhanced security. The latest NFS version 4.1 is the enhancement of NFSv4 and includes some new features, such as session model, parallel NFS (pNFS), and data retention.

3.a. With neat diagram explain storage security domains and threats in an application access domain.

Storage Security Domains

- To identify the threats that apply to a storage network, access paths to data storage can be categorized into three security domains: application access, management access, and backup, replication, and archive.
- Fig 5.1 depicts the three security domains of a storage system environment.

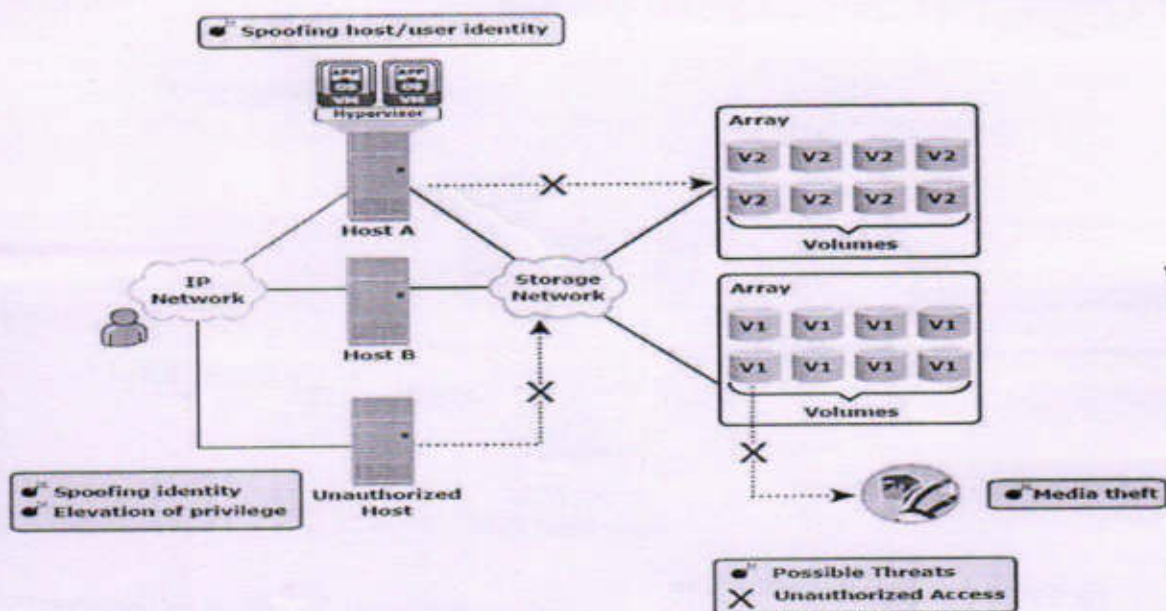


Storage security domains

- The first security domain involves application access to the stored data through the storage network.
- The second security domain includes management access to storage and interconnect devices and to the data residing on those devices. This domain is primarily accessed by storage administrators who configure and manage the environment.
- The third domain consists of backup, replication, and archive access. Along with the access points in this domain, the backup media also needs to be secured.

Securing the Application Access Domain

- The application access domain may include only those applications that access the data through the file system or a database interface.
- An important step to secure the application access domain is to identify the threats in the environment and appropriate controls that should be applied.
- Implementing physical security is also an important consideration to prevent media theft.
- Fig 5.2 shows application access in a storage networking environment.



Security threats in an application access domain

- Host A can access all V1 volumes; host B can access all V2 volumes. These volumes are classified according to the access level, such as confidential, restricted, and public.
- Some of the possible threats in this scenario could be host A spoofing the identity or elevating to the privileges of host B to gain access to host B's resources. Another threat could be that an unauthorized host gains access to the network; the attacker on this host may try to spoof the identity of another host and tamper with the data, snoop the network, or execute a DoS attack.
- Also any form of media theft could also compromise security. These threats can pose several serious challenges to the network security; therefore, they need to be addressed.

b. With neat diagram explain FC SAN security architecture and its protection strategies

FC-SAN

- Traditional FC SANs have an inherent security advantage over IP-based networks.
- An FC SAN is configured as an isolated private environment with fewer nodes than an IP network.

FC SAN Security Architecture

- Storage networking environments are a potential target for unauthorized access, theft, and misuse because of the vastness and complexity of these environments. Therefore, security strategies are based on the **defense in depth** concept, which recommends multiple integrated layers of security. This ensures that the failure of one security control will not compromise the assets under protection.
- Fig 5.5 illustrates various levels (zones) of a storage networking environment that must be secured and the security measures that can be deployed.

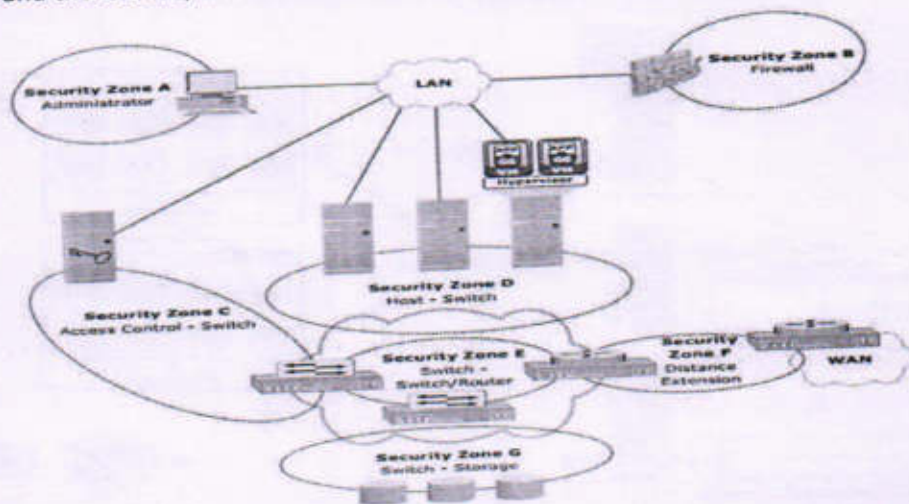


Fig 5.5: FC SAN security architecture

- Table 5.1 provides a comprehensive list of protection strategies that must be implemented in various security zones. Some of the security mechanisms listed in Table 5.1 are not specific to

SAN but are commonly used data center techniques. For example, two-factor authentication is implemented widely; in a simple implementation it requires the use of a username/password and an additional security component such as a smart card for authentication.

Table 5.1 : list of protection strategies

SECURITY ZONES	PROTECTION STRATEGIES
Zone A (Authentication at the Management Console)	(a) Restrict management LAN access to authorized users (lock down MAC addresses); (b) implement VPN tunneling for secure remote access to the management LAN; and (c) use two-factor authentication for network access.
Zone B (Firewall)	Block inappropriate traffic by (a) filtering out addresses that should not be allowed on your LAN; and (b) screening for allowable protocols, block ports that are not in use.
Zone C (Access Control-Switch)	Authenticate users/administrators of FC switches using Remote Authentication Dial In User Service (RADIUS), DH-CHAP (Diffie-Hellman Challenge Handshake Authentication Protocol), and so on.

SECURITY ZONES	PROTECTION STRATEGIES
Zone D (Host to switch)	Restrict Fabric access to legitimate hosts by (a) implementing ACLs: Known HBAs can connect on specific switch ports only; and (b) implementing a secure zoning method, such as port zoning (also known as hard zoning).
Zone E (Switch to Switch/Switch to Router)	Protect traffic on fabric by (a) using E_Port authentication; (b) encrypting the traffic in transit; and (c) implementing FC switch controls and port controls.
Zone F (Distance Extension)	Implement encryption for in-flight data (a) FC-SP for long-distance FC extension; and (b) IPsec for SAN extension via FCIP.
Zone G (Switch to Storage)	Protect the storage arrays on your SAN via (a) WWPN-based LUN masking; and (b) S_ID locking: masking based on source FC address.

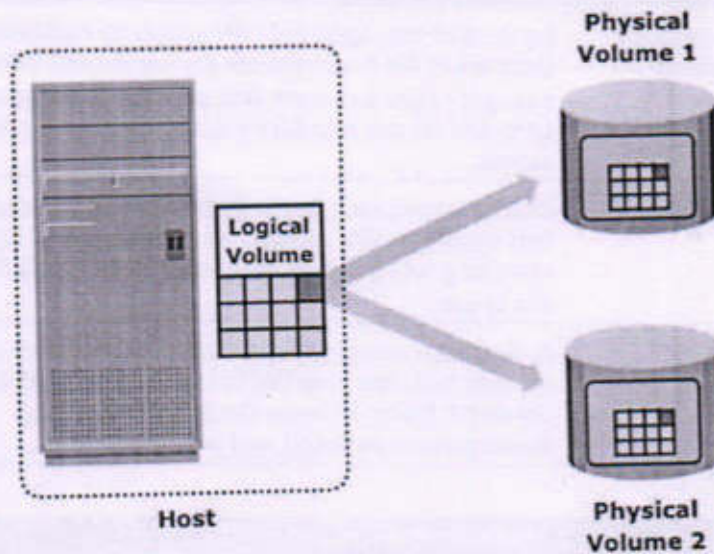
OR

4. a List and explain local replication Technologies.

- Host based
 - Logical Volume Manager (LVM) based replication (LVM mirroring)
 - File System Snapshot
- Storage Array based

- Full volume mirroring
- Pointer based full volume replication
- Pointer based virtual replication

Host Based Replication: LVM-based Replication

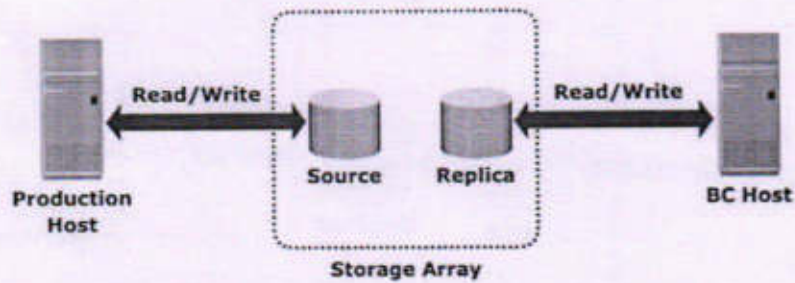


In LVM-based replication, each logical partition in a logical volume is mapped to two physical partitions on two different physical volumes, as shown in Figure.

- An application writes to a logical partition is written to the two physical partitions by the LVM device driver. This is also known as LVM mirroring. Mirrors can be split and the data contained therein can be independently accessed. LVM mirrors can be added or removed dynamically.

Storage Array Based Local Replication

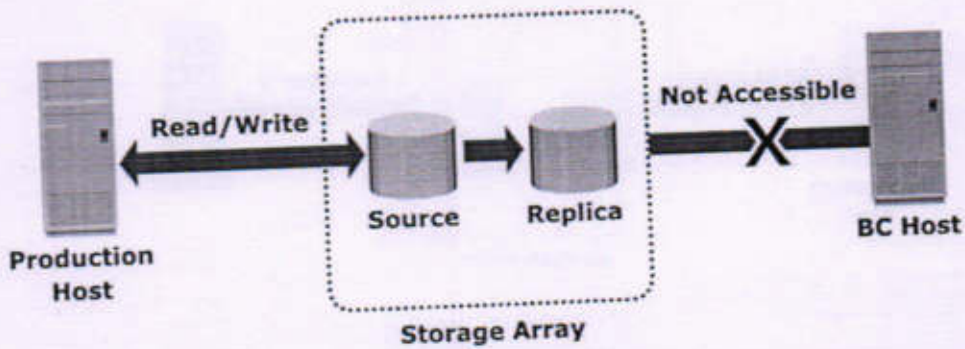
- Replication performed by the Array Operating Environment
- Replicas are on the same array
- Types of array based replication
 - Full-volume mirroring
 - Pointer-based full-volume replication
 - Pointer-based virtual replication



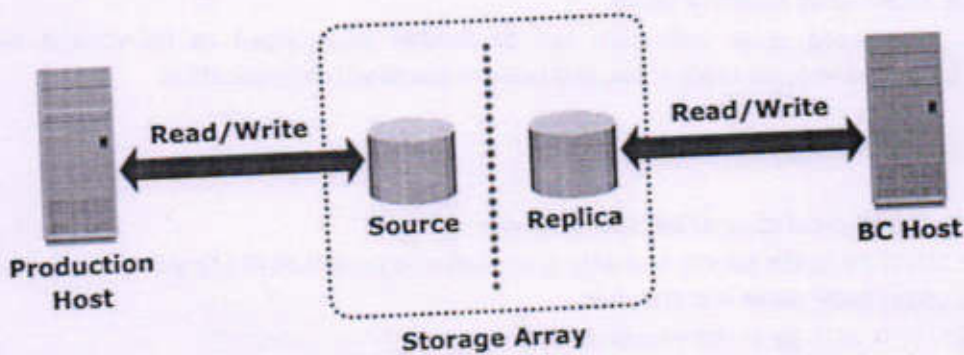
- The figure shows storage array-based local replication, where source and target are in the same array and accessed by different hosts.
- Storage array-based local replication can be further categorized as full-volume mirroring, pointer-based full-volume replication, and pointer-based virtual replication.

Full Volume Mirroring: Attached

- Target is a full physical copy of the source device
- Target is attached to the source and data from source is copied to the target
- Target is unavailable while it is attached
- Target device is as large as the source device
- Good for full backup, decision support, development, testing and restore to last PIT



(a) Full volume mirroring with source attached to replica



(b) Full volume mirroring with source detached from replica

- In full-volume mirroring, the target is attached to the source and established as a mirror of the source (Figure [a]). Existing data on the source is copied to the target. New updates to the source are also updated on the target. After all the data is copied and both the source and the target contain identical data, the target can be considered a mirror of the source.
- While the target is attached to the source and the synchronization is taking place, the target remains unavailable to any other host. However, the production host can access the source.
- After synchronization is complete, the target can be detached from the source and is made available for BC operations. Figure (b) shows full-volume mirroring when the target is detached from the source.

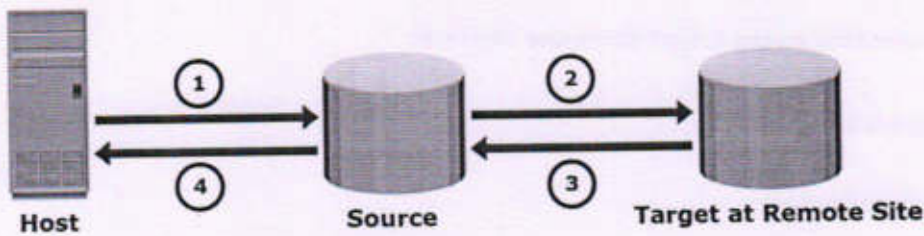
b Explain modes of Remote Replication.

The two basic modes of remote replication are synchronous and asynchronous.

Synchronous replication

- In synchronous remote replication, writes must be committed to the source and the target, prior to acknowledging "write complete" to the host.
- Additional writes on the source cannot occur until each preceding write has been completed and acknowledged. This ensures that data is identical on the source and the replica at all times.

- Hence, write ordering is maintained. In the event of a failure of the source site, synchronous remote replication provides zero or near-zero RPO, as well as the lowest RTO. However, application response time is increased with any synchronous remote replication.

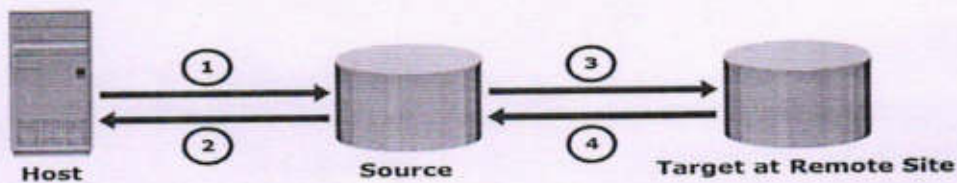


- Host writes data to source
- Data from source is replicated to target at remote site
- Target acknowledges back to source
- Source acknowledges write complete to host

Figure :Synchronous replication

Asynchronous replication

- In *asynchronous remote replication*, a write is committed to the source and immediately acknowledged to the host. Data is buffered at the source and transmitted to the remote site later.
- This eliminates the impact to the application's response time.
- Data at the remote site will be behind the source by at least the size of the buffer.
- Hence, asynchronous remote replication provides a finite (nonzero) RPO disaster recovery solution.
- RPO depends on the size of the buffer, available network bandwidth, and the write workload to the source.
- There is no impact on application response time, as the writes are acknowledged immediately to the source host. This enables deployment of asynchronous replication over extended distances.
- Asynchronous remote replication can be deployed over distances ranging from several hundred to several thousand kilometers between two sites.



- ① Host writes data to source
- ② Write is immediately acknowledged to host
- ③ Data is transmitted to the target at remote site later
- ④ Target acknowledges back to source

Figure : Asynchronous replication

Check List

1. Appointment Order
2. Calendar Of Events
3. Class Time Table
4. Individual Time Table
5. Approved Student List
6. Syllabus Copy
7. Lesson Plan
8. Internal Question Paper With Scheme Of Valuation
9. University Question Paper
10. Notes According to Syllabus
11. Assignment Questions
12. Teaching Aid : PPT, Demo, Case Studies, Self Study Topics


PRINCIPAL
SIET, TUMAKURU

Office Order

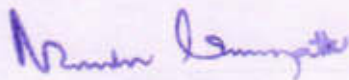
Prof. Renukaradhya P C is Appointed as a Course Instructor for the following courses .

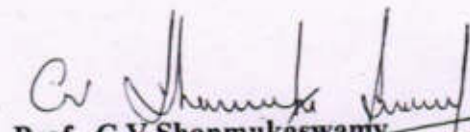
Sl. No	Course Name	Course Code	Remarks
1	UNIX and Shell Programming	17CS35	
2	Web Technology and its applications	15CS71	
3	Web Technology Laboratory with mini project	15CSL77	

6. Additional to above you are appointed as counselor for Batch - A4 of 7th sem students mentioned in the table below. You are requested to counsel the students once in fortnight periodically and submit the report.

Batch -A4:

Roll No.	USN	Name
35	1SV15CS051	Nethravathi B S
36	1SV15CS053	Nischitha K
37	1SV15CS054	Nissy Mathias
38	1SV15CS055	Pallavi Gangadhar Shet
39	1SV15CS056	Pavan K P
40	1SV15CS057	Pooja B K
41	1SV15CS058	Pooja C M
42	1SV15CS059	Prarthana M G
43	1SV15CS060	Prathibha B T
44	1SV15CS061	Punyashree B
45	1SV15CS062	Raghunath H P
46	1SV15CS063	Ramya M K
47	1SV15CS065	Rekha A
48	1SV15CS066	Shamanth H S
49	1SV15CS067	Sharon Biju George
50	1SV15CS086	Shilpa R


PRINCIPAL
SIET., TUMAKURU.


Prof. C V Shanmugaswamy
HOD, CSE
H.O.D.
Dept. of C.S. & E.,
SIET., TUMKUR-6. 3/17/18



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY, TUMAKURU-572 106
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
CALENDAR OF EVENTS

III & V Semester B E August 2018 to November 2018 Session



AUGUST			SEPTEMBER			OCTOBER			NOVEMBER		
Date	Day	Activities	Date	Day	Activities	Date	Day	Activities	Date	Day	Activities
* 1/8/18	Wed	Commencement	1/9/18	Sat		1/10/18	Mon	HODs Meeting	1/11/18	Thu	Kannada Rajyotsava
2/8/18	Tue	Technical Talk	2/9/18	Sun		2/10/18	Tue	Gandhi Jayanthi	2/11/18	Fri	
3/8/18	Fri		3/9/18	Mon	HODs Meeting	* 3/10/18	Wed	Deptl Staff Meeting	3/11/18	Sat	
4/8/18	Sat		4/9/18	Tue	Deptl Staff Meeting	4/10/18	Thu	Debate	4/11/18	Sun	
5/8/18	Sun		* 5/9/18	Wed		5/10/18	Fri		5/11/18	Mon	HODs Meeting
6/8/18	Mon	# HODs Meeting	6/9/18	Thu	Technical Talk	6/10/18	Sat		6/11/18	Tue	Naraka Chaturdashi
7/8/18	Tue	\$ Deptl Staff Meeting	7/9/18	Fri		7/10/18	Sun		* 7/11/18	Wed	Deptl Staff Meeting
* 8/8/18	Wed		8/9/18	Sat		8/10/18	Mon	Mahalaya Amavasye	8/11/18	Thu	Balipadyami
9/8/18	Thu	Debate	9/9/18	Sun		9/10/18	Tue	HODs Meeting	9/11/18	Fri	
10/8/18	Fri		10/9/18	Mon	HODs Meeting	* 10/10/18	Wed	Deptl Staff Meeting	10/11/18	Sat	
11/8/18	Sat		11/9/18	Tue	Deptl Staff Meeting	11/10/18	Thu	C,C++&Java Test	11/11/18	Sun	
12/8/18	Sun		* 12/9/18	Wed		12/10/18	Fri		12/11/18	Mon	HODs Meeting
13/8/18	Mon	HODs Meeting	13/9/18	Thu	Vinayaka Chaturthi	13/10/18	Sat		13/11/18	Tue	Deptl Staff Meeting
14/8/18	Tue	Deptl Staff Meeting	14/9/18	Fri		14/10/18	Sun		* 14/11/18	Wed	
* 15/8/18	Wed	Independence Day	15/9/18	Sat		15/10/18	Mon	HODs Meeting	15/11/18	Thu	C,C++&Java Test
16/8/18	Thu	Group Discussion	16/9/18	Sun		16/10/18	Tue	Deptl Staff Meeting	16/11/18	Fri	
17/8/18	Fri		17/9/18	Mon	HODs Meeting	* 17/10/18	Wed		17/11/18	Sat	
18/8/18	Sat		18/9/18	Tue	Deptl Staff Meeting	18/10/18	Thu	Ayudha Pooja	18/11/18	Sun	
19/8/18	Sun		* 19/9/18	Wed		19/10/18	Fri	Vijayadashami	19/11/18	Mon	HODs Meeting
20/8/18	Mon	HODs Meeting	20/9/18	Thu	IA Test I	20/10/18	Sat		20/11/18	Tue	Deptl Staff Meeting
21/8/18	Tue	Deptl Staff Meeting	21/9/18	Fri	Muharram	21/10/18	Sun		21/11/18	Wed	Id Milad
* 22/8/18	Wed	Bakrid	22/9/18	Sat	IA Test I	22/10/18	Mon	HODs Meeting	22/11/18	Thu	Prize Distribution
23/8/18	Thu	Extempore	23/9/18	Sun		23/10/18	Tue	Deptl Staff Meeting	23/11/18	Fri	
24/8/18	Fri		24/9/18	Mon	# IA Test I	* 24/10/18	Wed	Valmiki Jayanthi	24/11/18	Sat	
25/8/18	Sat		25/9/18	Tue	Deptl Staff Meeting	25/10/18	Thu	Extempore	25/11/18	Sun	
26/8/18	Sun		* 26/9/18	Wed		26/10/18	Fri		26/11/18	Mon	Kanaka Jayanthi
27/8/18	Mon	HODs Meeting	27/9/18	Thu	Expert Lecture	27/10/18	Sat		27/11/18	Tue	# IA Test III
28/8/18	Tue	Deptl Staff Meeting	28/9/18	Fri		28/10/18	Sun		28/11/18	Wed	\$ IA Test III
* 29/8/18	Wed		29/9/18	Sat		29/10/18	Mon	# IA Test II	29/11/18	Thu	IA Test III
30/8/18	Thu	Paper Presentation	30/9/18	Sun		30/10/18	Tue	\$ IA Test II	30/11/18	Fri	Closure
31/8/18	Fri					* 31/10/18	Wed	IA Test II	Total No of Working Days: 91		
No of Working Days: 25			No of Working Days: 23			No of Working Days: 22			No of Working Days: 21		
Practical Examinations: 13/12/2018			Practical Examinations: 13/12/2018			Practical Examinations: 17/12/2018			Practical Examinations: 17/12/2018		

[Signature]
 [Prof. C V Shanmugaswamy]
 Head of the Department

PRINCIPAL
 SIET, TUMAKURU.

[Signature]
 [Dr T Hemadri Naidu]

SHRIDEVI INSTITUTE OF ENGINEERING AND TECHNOLOGY, TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE & ENGG
 Academic year 2018-2019, ODD Semester



CLASS TIME TABLE

W.E.F: 01/08/2018
Lecture Hall:107

Class: III Sem

TIME DAY	8:30 A.M To 9:25 A.M	9:25 A.M To 10:20 A.M	10:20 A.M To 10:40 A.M	10:40 A.M To 11:35 A.M	11:35 A.M To 12:30 P.M	12:30P.M To 1:30 P.M	1:30 P.M To 2:25 P.M	2:25 P.M To 3:20 P.M	3:20 P.M To 4:15 P.M
MONDAY	DSC	CO	TEA BREAK	M3	USP	LUNCH BREAK	Tutorial	Tutorial	Tutorial
TUESDAY	ADE	DMS		CO	M3		Tutorial	Tutorial	Councilling
WEDNESDAY	CO	DSC		ADE(...A2...)(HLM+MSC) DS(...A1...)(CDG+YHS)					
THURSDAY	USP	DMS		ADE	DSC		M3	CO	KANNADIGAS
FRIDAY	M3	ADE		DSC	CO		DMS	USP	NON- KANNADIGAS
SATURDAY	ADE(...A1..)(HLM+MSC) DS(...A2...)(YHS+VT)				USP				

COURSE CODE

NAME OF THE COURSE

NAME OF THE STAFF

COUNSELOR

17MAT31 Engineering Mathematics -III
 17CS32 Analog & Digital Electronics
 17CS33 Data Structure and Application
 17CS34 Computer Organization
 17CS35 Unix and Shell Programming
 17CS36 Discrete Mathematical Structure
 17CSL37 Analog & Digital Electronics Laboratory
 17CSL38 Data Structure Laboratory

CC: Mrs.Chetana .C
 HLM: Mr. Mallesh H L
 YHS:Ms.Yahaswini H S
 MSC: Mr. Chethan M S
 RPC: Mr. Renukaradhya P C
 RRN: Mr. Raghunandan R
 Mr. Mallesh H L(HLM)+ Mr. Chethan M S(MSC)
 YHS:Ms.Yahaswini H S + CDG: Dr.Guruprakash C D + VT:Ms.Trupthi V

C1: Ms.Swetha K. H(CT*)
 C2: Mr. Chethan M S
 (CT*) Class Teacher

[Mr. Suthan R.]

Time Table Co-Ordinator

[Prof. C V Shanmukaswamy]
Head , Dept. of CSE

PRINCIPAL
 SIET., TUMKURU.

[Dr. Hemadri Naidu T]
Principal



SHRIDEVI INSTITUTE OF ENGINEERING AND TECHNOLOGY, TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE & ENGG
Academic year 2018-19, ODD Semester



TIME TABLE

W.E.F: 01/08/18

Faculty Name: Mr. Renukaradhya P C

TIME \ DAY	8:30 A.M To 9:25 A.M	9:25 A.M To 10:20 A.M	10:20 A.M To 10:40 A.M	10:40 A.M To 11:35 A.M	11:35 A.M To 12:30 P.M	12:30 P.M To 1:30 P.M	1:30 P.M To 2:25 P.M	2:25 P.M To 3:20 P.M	3:20 P.M To 4:15 P.M
MONDAY	WTA		TEA BREAK		USP	L U N C H B R E A K			
TUESDAY	WTA(...A2...)(RPC+VT)						WTA		
WEDNESDAY		WTA	TEA BREAK						
THURSDAY	USP								
FRIDAY	WTA(...A1...)(RPC+SR)							USP	
SATURDAY	WTA		TEA BREAK		USP				

<u>COURSE CODE</u>	<u>NAME OF THE COURSE</u>	<u>WORK LOADS IN UNITS</u>	<u>TOTAL WORKLOAD</u>
17CS35	UNIX and Shell Programming	08	26
15CS71	Web Technology and its applications	08	
15CSL77	Web Technology Laboratory with mini project	06	
15CPS78	PWP: Project work phase-II	02	
	Department Activity	02	

Staff Signature

[Mr. Suthan R]
Time Table Co-Ordinator

PRINCIPAL
SIET., TUMAKURU.

[Prof. C V Shanmukaswamy]
Head , Dept. Of CSE

SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY

Sira Road, TUMKUR-572106

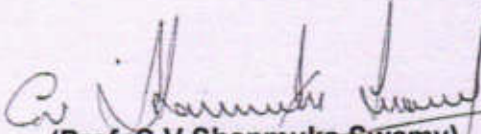
Dept. of Computer Science & Engg

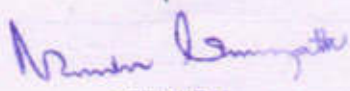
**STUDENT LIST FOR THE ACADEMIC YEAR 2018-19(ODD Sem)****CLASS: III SEM**

Roll No.	USN	Name
1	ISV15CS070	Priya Panda
2	ISV15CS0	Mohammed Agha
3	ISV17CS001	Abhishek Kumar Prasad
4	ISV17CS002	Abhishek Pandey
5	ISV17CS003	Aishwarya Mery E
6	ISV17CS004	Aman Prasad Kalwar
7	ISV17CS005	Anil Gowda
8	ISV17CS006	Anupriya Singh
9	ISV17CS007	Arun S Mattihalli
10	ISV17CS008	B Ramesh
11	ISV17CS009	Bhoomika M
12	ISV17CS010	Bli Dago Evariste
13	ISV17CS011	Chaithra M S
14	ISV17CS012	Chandana D Gowda
15	ISV17CS013	Chethan D
16	ISV17CS014	Eva Regmi
17	ISV17CS015	Gaganashree T U
18	ISV17CS016	Harshitha B A
19	ISV17CS017	Harshitha K
20	ISV17CS018	Junaid Ulla Khan
21	ISV17CS019	Kavya H S
22	ISV17CS020	Kavyashree Bk
23	ISV17CS021	Krupankh D N
24	ISV17CS023	Manasa N R
25	ISV17CS024	Manasa V
26	ISV17CS025	Mayank Sinha
27	ISV17CS026	Nanditha
28	ISV17CS027	Navya S
29	ISV17CS028	Nawaz Khan K N
30	ISV17CS029	Nidhi Anand
31	ISV17CS030	Nikesh Kumar Tiwari
32	ISV17CS031	Noor Asfiya
33	ISV17CS032	Prathamagowda Y P
34	ISV17CS033	Pruthviraj R B
35	ISV17CS034	Raghuram
36	ISV17CS035	Rajesh Kumar Kahar
37	ISV17CS036	Sabha Khanum

PRINCIPAL
SIET, TUMAKURU.

38	ISV17CS037	Sadanand Kumar
39	ISV17CS038	Saurabh Pandey
40	ISV17CS039	Tezashree Pokharel
41	ISV17CS040	Udaya
42	ISV17CS041	Vidya C M
43	ISV17CS042	Vijay Kumar Jha


(Prof. C V Shanmuka Swamy) / 1/8/18
HOD, CSE
H.O.D.
Dept. of C.S. & E.,
SIET., TUMKUR-6.


PRINCIPAL
SIET., TUMAKURU.



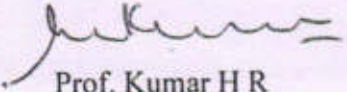
Shridevi Institute of Engineering and Technology
Tumakuru-572106
(An ISO 9001-2015 Certified Institution)



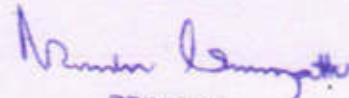
DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

3rd Semester Students List during 2018-19 Odd Semesters

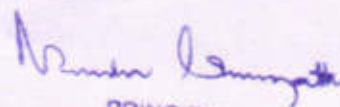
Sl No.	USN	NAME
1	1SV17IS001	NITHIN KUMAR B N
2	1SV17IS002	RACHANA V
3	1SV17IS003	RAKIYA UZMA
4	1SV17IS004	SANTHOSHBHARADWAJ H A


Prof. Kumar H R
Head, Dept. of ISE

HOD
Dept. of ISE
SIET, Tumkur-06.


PRINCIPAL
SIET, TUMAKURU.

UNIX AND SHELL PROGRAMMING [As per Choice Based Credit System (CBCS) scheme] (Effective from the academic year 2017 -2018) SEMESTER – III			
Subject Code	17CS35	IA Marks	40
Number of Lecture Hours/Week	03	Exam Marks	60
Total Number of Lecture Hours	40	Exam Hours	03
CREDITS – 03			
Module -1			Teaching Hours
Introduction, Brief history. Unix Components/Architecture. Features of Unix. The UNIX Environment and UNIX Structure, Posix and Single Unix specification. The login prompt. General features of Unix commands/ command structure. Command arguments and options. Understanding of some basic commands such as echo, printf, ls, who, date, passwd, cal, Combining commands. Meaning of Internal and external commands. The type command: knowing the type of a command and locating it. The man command knowing more about Unix commands and using Unix online manual pages. The man with keyword option and whatis. The more command and using it with other commands. Knowing the user terminal, displaying its characteristics and setting characteristics. Managing the non-uniform behaviour of terminals and keyboards. The root login. Becoming the super user: su command. The /etc/passwd and /etc/shadow files. Commands to add, modify and delete users. Topics from chapter 2 , 3 and 15 of text book 1,chapter 1 from text book 2			08 Hours
Module -2			08 Hours
Unix files. Naming files. Basic file types/categories. Organization of files. Hidden files. Standard directories. Parent child relationship. The home directory and the HOME variable. Reaching required files- the PATH variable, manipulating the PATH, Relative and absolute pathnames. Directory commands – pwd, cd, mkdir, rmdir commands. The dot (.) and double dots (..) notations to represent present and parent directories and their usage in relative path names. The related commands – cat, mv, rm, cp, wc and od commands. File attributes and permissions and knowing them. The ls command with options. Changing file permissions: the relative and absolute permissions changing methods. Recursively changing file permissions. Directory permissions. Topics from chapters 4, 5 and 6 of text book 1			
Module – 3			08 Hours
The vi editor. Basics. The .exrc file. Different ways of invoking and quitting vi. Different modes of vi. Input mode commands. Command mode commands. The ex mode commands. Illustrative examples Navigation commands. Repeat command. Pattern searching. The search and replace command. The set, map and abbr commands. Simple examples using these commands. The shell's interpretive cycle. Wild cards and file name generation. Removing the special meanings of wild cards. Three standard files and redirection. Connecting commands: Pipe. Splitting the output: tee. Command substitution. Basic and Extended regular expressions. The grep, egrep. Typical examples involving different regular expressions. Topics from chapters 7, 8 and 13 of text book 1. Topics from chapter 2 and 9 ,10 of text book 2			
Module-4			


 PRINCIPAL
 SIET, TUMAKURU.

Shell programming. Ordinary and environment variables. The .profile. Read and readonly commands. Command line arguments. exit and exit status of a command. Logical operators for conditional execution. The test command and its shortcut. The if, while, for and case control statements. The set and shift commands and handling positional parameters. The here (<<) document and trap command. Simple shell program examples. File inodes and the inode structure. File links – hard and soft links. Filters. Head and tail commands. Cut and paste commands. The sort command and its usage with different options. The umask and default file permissions. Two special files /dev/null and /dev/tty.

Topics from chapter 11, 12, 14 of text book 1, chapter 17 from text book 2

08 Hours

Module-5

Meaning of a process. Mechanism of process creation. Parent and child process. The ps command with its options. Executing a command at a specified point of time: at command. Executing a command periodically: cron command and the crontab file.. Signals. The nice and nohup commands. Background processes. The bg and fg command. The kill command. The find command with illustrative example.

Structure of a perl script. Running a perl script. Variables and operators. String handling functions. Default variables - \$_ and \$. – representing the current line and current line number. The range operator. Chop() and chomp() functions. Lists and arrays. The @- variable. The splice operator, push(), pop(), split() and join(). File handles and handling file – using open(), close() and die () functions.. Associative arrays – keys and value functions. Overview of decision making loop control structures – the foreach. Regular expressions – simple and multiple search patterns. The match and substitute operators. Defining and using subroutines.

08 Hours

Topics from chapter 9 and 19 of text book 1. Topics from chapter 11 of reference book 1

Course outcomes:

After studying this course, students will be able to:

- Explain UNIX system and use different commands.
- Compile Shell scripts for certain functions on different subsystems.
- Demonstrate use of editors and Perl script writing

Question paper pattern:

The question paper will have ten questions.
 There will be 2 questions from each module.
 Each question will have questions covering all the topics under a module.
 The students will have to answer 5 full questions, selecting one full question from each module.

Text Books:

1. Sumitabha Das., Unix Concepts and Applications., 4th Edition., Tata McGraw Hill
2. Behrouz A. Forouzan, Richard F. Gilberg : UNIX and Shell Programming- Cengage Learning –, India Edition. 2009.

Reference Books:

1. M.G. Venkatesh Murthy: UNIX & Shell Programming, Pearson Education.
2. Richard Blum , Christine Bresnahan : Linux Command Line and Shell Scripting Bible, 2nd Edition , Wiley, 2014.

DISCRETE MATHEMATICAL STRUCTURES
 [As per Choice Based Credit System (CBCS) scheme]
 (Effective from the academic year 2017 -2018)
SEMESTER – III



COURSE LECTURE PLAN

Semester: III

Year: 2018-2019

Course Title: UNIX AND SHELL PROGRAMMING	Course Code: 17CS35
Total contact Hours: 62	Duration of Exam: 03 Hrs.
Total exam marks: 60	Total I.A. marks: 40
Lesson plan author: Mr Renukaradhya P C	Date: 30/07/2018
Checked by: Prof. C V Shanmuka Swamy	Date: 30/07/2018

OBJECTIVES

This course will enable students to

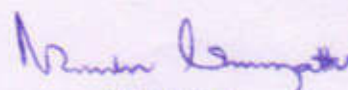
- Understand the UNIX Architecture, File systems and use of basic Commands.
- Use of editors and Networking commands.
- Understand Shell Programming and to write shell scripts.
- Understand and analyze UNIX System calls, Process Creation, Control Relationship.

OUTCOMES

- Explain UNIX system and use different commands.
- Compile Shell scripts for certain functions on different subsystems.
- Demonstrate use of editors and Perl script writing

PRINCIPAL
SIET, TUMAKURU.

SL No	DATE	TOPIC NAMES	TOPICS COVERED	REMARKS
1	01-08-18	Module-1: Introduction, Brief history. Unix Components/Architecture. Features of Unix.		
2	02-08-18	The UNIX Environment and UNIX Structure, Posix and Single Unix specification.		
3	04-08-18	The login prompt. General features of Unix commands/ command structure		
4	06-08-18	Command arguments and options		
5	08-08-18	Understanding of some basic commands such as echo, printf, ls, who, date, passwd, cal,		
6	09-08-18	Combining commands. Meaning of Internal and external commands		
7	11-08-18	The type command: knowing the type of a command and locating it		
8	13-08-18	The man command knowing more about Unix commands and using Unix online manual pages.		
9	16-08-18	The man with keyword option and whatis. The more command and using it with other commands		
10	18-08-18	Knowing the user terminal, displaying its characteristics and setting characteristics.		
11	20-08-18	Managing the non-uniform behaviour of terminals and keyboards. The root login.		
12	23-08-18	Becoming the super user: su command. The /etc/passwd and /etc/shadow files. Commands to add, modify and delete users.		
13	25-08-18	Module – 2: Unix files. Naming files. Basic file types/categories. Organization of files. Hidden files.		
14	27-08-18	Standard directories. Parent child relationship. The home directory and the HOME variable		
15	29-08-18	Reaching required files- the PATH variable, manipulating the PATH, Relative and absolute pathnames		
16	30-08-18	Directory commands – pwd, cd, mkdir, rmdir commands. The dot (.) and double dots (..) notations to represent present and parent directories and their usage in relative path names		


 PRINCIPAL
 SIET, TUMAKURU.

17	01-09-18	File related commands – cat, mv, rm, cp, wc and od commands		
18	03-09-18	File attributes and permissions and knowing them.		
19	05-09-18	The ls command with options.		
20	06-09-18	Changing file permissions: the relative and absolute permissions		
21	08-09-18	changing methods. Recursively changing file permissions. Directory permissions		
	10-09-18			
22	12-09-18	Module – 3: The vi editor. Basics. The .exrc file. Different ways of invoking and quitting vi		
23	15-09-18	Different modes of vi. Input mode commands. Command mode commands. The ex mode commands		
24	17-09-18	Illustrative examples Navigation commands. Repeat command. Pattern searching.		
25	19-09-18	The search and replace command. The set, map and abbr commands.		
26	26-09-18	Simple examples using these commands. The shells interpretive cycle		
27	27-09-18	Wild cards and file name generation. Removing the special meanings of wild cards		
28	29-09-18	Three standard files and redirection. Connecting commands: Pipe. Splitting the		
29	01-10-18	output: tee. Command substitution. Basic and Extended regular expressions		
30	03-10-18	The grep, egrep. Typical examples involving different regular expressions.		
31	04-10-18	Module – 4: Shell programming. Ordinary and environment variables. The .profile. Read and readonly commands		
32	06-10-18	Command line arguments. exit and exit status of a command.		
33	10-10-18	Logical operators for conditional execution. The test command and its shortcut		
34	11-10-18	The if, while, for and case control statements.		
35	13-10-18	The set and shift commands and handling positional parameters		
36	15-10-18	The here (<<) document and trap command. Simple shell program examples		
37	17-10-18	File inodes and the inode structure. File links – hard and soft links		

38	20-10-18	Filters. Head and tail commands. Cut and paste commands. The sort command and its usage with different options		
39	22-10-18	The umask and default file permissions. Two special files /dev/null and /dev/tty		
40	25-10-18	Module – 5 Meaning of a process. Mechanism of process creation. Parent and child process. The ps command with its options		
41	27-10-18	Executing a command at a specified point of time: at command.		
42	03-11-18	Executing a command periodically: cron command and the crontab file.. Signals		
43	05-11-18	The nice and nohup commands. Background processes. The bg and fg command. The kill command		
44	07-11-18	The find command with illustrative example. Structure of a perl script. Running a perl script		
45	10-11-18	Variables and operators. String handling functions. Default variables - \$_ and \$. – representing the current line and current line number		
46	12-11-18	The range operator. Chop() and chomp() functions. Lists and arrays. The @- variable		
47	12-11-18	The splice operator, push(),		
48	14-11-18	pop(), split() and join().		
49	14-11-18	File handles and handling file – using open(), close() and		
50	15-11-18	die ()functions		
51	15-11-18	Associative arrays – keys and value functions.		
52	17-11-18	Overview of decision making loop control structures – the foreach		
53	19-11-18	Regular expressions – simple and multiple search patterns		
54	22-11-18	The match and substitute operators.		
55	24-11-18	Defining and using subroutines.		

Text Books:

1. Sumitabha Das., Unix Concepts and Applications., 4th Edition., Tata McGraw Hill
2. Behrouz A. Forouzan, Richard F. Gilberg : UNIX and Shell Programming- Cengage Learning – India Edition. 2009.

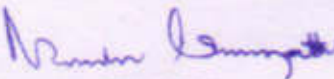
Reference Books:

1. M.G. Venkatesh Murthy: UNIX & Shell Programming, Pearson Education.
2. Richard Blum , Christine Bresnahan : Linux Command Line and Shell Scripting Bible, 2ndEdition , Wiley,2014.


Mr. Renukaradhya P C
Staff-Incharge


Prof. C V Shanmuka Swamy
Head, Dept of CSE


Dr. T Hemadri Naidu
Principal


PRINCIPAL
SIET, TUMAKURU.

COURSE LECTURE PLAN

Semester: III

Year: 2018-2019

Course Title: UNIX AND SHELL PROGRAMMING	Course Code: 17CS35
Total contact Hours: 62	Duration of Exam: 03 Hrs.
Total exam marks: 60	Total I.A. marks: 40
Lesson plan author: Mr Renukaradhya P C	Date: 30/07/2018
Checked by: Prof. H R Kumar	Date: 30/07/2018

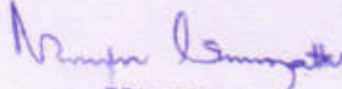
OBJECTIVES

This course will enable students to

- Understand the UNIX Architecture, File systems and use of basic Commands.
- Use of editors and Networking commands.
- Understand Shell Programming and to write shell scripts.
- Understand and analyze UNIX System calls, Process Creation, Control Relationship.

OUTCOMES

- Explain UNIX system and use different commands.
- Compile Shell scripts for certain functions on different subsystems.
- Demonstrate use of editors and Perl script writing


PRINCIPAL
SIET, TUMAKURU.

SL No	DATE	TOPIC NAMES	TOPICS COVERED	REMARKS
1	01-08-18	Module-1: Introduction, Brief history. Unix Components/Architecture. Features of Unix.		
2	02-08-18	The UNIX Environment and UNIX Structure, Posix and Single Unix specification.		
3	04-08-18	The login prompt. General features of Unix commands/ command structure		
4	06-08-18	Command arguments and options		
5	08-08-18	Understanding of some basic commands such as echo, printf, ls, who, date, passwd, cal,		
6	09-08-18	Combining commands. Meaning of Internal and external commands		
7	11-08-18	The type command: knowing the type of a command and locating it		
8	13-08-18	The man command knowing more about Unix commands and using Unix online manual pages.		
9	16-08-18	The man with keyword option and whatis. The more command and using it with other commands		
10	18-08-18	Knowing the user terminal, displaying its characteristics and setting characteristics.		
11	20-08-18	Managing the non-uniform behaviour of terminals and keyboards. The root login.		
12	23-08-18	Becoming the super user: su command. The /etc/passwd and /etc/shadow files. Commands to add, modify and delete users.		
13	25-08-18	Module – 2: Unix files. Naming files. Basic file types/categories. Organization of files. Hidden files.		
14	27-08-18	Standard directories. Parent child relationship. The home directory and the HOME variable		
15	29-08-18	Reaching required files- the PATH variable, manipulating the PATH, Relative and absolute pathnames		
16	30-08-18	Directory commands – pwd, cd, mkdir, rmdir commands. The dot (.) and double dots (..) notations to represent present and parent directories and their usage in relative path names		

17	01-09-18	File related commands – cat, mv, rm, cp, wc and od commands		
18	03-09-18	File attributes and permissions and knowing them.		
19	05-09-18	The ls command with options.		
20	06-09-18	Changing file permissions: the relative and absolute permissions		
21	08-09-18	changing methods. Recursively changing file permissions. Directory permissions		
	10-09-18			
22	12-09-18	Module – 3: The vi editor. Basics. The .exrc file. Different ways of invoking and quitting vi		
23	15-09-18	Different modes of vi. Input mode commands. Command mode commands. The ex mode commands		
24	17-09-18	Illustrative examples Navigation commands. Repeat command. Pattern searching.		
25	19-09-18	The search and replace command. The set, map and abbr commands.		
26	26-09-18	Simple examples using these commands. The shells interpretive cycle		
27	27-09-18	Wild cards and file name generation. Removing the special meanings of wild cards		
28	29-09-18	Three standard files and redirection. Connecting commands: Pipe. Splitting the		
29	01-10-18	output: tee. Command substitution. Basic and Extended regular expressions		
30	03-10-18	The grep, egrep. Typical examples involving different regular expressions.		
31	04-10-18	Module – 4: Shell programming. Ordinary and environment variables. The .profile. Read and readonly commands		
32	06-10-18	Command line arguments. exit and exit status of a command.		
33	10-10-18	Logical operators for conditional execution. The test command and its shortcut		
34	11-10-18	The if, while, for and case control statements.		
35	13-10-18	The set and shift commands and handling positional parameters		
36	15-10-18	The here (<<) document and trap command. Simple shell program examples		
37	17-10-18	File inodes and the inode structure. File links – hard and soft links		

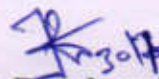
38	20-10-18	Filters. Head and tail commands. Cut and paste commands. The sort command and its usage with different options		
39	22-10-18	The umask and default file permissions. Two special files /dev/null and /dev/tty		
40	25-10-18	Module – 5 Meaning of a process. Mechanism of process creation. Parent and child process. The ps command with its options		
41	27-10-18	Executing a command at a specified point of time: at command.		
42	03-11-18	Executing a command periodically: cron command and the crontab file.. Signals		
43	05-11-18	The nice and nohup commands. Background processes. The bg and fg command. The kill command		
44	07-11-18	The find command with illustrative example. Structure of a perl script. Running a perl script		
45	10-11-18	Variables and operators. String handling functions. Default variables - \$_ and \$. – representing the current line and current line number		
46	12-11-18	The range operator. Chop() and chomp() functions. Lists and arrays. The @- variable		
47	12-11-18	The splice operator, push(),		
48	14-11-18	pop(), split() and join().		
49	14-11-18	File handles and handling file – using open(), close() and		
50	15-11-18	die ()functions		
51	15-11-18	Associative arrays – keys and value functions.		
52	17-11-18	Overview of decision making loop control structures – the foreach		
53	19-11-18	Regular expressions – simple and multiple search patterns		
54	22-11-18	The match and substitute operators.		
55	24-11-18	Defining and using subroutines.		

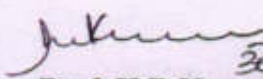
Text Books:

1. Sumitabha Das., Unix Concepts and Applications., 4th Edition., Tata McGraw Hill
2. Behrouz A. Forouzan, Richard F. Gilberg : UNIX and Shell Programming- Cengage Learning – India Edition. 2009.

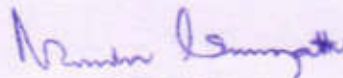
Reference Books:

1. M.G. Venkatesh Murthy: UNIX & Shell Programming, Pearson Education.
2. Richard Blum , Christine Bresnahan : Linux Command Line and Shell Scripting Bible, 2ndEdition , Wiley,2014.


Mr. Renukaradhya P C
Staff-Incharge


Prof. H R Kumar
Head, Dept of ISE


Dr. T Hemadri Naidu
Principal


PRINCIPAL
SIET, TUMAKURU.



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INTERNAL ASSESMENT TEST: I



COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. Explain the architecture of UNIX operating system with a neat diagram? 05M
b. Explain the features of UNIX? 05M
c. What is file? Explain the categories of files? 05M
OR
- 2 a. Explain with example what are internal and external commands in? 05M
b. with the help of a diagram, explain parent-child relationship, explain the UNIX file system? 05M
c. Explain the changing file permissions in UNIX 05M
- 3 a.. Explain the following command with example
i) ls -axF
ii) lp rpc.ps
iii) chown Sharma note; ls -l note 05M
b Explain the use of more command and its options? 05M
c Explain the security implications of files in UNIX? 05M
OR
- 4 a Write a note on man command and its documentation? 05M
b Explain Absolute and Relative path names? 05M
c Explain how to change the ownership of files in UNIX? 05M



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INTERNAL ASSESMENT TEST: I



COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. Explain the architecture of UNIX operating system with a neat diagram? 05M
b. Explain the features of UNIX? 05M
c. What is file? Explain the categories of files? 05M
OR
- 2 a. Explain with example what are internal and external commands in? 05M
b. with the help of a diagram, explain parent-child relationship, explain the UNIX file system? 05M
c. Explain the changing file permissions in UNIX 05M
- 3 a.. Explain the following command with example
i) ls -axF
ii) lp rpc.ps
iii) chown Sharma note; ls -l note 05M
b Explain the use of more command and its options? 05M
c Explain the security implications of files in UNIX? 05M
OR
- 4 a Write a note on man command and its documentation? 05M
b Explain Absolute and Relative path names? 05M
c Explain how to change the ownership of files in UNIX? 05M

Principal Signature
PRINCIPAL
SIET., TUMAKURU.

Scheme of Evaluation IA-I

UNIX & Shell programming

- 1
- a) Diagram - 1M Explanation - 4M
 - b) 5 Features 5M
 - c) File Definition - 1M, categories - 4M
- 2
- OR
- a) Internal Commands - 2.5M
External Commands - 2.5M
 - b) diagram - 1M Explanation - 2M
~~UNIX~~ UNIX file System - 2M
 - c) Explanation of changing file permissions. - 5M
- 3
- a) i) 1.5M
ii) 1.5M
iii) 2M
 - b) Explanation of more Command and its options 5M
 - c) Security implications of files in UNIX - 5M
- OR
- 4
- a) Note on man Command and its documentation - 5M
 - b) Absolute path Names - 2.5M Relative path Names - 2.5M
 - c) Explanation of how to change the ownership of files in UNIX - 5M

Unix and Shell programming

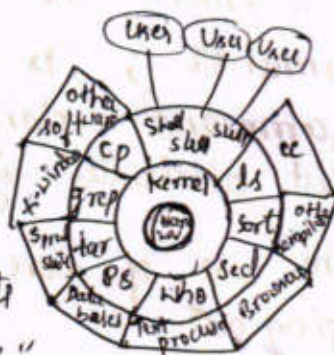
Internal Assessment test : I

15. a. explain the architecture of UNIX operating system with a neat diagram?

→ UNIX architecture has two parts namely Kernel & Shell.

Kernel is a core of operating system which interacts with machine hardware.

Shell is an ^{output of} operating system which interacts with user called as an "command interpreter".



UNIX ARCHITECTURE

b) Explain the features of UNIX?

⇒ Features of UNIX are,

- * A multiuser system
- * A multitasking system
- * The building block approach
- * The UNIX toolkit
- * Pattern matching
- * Programming facilities

c) What is file? explain the categories of files?

⇒ A file is a container for storing information.

Types of file: 1) ordinary file 2) Directory file 3) Device file.

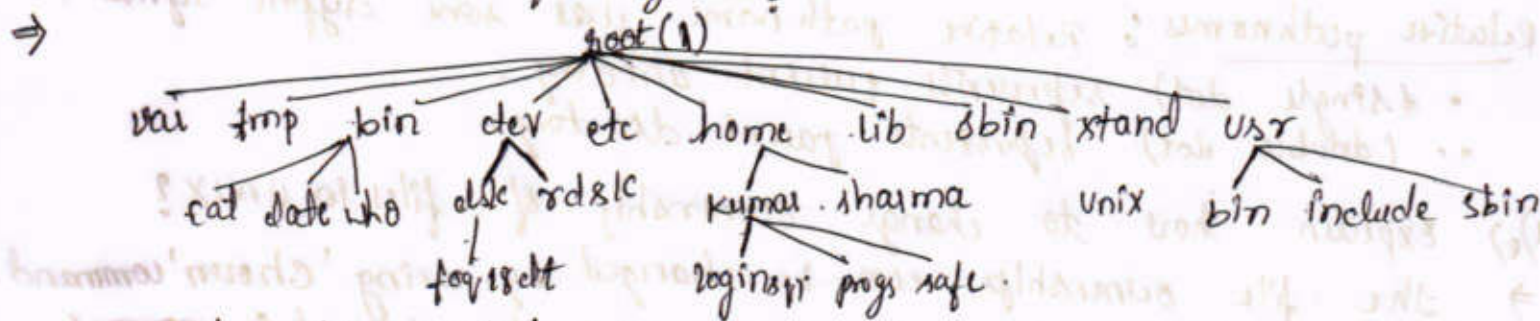
2a) explain with example what are internal and external commands?

Internal commands are built in commands. eg: echo.

External commands are which stored in some separate files.

eg: ls, who, cat etc

2b) With the help of a diagram, explain parent-child relationship, explain the UNIX file system?



UNIX file system contains many commands.

2c) explain the changing file permissions in UNIX.

⇒ File permissions can be changed by using chmod (change mode) command.

Syntax: \$ chmod category operation permissions filename

→ \$ chmod u+x -chap 01

changes the file mode & add executable permission to user group.

3) a) explain the following command with example.

i) `ls -axF`

This command list the hidden files output in multiple columns.

ii) `lprpe.ps` → prints the line of file `rpe.ps`

iii) `chown sharma:note ; ls -l note`

→ first ^{cmd} command change ownership of file from sharma to note
& second ^{cmd} list attributes of file note.

3) b) explain the use of more command and its options?

→ more: paging out

more is the pager program used in UNIX.

It uses `f` for forward scroll & `b` for backward scroll.

3) c) Explain the security implications of files in UNIX?

→ In UNIX system password is in encrypted form, it does not display as string for security purpose.

4) a) Write a note on man command & its documentation?

→ Man: Browsing online information.

Man command gives the information about some commands & its output is send to pager program. It consists of name of the command, synopsis & description.

4) b) Explain absolute and relative pathnames?

→ Absolute pathnames: If the first character of pathname is the file location must be determined with respect to root (/) such a pathname are Absolute pathnames.

• /home/sharma/a.c

Relative pathnames: Relative pathname uses some cryptic symbols.

• `.` (single dot) represents current directory.

• `..` (double dot) represents parent directory.

4) c) Explain how to change ownership of files in UNIX?

→ The file ownership can be changed by using 'chown' command.

The ownership of the command cannot be changed in normal mode, initially the computer mode has to be changed to super user mode the the ownership has to be changed.

```
$ chown amar xy30
```

```
$ su (superuser)
```

```
****  
# chown -amar xy3
```

```
# ls -l xy3
```




SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: II
COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

Date :30/10/2018
MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. Explain with a neat diagram, the three modes of editor? **05M**
b. Explain the three standard files in UNIX? **05M**
c. Explain the two /dev/null and /dev/tty special files? **05M**
OR
- 2 a. Explain the pipes and tee command? **05M**
b. Explain the pattern matching wild-cards with examples? **05M**
c. Explain the shell interpretive cycle? **05M**
- 3 a.. Explain the following command with example
i) grep -e "Agarwal" -e "aggarwal" -e "agrwal" emp.lst
ii) grep "j.*saxena" emp.lst
iii) grep "^5" emp.lst **05M**
b Explain the Escaping and Quoting in UNIX? **05M**
c Explain the purpose of grep command and its options? **05M**
OR
- 4 a Explain the Ordinary and Environment variables in shell programming? **05M**
b Explain the purpose of Extended regular expressions with examples? **05M**
c Explain matching multiple patterns | and () in UNIX? **05M**



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: II
COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

Date :30/10/2018
MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. Explain with a neat diagram, the three modes of editor? **05M**
b. Explain the three standard files in UNIX? **05M**
c. Explain the two /dev/null and /dev/tty special files? **05M**
OR
- 2 a. Explain the pipes and tee command? **05M**
b. Explain the pattern matching wild-cards with examples? **05M**
c. Explain the shell interpretive cycle? **05M**
- 3 a.. Explain the following command with example
i) grep -e "Agarwal" -e "aggarwal" -e "agrwal" emp.lst
ii) grep "j.*saxena" emp.lst
iii) grep "^5" emp.lst **05M**
b Explain the Escaping and Quoting in UNIX? **05M**
c Explain the purpose of grep command and its options? **05M**
OR
- 4 a Explain the Ordinary and Environment variables in shell programming? **05M**
b Explain the purpose of Extended regular expressions with examples? **05M**
c Explain matching multiple patterns | and () in UNIX? **05M**

[Handwritten Signature]

PRINCIPAL
SIET, TUMAKURU.

UNIX and Shell programming

Scheme of Evaluation I A - II

- 1 a) Diagram - 2M Explanation - 3M
- b) Explanation of three standard files - 5M
- c) Explanation of /dev/null - 3M /dev/tty - 2M

OR

- 2 a) Pipes - 2.5M tee command - 2.5M
- b) Explanation of pattern matching wild cards with examples - 5M
- c) Shell interpretive cycle 5 steps x 1M = 5M

- 3 a) i) 1M ii) 2M iii) 2M

- b) Explanation of Escaping and quoting - 5M

OR

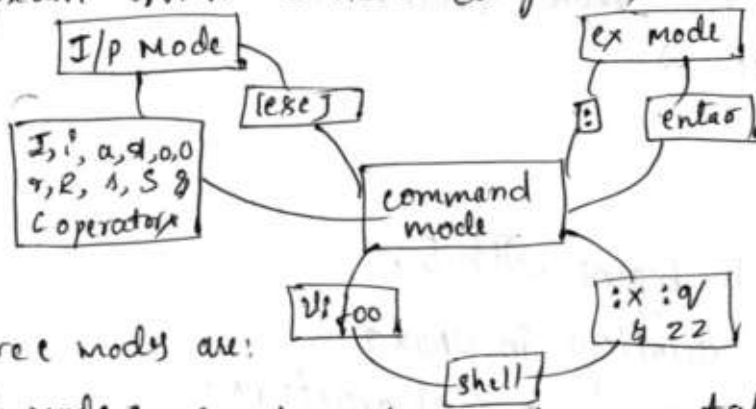
- 4 a) Explanation of Ordinary and Environment Variables - 5M

- b) Explain the purpose of Extended regular Expression with examples - 5M

- c) Explanation of matching multiple patterns | and () in UNIX - 5M

Internal assessment test? - II

1) a) explain with a neat diagram, the three modes of editor?



The three modes are:

Command mode, Input mode and executable mode.

1) b) explain the three standard files in UNIX?

→ The three standard files in UNIX are;

- i) standard input : represents input which is connected to keyboard.
- ii) standard output : represents output which is connected to display.
- iii) standard error : Represents error messages that emanate from command.

1) c) Explain the two /dev/null and /dev/tty special files?

→ /dev/null : the size of this file is always zero.

/dev/tty file represents the terminal running in system.

2) a) explain the pipes and tee command?

→ Pipe command combines or connect two commands.

tee command displays the output of the command ^{on} and terminal and saves one copy of output in a file.

2) b) explain the pattern matching with -cards with examples?

→ The pattern followed by '/' (front slash) searches pattern present in existing files.

\$! while a.c.

2) c) explain the shell interpretive cycle?

→ The input is entered from standard input device keyboard. When input is entered the shell scans for special character, if not then it sends it command to kernel, when program is running in kernel, shell is not active, then shell displays & sends required output to terminal. Then shell get ready to second interpretive cycle.

3) a) explain the following command with example.

1) `grep -e "agrawal" -e "aggawal" -e "agrwal" emp.lst`

→ The grep command with -e option matches multiple patterns one at a time.

ii) `grep "j.*saxena" emp.lst`

⇒ In this command `.*` matches one, more characters like
`$ grep "j.*saxena" emp.lst`
 j.b.saxena.
 j.ccc.saxena.

iii) `grep "5" emp.lst`

It searches for line which begins with 5.

3) b) explain the escaping and quoting in UNIX?

⇒ `\` → escapes the meaning of special characters.
`rm 'chap*' → it treats all as normal characters which are inside single quot.`
`$ ls chap 0[1-31]` `$ rm "my doc.txt"`
`chap 0[1-3]`

3) c) explain the purpose of `grep` command & its options?

⇒ `grep` command searches for pattern. `grep` options are,
`grep -i` → ignores case for matching.
`grep -n` → display line number along with lines.
`grep -c` → displays count of number of occurrences.

4) a) explain the ordinary & environment variables in shell programming?

⇒ `PATH`, `HOME` & `SHELL` are environment variables. they are called `BO` because they are available in users total environment.
`$ echo $DOWNLOAD_DIR = /home/kumar/download.`
`DOWNLOAD_DIR` is local variable: its value is not available to other process.

4) b) explain the purpose of Extended regular expressions with examples?

⇒ ERE make it possible to match dissimilar patterns with a single expressions.
`+` - matches one or more occurrences of previous character
`?` - matches zero or one occurrence of previous character
`$ grep -E "[aA]gg? arwal" emp.lst`

4) c) explain multiple matching patterns `|` & `()` in UNIX?

⇒ `|` & `()` matches multiple patterns.
`$ grep -E "(sengupta / dasgupta)" emp.lst`

ⓐ `$ grep -E "(sen|das)gupta" emp.lst`

2001 | sengupta | D.M | salary | 5/1/59 | 5000

2002 | dasgupta | G.M | Salary | 3/12/45 | 8000



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



INTERNAL ASSESMENT TEST: III
COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

Date : 28/11/2018
MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. What is shell programming? Write a shell program to create a menu which displays
i) list of files ii) current date iii) process status iv) current user of the system v) quit to UNIX **08M**
b. Explain the shell conditional statements in with proper syntax **07M**
OR
- 2 a. Explain the use of test and [] to evaluate an expression in shell? **08M**
b. Explain the following command with example.
i) set and shift
ii) *HERE DOCUMENT* (<<) **07M**
- 3 a. Define Process, Explain the mechanism of process creation? **08M**
b Explain the cron command, and the cron tab file? **07M**
OR
- 4 a Write a perl program to find the average of three numbers? **08M**
b Explain the nice, nohup, bg and fg commands with examples? **07M**



SHRIDEVI INSTITUTE OF ENGINEERING & TECHNOLOGY
TUMKUR-572106
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

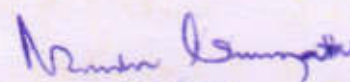


INTERNAL ASSESMENT TEST: III
COURSE : UNIX & Shell Programming
SEM : III SEM (17CS35)

Date : 28/11/2018
MAX MARKS : 30
TIME : 75 min

NOTE : Answer any TWO full questions.

- 1 a. What is shell programming? Write a shell program to create a menu which displays
i) list of files ii) current date iii) process status iv) current user of the system v) quit to UNIX **08M**
b. Explain the shell conditional statements in with proper syntax **07M**
OR
- 2 a. Explain the use of test and [] to evaluate an expression in shell? **08M**
b. Explain the following command with example.
i) set and shift
ii) *HERE DOCUMENT* (<<) **07M**
- 3 a. Define Process, Explain the mechanism of process creation? **08M**
b Explain the cron command, and the cron tab file? **07M**
OR
- 4 a Write a perl program to find the average of three numbers? **08M**
b Explain the nice, nohup, bg and fg commands with examples? **07M**


PRINCIPAL
SIET., TUMAKURU.

Scheme of Evaluation

USP-2A-III

1 a) Shell programming - 1M , Prg - 7M

b) Conditional statements with proper syntax - 7M

2 a) OR
Explanation of test and [] to evaluate the expressions - 8M

b) i) set - 2M & shift - 2M

ii) HERE DOCUMENT (<<) - 3M

3 a) Process definition - 1M , Mechanism of process creation - 7M

b) Explanation of Cron & Crontab file - 7M

OR

4 a) Perl prg to find the average of three numbers - 8M

b) Explanation of nice, nohup, bg and fg Commands. - 7M

Internal Assessment test : III

- 1) a) What is shell programming? Write a shell program to create a menu which displays i) list of files ii) current date iii) process status iv) current user of the system v) quit to UNIX.
- Group of commands executing regularly; they should be stored in file & the file itself executed as shell program.

```
#!/bin/sh
# menu.sh : uses case to offer 5-item menu
#
echo "MENU in
1. List of files in 2. Processes of user in 3. today's date in
4. User of system in 5. Quit to UNIX in enter your option: 1c"
read choice
case "$choice" in
  1) ls -l ;;
  2) ps -f ;;
  3) date ;;
  4) who ;;
  5) exit ;;
  *) echo "Invalid option"
esac
```

- 1) b) Explain shell conditional statements with proper syntax.
- if: if command is successful then execute commands. fi
- while: while condition is true do commands done
- for: for variable in list do commands done
- case: case expression in pattern 1) command 1;; pattern 2) command 2;; pattern 3) command 3;; esac

- 2) a) Explain the use of test and [] to evaluate an expression in shell?
- test works in three ways.
- 1) compares two numbers.
 - 2) compares two strings or a single one for a null value
 - 3) checks a file attributes.

- 2) b) Explain the following command with example.
- is set and shift
- set command saves the given arguments in the positional parameters \$1, \$2, \$3, ...
- shift command shifts the arguments towards left.
- ```
$ echo " $#@"
Wed Jan 8 09:44:45 2017
$ shift
$ echo " $#@"
Jan 8 09:44
```

ii) HERE Document (<<):

The ~~the~~ HERE DOCUMENT (<<) is used when input of the file is strictly predicted and when the <sup>computer</sup> file doesn't accept the file name as argument.

\$ mailx rachana << I'm printing the programs till on 'date'.

3) a) Define process. Explain the mechanism of process creation?

⇒ A program in execution called process.

The mechanism of process involves three steps

i) Fork: The process in UNIX is created with fork system call, which creates a copy of the process which invokes it. Process get PID.

ii) Exec: Forking creates process, but to run, it has to be overwritten data with new program data. This mechanism is called exec.

iii) wait: Parent <sup>executed</sup> wait system call to wait for child process to complete process, it picks up exit status of child.

3) b) Explain the cron command, and the cron tab file?

⇒ Cron: running jobs periodically. cron - creating a cron tab file.  
\$ crontab 'cron.txt'

We can see the contents of this crontab file with crontab -l & remove them with crontab -r.

4) a) Write a perl program to find the average of three commands?

⇒  
#!/usr/bin/perl  
print ('enter the first no');  
\$a = <STDIN>  
print ('enter the 2nd no');  
\$b = <STDIN>  
print ('enter 3rd no');  
\$c = <STDIN>  
\$arg = (\$a + \$b + \$c) / 3;  
print "Avg of 3 no's = \$arg\n";

4) b) Explain the nice, nohup, bg and fg command with examples?

⇒ Nice: job execution with low priority

\$ nice +nrc 5 wc -l emp1.txt &

nohup: (no hang up) To logout safely. \$ nohup sort -o emp1.txt &

bg: Relegate a job to the background

fg: Bring it back to the foreground.



|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

## Fourth Semester B.E. Degree Examination, Dec.2015/Jan.2016

### Unix and Shell Programming

Time: 3 hrs.

Max. Marks:100

**Note: Answer FIVE full questions, selecting at least TWO questions from each part.**

#### PART – A

- 1 a. Explain the Architecture of UNIX operating system with a neat diagram. (08 Marks)  
b. Describe the salient features of UNIX operating system. (08 Marks)  
c. Write a note on man command. (04 Marks)
- 2 a. Explain the different types of files supported in UNIX. (06 Marks)  
b. Which command is used for listing file attributes? Explain significance of each field in the output. (08 Marks)  
c. Explain with a neat diagram the three modes of Vi – editor. (06 Marks)
- 3 a. What are standard input, standard output and standard error? Explain in detail with example. (06 Marks)  
b. Define the term process. Explain the mechanism of process creation in UNIX. (06 Marks)  
c. Explain the following command with an example  
i) Running jobs in background (& and nohup)  
ii) Execute later (at and batch) (08 Marks)
- 4 a. Write a note on sort and find command. (08 Marks)  
b. Differentiate between Hard link and Soft link in UNIX with example. (06 Marks)  
c. Explain the following commands with example  
i) Head ii) tail iii) Pr (06 Marks)

#### PART – B

- 5 a. What is the difference between a wild card and regular expression? Explain 'grep' command using n, l and f option with example. (06 Marks)  
b. What are Extended Regular Expressions? Explain any four ERE set used by grep and egrep. (06 Marks)  
c. Explain line addressing and context addressing in sed with example. (08 Marks)
- 6 a. What is shell programming? Write a shell program to create a menu which displays,  
i) List of files ii) Current date iii) Process status  
iv) Current user of the system and v) Quit to UNIX (08 Marks)  
b. Explain shell features of 'while' and 'for' with syntax. (06 Marks)  
c. Explain the use of test and [ ] to evaluate an expression in shell. (06 Marks)
- 7 a. What is AWK? Explain any three built – in function in AWK. (06 Marks)  
b. Write an AWK sequence to find HRA, DA and Netpay of an employee, where DA is 50% of basic, HRA is 12% of basic and the Netpay is the sum of HRA, DA and Basic pay. (08 Marks)  
c. Briefly describe built in variables in AWK. (06 Marks)
- 8 a. Explain with example the string handling function supported by perl. (08 Marks)  
b. Explain Lists, Arrays and Associative Arrays with respect to perl. (06 Marks)  
c. Write a perl script to convert decimal number to binary number. (06 Marks)

\*\*\*\*\*

  
PRINCIPAL  
SIET., TUMAKURU.



**Third Semester B.E. Degree Examination, Dec.08/Jan.09**  
**UNIX & Shell Programming**

Time: 3 hrs.

Max. Marks:100

**Note: Answer any FIVE full questions, selecting at least TWO questions from each part.**

**PART - A**

- 1
  - a. Explain the salient features of UNIX operating system. (07 Marks)
  - b. Explain the concept of absolute pathname and relative pathname with suitable examples. (04 Marks)
  - c. What is parent-child relationship? With the help of neat diagram explain, UNIX file system tree. (05 Marks)
  - d. What is the output of the following command. (04 Marks)
    - (i) echo \$PATH
    - (ii) man man
    - (iii) cmp f<sub>1</sub> f<sub>2</sub> [f<sub>1</sub> and f<sub>2</sub> are identical]
    - (iv) ls -i
  
- 2
  - a. What is file permission? What are the different ways of setting file permission? Explain (08 Marks)
  - b. Explain the following with respect to vi editor. (06 Marks)
    - (i) Search for a pattern printf, then repeat the search in both forward and backward direction
    - (ii) : \$s | director | member | g
    - (iii) : .w tempfile
    - (iv) : . , \$w tempfile
    - (v) recover
    - (vi) 20h
  - c. Explain the different modes of operation in a vi editor with a suitable diagram. (06 Marks)
  
- 3
  - a. Explain the concept of Escaping and quoting with suitable examples. (06 Marks)
  - b. Frame wild card pattern for the following: (06 Marks)
    - (i) Retrieve hidden files
    - (ii) Any number of characters followed by 4 characters.
    - (iii) Matches all filenames that do not begin with an alphabetic character.
  - c. What is a process? Explain how a process is created using the three primitives fork, exec and wait. (08 Marks)
  
- 4
  - a. Discuss standard input, standard output and standard errors files, in UNIX. (06 Marks)
  - b. Explain the following environment variables with examples. (06 Marks)
    - (i) SHELL
    - (ii) PS2
    - (iii) HOME
  - c. Explain the following commands with examples. (08 Marks)
    - (i) tr
    - (ii) tee
    - (iii) sort
    - (iv) pr



**PART - B**

- 5 a. Explain grep command with any five options with suitable examples. (06 Marks)  
b. List and explain Extended Regular Expression (ERE) set used by grep, egrep and awk. (06 Marks)  
c. Explain the following:  
(i) sed '3q' abc  
(ii) ls -l | grep '^d' > directories  
(iii) sed -n '\$p' abc  
(iv) sed -n '3, \$!p' abc (08 Marks)
- 6 a. Explain special parameters used by the shell. (06 Marks)  
b. Explain the expr command applicable to numeric and string functions. (06 Marks)  
c. Explain here document with an example. Also mention its use. (04 Marks)  
d. Explain trap in shell scripts with a suitable example. (04 Marks)
- 7 a. Explain awk built-in variables and built-in function with suitable examples. (10 Marks)  
b. Explain if statement, for and while loop statement with respect to awk with suitable example. (10 Marks)
- 8 a. Explain the following in perl.  
(i) split  
(ii) join  
(iii) \$ - default variable (06 Marks)  
b. Using command line arguments, write a perl program to find whether a given year is leap year. (07 Marks)  
c. Write a perl program that accepts decimal number as arguments and convert it into binary number. (07 Marks)

\*\*\*\*\*



PRINCIPAL  
SIET, TUMAKURU.

USN

|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

Fifth Semester B.E. Degree Examination, July/August 2005  
 Computer Science Engineering/Information Science  
**Unix & Shell Programming**

Time: 3 hrs.]

[Max.Marks : 100

**Note:** Answer any FIVE full questions.

1. (a) Discuss the salient features of a UNIX system. (5 Marks)
- (b) Discuss the commands using which one can
  - i) Know the users who are using the system at present.
  - ii) Know the terminal settings and change them, if required.
  - iii) Record a session.
- (c) Give the meaning of relative and absolute path names. (5 Marks)
2. (a) Explain the command using which two or more files can be concatenated. Hence or otherwise write a command line to concatenate two files fool and fooz with some text input from the terminal inserted between the contents of these two files. The output has to be stored on a file called fooz. Explain the command line written by you. (10 Marks)
- (b) Mention the different file comparing commands that you know. Explain the one that is used to print out the common entries on two sorted files. (6 Marks)
- (c) Write a command line using which one can assign all permissions to the owner of a file and assign only read permission to group and others. (4 Marks)
3. (a) Clearly differentiate between hard links and soft links. (6 Marks)
- (b) Explain the three modes of the  $v_i$  editor. (6 Marks)
- (c) Explain how one can customize his or her  $v_i$  editing environment. (6 Marks)
- (d) What does the following command do when editing with  $v_i$  : 4, 8s / Unix / UNIX / gc (2 Marks)
4. (a) Mention some of the wild cards that could be used with the shell and explain the special meanings associated with them in the shell environment. If required how can you despecialize the wild cards. (6 Marks)
- (b) Explain the meaning of command substitution with the aid of an example. (4 Marks)
- (c) Explain the mechanism of a process creation. (6 Marks)
- (d) Write a command line to execute a file called myscript on all weekdays, every half hour once between 9 and 17 hrs. (4 Marks)

*Principal*

PRINCIPAL  
SLET, TUMAKURU.

Contd.... 2



6. (a) Explain the local and global variables in shell scripts. (6 Marks)
- (b) Write an awk sequence to find HRA, DA and Net pay of an employee where DA is 50% of basic, HRA is 12% of basic and the Net pay is the sum of HRA, DA and Basic Pay. (8 Marks)
- (c) Explain the built in variables used by awk. (6 Marks)
7. (a) Write a perl script to convert a decimal number to binary. (6 Marks)
- (b) Write a perl program that prompts a user to input a string and a number, and prints the string that many times, with each string on a seperate line. (8 Marks)
- (c) Explain file handling in Perl. (6 Marks)
8. (a) What the privileges of the system administrator of UNIX? (6 Marks)
- (b) Explain mount and unmount commands. (8 Marks)
- (c) Explain UNIX startup and shutdown processes. (6 Marks)

\*\* \* \*\*

  
PRINCIPAL  
SIET, TUMAKURU.

5. (a) Write a sample data base file called mybase.lst that contains 4 records each having 4 fields with the 3rd field being the total masses field. With reference to this file with the names of the highest scores appearing first. (8 Marks)
- (b) Explain the grep command along with some of the options that would be used with it. (7 Marks)
- (c) What is a here document ? Explain. (5 Marks)
6. (a) How can you input data to a shell script interactively ? Discuss. (6 Marks)
- (b) Write a shell script that accepts any number of arguments and prints them in the reverse order. For example if the input arguments are  $x y z$  then the output produced should be  $z y x$ . (8 Marks)
- (c) Explain the expression command. (6 Marks)
7. (a) Discuss the operational mechanism of awk. (6 Marks)
- (b) Explain the awk function that is used to determine the length of a record on a file, say, marks.pu Assume that the record number is 15. (4 Marks)
- (c) Explain the chop ( ) and chomp ( ) functions in perl. (5 Marks)
- (d) With the aid of a simple example explain the split function as available in perl. (5 Marks)
8. (a) Briefly discuss the duties and privileges of a system administrator. (6 Marks)
- (b) Discuss the various partitions of a hard disk on which a file system is implemented. (7 Marks)
- (c) Discuss the crypt command. (5 Marks)

\*\* \* \*\*

  
PRINCIPAL  
SIET, TUMAKURU.



**Third Semester B.E. Degree Examination, Dec.09/Jan.10**  
**Unix and Shell Programming**

Time: 3 hrs.

Max. Marks:100

**Note: Answer any FIVE full questions, selecting  
at least TWO questions from each part.**

**PART - A**

- 1 a. Define an operating system. Discuss the salient features of UNIX operating system. (06 Marks)
- b. Explain the architecture of the UNIX operating system. (08 Marks)
- c. Explain the following commands with examples : i) cat ; ii) rmdir ; iii) pwd. (06 Marks)
- 2 a. A file's current permission are rw\_r\_xr\_.. Specify the chmod expression required to change them for the following:  
i) rw x rw x rw x ; ii) r\_\_r\_\_\_\_ ; iii) \_\_\_\_\_ ; iv) \_\_\_r\_\_r\_\_\_. (08 Marks)
- Using both the relative and absolute methods of assigning permissions.
- b. Explain briefly the file attributes listed using ls -l command. (06 Marks)
- c. What are the different modes of Vi editor? Explain with a diagram. (06 Marks)
- 3 a. Explain what wild - card patterns match:  
i) [A - Z] ????\*      ii) \* [0 - 9] \*  
iii) \* [!0 - 9] and    iv) \* . [!s] [!h]. (08 Marks)
- b. What are standard input, standard output and standard error? Explain with respect to UNIX. (06 Marks)
- c. What is process status? Explain PS command with options. (06 Marks)
- 4 a. Explain the following environment variables with examples:  
i) SHELL ; ii) LOGNAME ; iii) PATH. (06 Marks)
- b. Differentiate between hard link and soft link in UNIX with examples. (06 Marks)
- c. Explain the following commands with examples: i) tail ; ii) paste ; iii) tr ; iv) pr. (08 Marks)

**PART - B**

- 5 a. What is the difference between a wildcard and a regular expression? Explain the "grep" command using n, l and f options with examples. (08 Marks)
- b. What do these regular expressions match? i) a . \* b ; ii) ^ { \$ . (04 Marks)
- c. What is sed? With examples explain the difference between line addressing and context addressing in sed? (08 Marks)
- 6 a. What is shell programming? Write a shell program that will do the following tasks, in order:  
Clear the screen  
Print the current directory  
Display current login users. (08 Marks)
- b. Explain the shell features of "while" and "for" with syntax. (08 Marks)
- c. What is the "exit" status of a command and where is it stored? (04 Marks)
- 7 a. Write an awk program to find square of all the numbers from 1 to 10. (08 Marks)
- b. With example, explain the following built-in variables of awk:  
i) FS ; ii) NF ; iii) FILENAME. (06 Marks)
- c. Explain the following built-in functions of awk with examples:  
i) substr ; ii) index ; iii) length. (06 Marks)
- 8 a. Explain the following string handling functions of PERL with examples:  
i) length ; ii) index ; iii) substr ; iv) reverse. (08 Marks)
- b. Write a PERL program to print numbers that are accepted from the keyboard using while and an array construct. (06 Marks)
- c. Explain the following in PERL with examples:  
i) Foreach looping construct ; ii) Join. (06 Marks)

\*\*\*\*\*

*Principal*

PRINCIPAL  
SIET, TUMAKURU.



USN

15U05CS050

06CS36

**Third Semester B.E. Degree Examination, May/June 2010**  
**UNIX and Shell Programming**

Time: 3 hrs.

Max. Marks:100

*Note: Answer any FIVE full questions, selecting  
at least TWO questions from each part.*

**PART - A**

- 1 a. Explain salient features of UNIX operating system. (06 Marks)  
 b. Explain the different types of files supported in UNIX. (06 Marks)  
 c. Explain the following commands, with example :  
 mailx, passwd, stty, who (08 Marks)
- 2 a. Briefly describe the different ways of setting file permissions. (06 Marks)  
 b. What are the three modes of vi editor? Explain. (06 Marks)  
 c. What is a navigation? What are commands used for navigation in vi editor? (08 Marks)
- 3 a. What are environmental variables? State their significance. (06 Marks)  
 b. What are three standard files used by UNIX commands? Explain. (06 Marks)  
 c. What is shell process? What are three different phases in the creation of process? (08 Marks)
- 4 a. What are hard links and symbolic links? (06 Marks)  
 b. Explain with an example, find command and its operators. (06 Marks)  
 c. Explain the following filters, with examples :  
 head, tail, cut, tr. (08 Marks)

**PART - B**

- 5 a. How to search for a pattern using grep? What are the options used by grep? (08 Marks)  
 b. Explain extended regular expression (ERE) set used by grep. (06 Marks)  
 c. What are internal commands used by sed? (06 Marks)
- 6 a. What are the special parameters used by the shell? (06 Marks)  
 b. Explain how numeric and string comparison is done by using test. (06 Marks)  
 c. Write a menu driven shell script to display list of files, process of user, today's date and users of the system. (08 Marks)
- 7 a. Explain any three built in variables used in awk. (06 Marks)  
 b. Give the syntax of three control flow statements used by awk. (06 Marks)  
 c. Explain built in functions used in awk. (08 Marks)
- 8 a. Explain string handling function in Perl. (06 Marks)  
 b. Explain split and join functions. (06 Marks)  
 c. Write a Perl script to convert a decimal number to binary. (08 Marks)

\*\*\*\*\*



PRINCIPAL  
SIET, TUMAKURU.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.  
2. Any revealing of identification mark to evaluator and/or equations written eg, 42+8 = 50, will be treated as malpractice.



USN

|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

2002 SCHEME

CS55

**Fifth Semester B.E. Degree Examination, Dec 08 / Jan 09**  
**Unix and Shell Programming**

Time: 3 hrs.

Max. Marks:100

**Note : Answer any FIVE full questions.**

- 1
  - a. Describe briefly the UNIX architecture with the help of neat diagram. (06 Marks)
  - b. Differentiate between : i) Internal and External command ii) Argument and Option. (06 Marks)
  - c. Explain the following commands with examples. (08 Marks)
    - i) Uname ii) script iii) stty iv) wc
  
- 2
  - a. Give the output of the following commands : i) cat foo foo foo ii) ls - d .. (06 Marks)
    - iii) chmod u = rwx, g + w, o - w demo.
  - b. What are the default permissions for all files and directories created after issuing the command umask 012? Can the super user read and write the file which has 0 0 0 permission? (06 Marks)
  - c. Explain briefly the significance of the output of ls - l command. (08 Marks)
  
- 3
  - a. Explain different modes of vi editor with a neat diagram. (06 Marks)
  - b. Explain input and output redirection with an example to each. (06 Marks)
  - c. Explain find command with examples. Use find command to locate in /bin and /usr / bin all filenames that i) begin with s ii) have the extension .html and .c (08 Marks)
  
- 4
  - a. Explain how a process is created. (06 Marks)
  - b. Explain sort command with atleast four options. (06 Marks)
  - c. What do the following commands do : i) grep pattern foo | tail - n 1. (08 Marks)
    - ii) ls - l 'grep - l wait \*.c' iii) tr '[a-z]' [A-Z] < emp iv) sed - n '3, 10p' foo.
  
- 5
  - a. Explain the positional parameters in UNIX. (08 Marks)
  - b. Write a shell script to accept the filename and check the attributes of the file and display suitable message. (06 Marks)
  - c. Write an interactive shell script to list i) files ii) processes iii) users iv) date using case statement. (06 Marks)
  
- 6
  - a. Explain here document with an example. (06 Marks)
  - b. Explain built - in variables used by awk. (06 Marks)
  - c. Discuss built - in functions in awk with examples. (08 Marks)
  
- 7
  - a. Devise a perl script to convert decimal number to binary. (08 Marks)
  - b. Explain the concept of file handling in perl. (06 Marks)
  - c. Explain export and exec commands in shell script. (06 Marks)
  
- 8
  - a. What are the privileges of an administrator? Explain. (08 Marks)
  - b. Explain the following commands with examples: i) mkfs ii) fsck iii) crypt. (06 Marks)
  - c. Explain the concept of mounting and unmounting in UNIX with examples. (06 Marks)

\*\*\*\*\*

PRINCIPAL  
SIET, TUMAKURU.



# 2002 SCHEME

USN

|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

CS55

## Fifth Semester B.E. Degree Examination, Dec.09/Jan.10 UNIX and Shell Programming

Time: 3 hrs.

Note: Answer any FIVE full questions.

Max. Marks:100

Important Note: 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.  
2. Any revealing of identification, appeal to evaluator and/or equations written eg. 42+8=50, will be treated as malpractice.

- 1 a. Describe the salient features of unix operating systems. (08 Marks)  
b. With a neat diagram explain the relationship between kernel and the shell. (06 Marks)  
c. Explain the different types of files in unix. (06 Marks)
- 2 a. Discuss the various unix commands used for comparing two files. (08 Marks)  
b. Explain the need for two special files /dev/null and /dev/tty. (04 Marks)  
c. What are inodes? State the difference between hardlink and soft link. (08 Marks)
- 3 a. What are file attributes? Explain how basic file permissions can be changed. (08 Marks)  
b. What is redirection? Discuss the sources of standard input and standard output with example. (06 Marks)  
c. Describe the important system calls used for process creation. (06 Marks)
- 4 a. What is job control? Discuss the commands provided by job control shell for job controlling. (08 Marks)  
b. Discuss the cron unix command. (04 Marks)  
c. Write note on finger and talk unix commands. (08 Marks)
- 5 a. What are filters? Discuss the cut and paste filters with options. (10 Marks)  
b. Discuss the sort and tr unix filters. (08 Marks)  
c. Describe the uniq filter. (02 Marks)
- 6 a. Describe the grep filter. write the grep command to accomplish the following (Assume input file as infile)  
i) Locate SQKSena or Sa XSena  
ii) Count no of lines containing words made of numerals (not necessarily purely numeral)  
iii) Locate lines that end with 'A' followed by 5 characters  
iv) Locate lines not containing Rakhee Rakhi Raki.  
v) Locate ordinary files that are not executable  
vi) Locate lines beginning with 'A' and ending with Z. (08 Marks)  
b. What is "here document"? Discuss its use. (06 Marks)  
c. What are positional parameters? How are they useful in shell script? Discuss. (06 Marks)
- 7 a. Write a shell script that accepts rollnumber and name of student if not as command line arguments and append it to new list. Append the information of atleast 10 students (use read statement for remaining students information extraction). (10 Marks)  
b. Write a shell script which checks after everyone minute whether your friend has logged in or not. The script should report whenever he logs in along with the amount of time he was late in logging in. Input username as command line argument. (10 Marks)
- 8 a. Write a perl program to simulate a simple hand calculator using basic operations like addition, subtraction, multiplications and division. Program should take operations as command line arguments. (08 Marks)  
b. Discuss the different types of file tests that can be performed using test command. (06 Marks)  
c. What do you mean by mounting and unmounting of file system? Explain briefly. (06 Marks)

\*\*\*\*\*



PRINCIPAL  
SIET, TUMAKURU.



|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

**Fifth Semester B.E. Degree Examination, Dec. 07 / Jan. 08**  
**UNIX and Shell Programming**

Time: 3 hrs.

Max. Marks:100

Note : Answer any FIVE full questions.

1.
  - a. With a neat diagram, explain the relationship between the kernel and the shell of UNIX. (08 Marks)
  - b. Describe the salient features of UNIX operating system. (08 Marks)
  - c. With example, explain date and bc command. (04 Marks)
2.
  - a. What is parent-child relationship? With the help of diagram explain the UNIX file system tree. (06 Marks)
  - b. What are the different types of files in UNIX? Explain them briefly. (08 Marks)
  - c. Explain the absolute path name and relative path name with example. (06 Marks)
3.
  - a. Explain the different modes of operation in a Vi editor. (08 Marks)
  - b. What are standard input, standard output and standard error? Explain with respect to UNIX. (10 Marks)
  - c. Explain the following environment variables
    - i) IFS
    - ii) TERM
    - iii) PATH. (06 Marks)
4.
  - a. What is a process? Discuss the mechanism of a process creation in UNIX. (08 Marks)
  - b. How do you use the following commands in UNIX? Explain with suitable examples -  
msg, talk, mailx, pine. (08 Marks)
  - c. Write a note on finger command. (04 Marks)
5.
  - a. Explain "grep" command with all options. (06 Marks)
  - b. What is shell programming? Explain with examples how expressions are evaluated in shell programming. (09 Marks)
  - c. Explain the shell scripts used for debugging. (05 Marks)
6.
  - a. State any six built in variables in awk and explain each. (07 Marks)
  - b. Explain the shell features of "while" and "for" with syntax. (06 Marks)
  - c. What is a here document? Explain with an example. Also mention its use. (07 Marks)
7.
  - a. Explain string handling function in Perl. (06 Marks)
  - b. Explain the list and array used in Perl. Also write a Perl program to convert decimal number to binary number. (08 Marks)
  - c. Explain file handling in Perl. (06 Marks)
8.
  - a. Mention the various privileges of a super user. Discuss how a user can become a super user. (08 Marks)
  - b. Explain the facility of mounting and Un-mounting the files in UNIX. (06 Marks)
  - c. Discuss the Crypt command. (06 Marks)

\*\*\*\*\*

  
 PRINCIPAL  
 SIET, TUMAKURU.

USN

|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|

CS55

**Fifth Semester B.E. Degree Examination, June / July 08**  
**UNIX and Shell Programming**

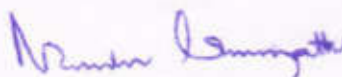
Time: 3 hrs.

Max. Marks: 100

Note : Answer any FIVE full questions.

1.
  - a. Discuss the salient features of UNIX operating system. (07 Marks)
  - b. Explain the following commands with examples:  
i) date ii) bc iii) cal. (06 Marks)
  - c. Explain in brief the different types of files in UNIX. (07 Marks)
2.
  - a. Explain the following file handling commands used in UNIX with examples:  
i) cmp ii) comm iii) diff (06 Marks)
  - b. Describe the various file attributes displayed with ls -l command. (03 Marks)
  - c. Clearly differentiate between hard links and soft links. (06 Marks)
3.
  - a. Explain the different modes in which vi editor works. (06 Marks)
  - b. What does the following command do when editing with vi?  
:4,8s/unix/UNIX/gc (02 Marks)
  - c. What are standard input, standard output and standard error? Explain with respect to UNIX operating system. (06 Marks)
  - d. Explain the following environment variables:  
i) IFS ii) PS1 iii) TERM iv) PATH. (06 Marks)
4.
  - a. What is a process? Discuss the mechanism of process creation in UNIX. (07 Marks)
  - b. Define job. How is job control done in UNIX? (07 Marks)
  - c. Explain finger, talk, and msg commands as related to communication under UNIX operating system. (06 Marks)
5.
  - a. Explain pr and tail commands with options. (08 Marks)
  - b. How does grep help in searching for a pattern? Explain its i, c, and v options. (08 Marks)
  - c. Write a note on here document. (04 Marks)
6.
  - a. Write shell script to create a menu which displays the list of files, current date, process status and current users of the system. (08 Marks)
  - b. Explain the use of test and [ ] to evaluate expressions in shell. (08 Marks)
  - c. Write a note on debugging shell scripts. (04 Marks)
7.
  - a. List and explain any six built in variable of awk. (06 Marks)
  - b. Write an awk sequence to find HRA, DA and net pay of an employee where DA is 80% of the basic, HRA is 10% of basic and net pay is the sum of basic, HRA and DA. (08 Marks)
  - c. Explain the string handling functions of Perl. (06 Marks)
8.
  - a. Write a perl script to convert a decimal number to binary. (06 Marks)
  - b. List and explain the various privileges of system administrator. (08 Marks)
  - c. Explain epio command for back up and restoring files. (06 Marks)

\*\*\*\*\*



PRINCIPAL  
SIET, TUMAKURU.



Unix:- Unix is an operating system, i.e software that manages the computer's hardware and provides a convenient and safe environment for running program.

THE UNIX COMPONENTS / ARCHITECTURE :

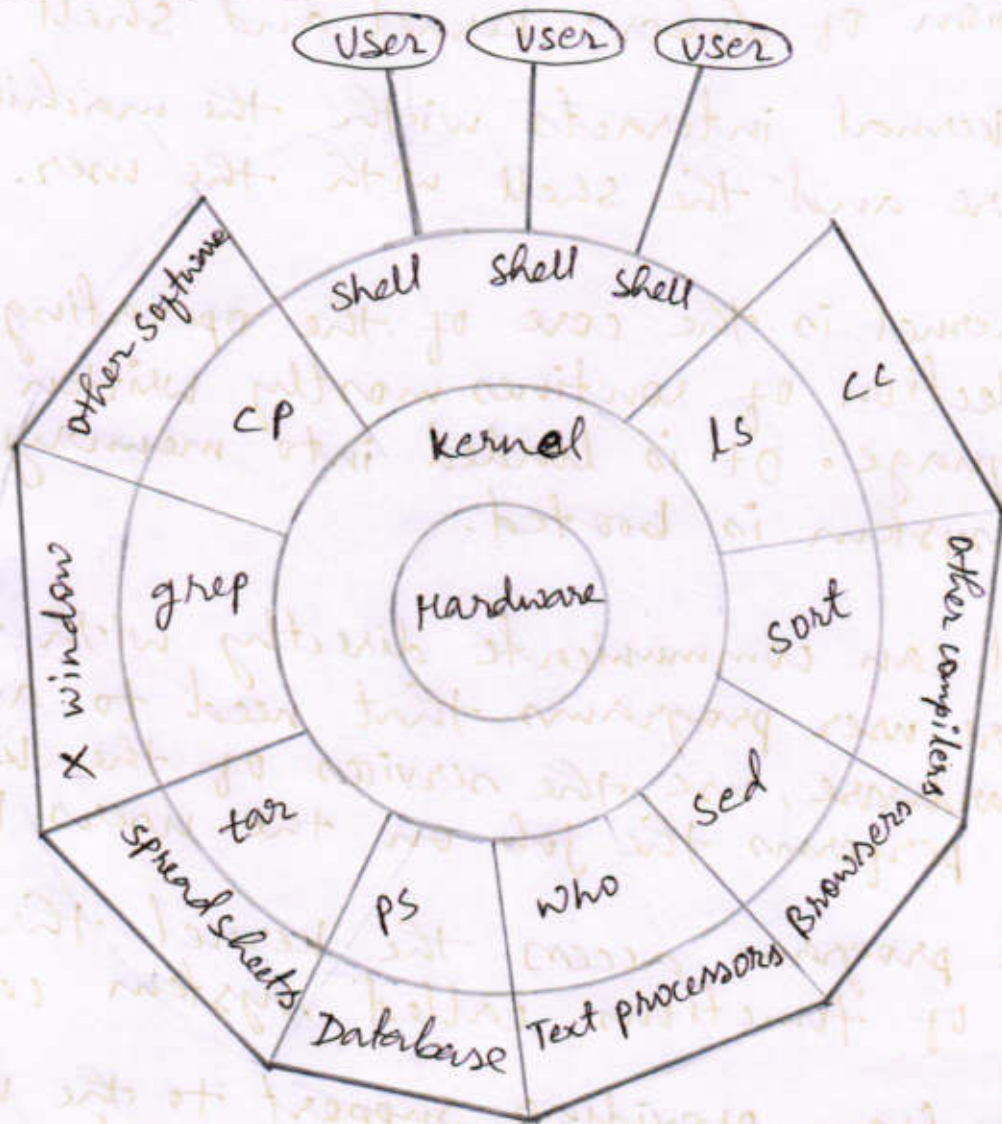


Fig: The kernel - shell relationship

The entire UNIX system is supported by hand full of essentially simple, through somewhat abstract concepts :-

\* The success of UNIX according to Thompson and Ritchie, " lies not so much in new



inventions but rather in the full exploitations of a carefully selected set of ideas.

- \* Especially in showing that they can be keys of the implementation of a small and yet powerful operating system.

Division of labor: kernel and shell

- \* The kernel interacts with the machine's hardware and the shell with the user.
- \* The kernel is the core of the operating system a collection of routines mostly written in C-language. It is loaded into memory when the system is booted.
- \* Kernel can communicate directly with the hardware user programs that need to access the hardware, use the services of the kernel, which performs the job on the users behalf.
- \* These program access the kernel through a set of functions called system calls.
- \* Apart from providing support to the user program the kernel has a great deal of house-keeping to do.
- \* It manages the system memory schedules processes, decides priorities and performs other tasks.



- \* Computers don't have any inherent capability of translating commands into action. That requires a command interpreter, a job that is handled by the "outer part" of the operating system or the shell.
- \* The shell is the interface between the user and kernel.
- \* Even though there is only one kernel running on the system, there could be several shells in action. One for each user who is logged in.
- \* When a command is entered through the keyboard, the shell thoroughly examines the keyboard input for special characters.
- \* If it finds any, it rebuilds a simplified command line and finally communicates with the kernel to see that the command is executed.

### # Features of UNIX operating system:-

- \* A multiuser system  $\begin{cases} \rightarrow \text{multiple user - separate job} \\ \leftarrow \text{single user - multiple job.} \end{cases}$
- \* A multitasking system too.
- \* The Building-Block approach.
- \* The UNIX Toolkit.
- \* pattern matching.
- \* programming facility.
- \* Documentation.



## POSIX AND THE SINGLE UNIX SPECIFICATION:

The portable operating system interface for computer environments (POSIX), were developed at the behest of the institution of IEEE. POSIX refers to operating system in general, but was based on UNIX. Two of the most cited standards from the POSIX family are known as POSIX.1 and POSIX.2. POSIX.1 specifies the C application program interface - the system calls. POSIX.2 deals with the shell and utilities.

In 2001, a joint initiative of X/Open and IEEE resulted in the unification of the two standards. This is the single UNIX specification, version 3 (SUSV3). The "write once, adopt everywhere" approach to this development means that once software has been developed on any POSIX compliant UNIX system, it can be easily ported to another POSIX-compliant UNIX machine with minimum modifications. We make references to POSIX throughout this text, but these references should be interpreted to mean the SUSV3 as well.



## Features of Unix :-

1. multiplexer System
2. multitasking System
3. Building block Approach
4. Programming facility
5. Unix tool kit
6. Documentation
7. Pattern matching

### → Multiuser System :-

Unix is a multiuser system. It permits multiple programs to run and compete for the attention of the CPU.

This can happen in two ways:

- \* multiple users can run separate jobs
- \* A single user can also run multiple jobs.

In Unix, resources are shared b/w all users. In multiuser the compiler break up of unit of time in several segments & each user allotted segment so that only point in the w/c will doing jobs of single user. the moment allotted time expires the previous allotted job is suspended by the next user job is taken

up. windows is a single user system where CPU & memory are Harddisk are all dedicated to single user process is Repeated.

### → Unix is Multitasking :-

A single user can also run the multiple jobs are called multitasking. The one job is running on the foreground and rest run in background we can switch the jobs between foreground & background



→ Building - Block Approach:-

Unix Commands are Design to perform the simple jobs. Lets us to develop the complex Commands by connecting simple Commands. Unix tools are Designed with the requirement that o/p of one be used as I/p of another tool.

The Command that can be connected this way are 'filters' because the filter manipulates data in different way.

→ Programming - facility:-

Unix Shell is also a programming language. has control structure tools and variables using all these features we can design shell scripts which are the plm that can invoke use Commands. System function be controlled & automated using these shell scripts.

→ Unix tool-kit :-

Unix System are shipped with lot of Applications. it consists of various tools like general purpose tools, Interpreters, text manipulation utilities, compilers, H/W application of system. Administration tools.

→ Documentation:-

There are online Documentation of all Commands can access this facility by using 'man' Command.

There are vast Unix resources are available IAG a document that Address common problem is as also widely used available on the net.

→ Pattern matching:-

Unix has a pattern making features. there are special characters like \*, ?, & after which are used for pattern making. these characters are called as



'meta characters'. Used for making file names. Some tools are special Expression called 'regular Expression' is termed with these 'meta characters'.

## POSIX AND SINGLE UNIX SPECIFICATION:-

Dennis Ritchie's rewritten Unix in C lost portability due to fragmentation and absence of single std.

- ✓ AT and T created SVI (System v Interface Defn) then X/Open & vendors created the X/Open portability guide (XPG). Products conforming to this specification were bounded as Unix95, Unix98, Unix03 depending on their version of specification.
- ✓ Another group of std POSIX were developed at IEEE. POSIX refers to the OS based on Unix.
- The two most cited standards from POSIX are known as POSIX-1 & POSIX-2.
- In 2001 X/Open and IEEE unified these two stds with this unified Unix specification version 3 (SVSV3).
- The write once adopt anywhere i.e. once slw has been developed on any POSIX compliant Unix s/m it can be easily ported to another POSIX compliant Unix machine with the modification.

## LOCATING COMMANDS:-

The Unix is a command-based i.e. thing happens because of the commands that we key in. Unix commands are seldom more than four characters long. All Unix commands are single word like ls, cd, who etc, their names are all in lower case, and you must start shedding your old Dos habits of being indifferent to case. For instance if you enter LS instead of ls this is how that system will respond



```
$ ls
```

```
bash: ls: command not found
```

```
$ type ls
```

```
ls is /bin/ls
```

When you execute the `ls` command the shell locates this file in the `/bin` directory and makes arrangements to execute it. `type` looks up only the directories specified in the `PATH` variable.

### THE PATH:

The sequence of directories that the shell searches to look for a command is specified in its own `PATH` variable.

Use `echo` to evaluate this variable and you'll see a directory list separated by colons.

```
$ echo $PATH
```

```
/bin:/usr/bin:/usr/local/bin:/usr/ccs/bin:/usr/local/java/bin;
```

These are six directories in this colon-separated list. To consider one, `/usr/bin` represents a hierarchy of three directory names. The first `/` indicates the top-most directory called 'root', so `usr` is below the root directory and `bin` is below `usr`.



## INTERNAL AND EXTERNAL COMMANDS:-

The commands which are built into the shell is called 'Internal Command'

When we type these commands the shell weren't takes in its path to locate it. rather it will execute it from its own set of built in commands.

Ex: Echo, cd

The commands which have independent existence in directories specified in path. the shell look in the path locate the file of the command is executed is called 'External Command'

Ex:- ls.

We can know that whether a command is built in or not we type command

```
$ type echo
```

```
echo is a shell built-in
```

```
$ type ls
```

```
ls is /bin/ls
```

Note:- we can execute more than one command in the line by separating the commands using : (semi colon)

```
$ wc note ; ls
```

→ we can group them using parenthesis & redirect their o/p to a file

```
$ (wc note ; ls) > note1
```

→ we can split a long command line into multiple lines in that case the shell a secondary prompt to indicate that and line is not complete.

Ex: `$ echo that is`  
`> a three - line`  
`> text msg;`

O/p: this is a three line text msg



## type command

- \* The unix is command based system i.e. - things happens because the user enters the command in.
- \* Usually unix commands are less than 5 characters long.
- \* All unix commands are single <sup>words</sup> like - cat, ls, pwd, date, mkdir
- \* cd, grep etc...
- \* The command names all are in lowercases.
- \* Ex:
  - ↳ \$ ls bash:
  - ↳ ls: command not found.
- \* All unix commands are file containing programmes mainly written in 'c'
- \* All unix commands are stored in directory
- \* If you want to know the location of executable programme

## Use type command

Eg: \$ type ls

ls /bin /bin /ls

- \* When you execute is command, the shell locate this files in the /bin directory & makes arrangement to execute it.
- man: Browning the manual pages online.
- man searches the manuals starting from the section of it locate a keyboard on one section it won't continue the search even keyboard occurs on another system, when keyboard is found in multiple section use section no as a argument.
- Eg: man and password.



\* This displays the documentation for a Configuration this named etc.. password from section and below all main password this will be display about the command Password.

\* A man page is divided into no. of optional section - all but generally run section in all pages are NAME, SYNOPSIS, DESCRIPTION.

NAME: presents a one-line instruction to command.

SYNOPSIS: shows the syntax used by the command. It follows the

same rules of it.

\* If a command argument is enclosed in a rectangular brackets, then it's optional otherwise argument is required.

\* The ellipse (...) implies that there can be more instances of the preceding word.

\* All character mean that only one of the options can be used.

OPTION: Shows list of all options used by commands.

DESCRIPTION: Gives identical description of command do.

EXIT STATUS: Lists possible error condition and their numeric representation.

Man command can be used to view the documentation of man command itself.

\$ man man.

we can set pages by using shell variable page and

\$ pager less : Export pager ↴

now less pager is used when we run man command.

there is another command called 'what is', which is equivalent to -R.



more: more command can be used to display the content of a file on the screen. one page at a time.

\* After each page, it stops and waits for you to tell it what to do.

\* If the file content is more, it will show the filename & percentage of file that has been viewed. --- more --- (15%)

- Syntax  
more FILENAME

## Navigation

For movement in the four direction h, j, k & l are the keys 'vi' provides to move the cursor in four direction these are placed adjacent to one another without a repeat factor they move the cursor by one position.

k - moves cursor up

j - moves cursor down

h - moves cursor left

l - moves cursor right

Repeat factor can be used as a command prefix with all four commands.

Ex: Hk moves cursor H lines up.

l takes right & h takes you left

k takes you up

J takes you down.

## word navigation

There are 3 word navigation cmds.

b - moves back to beginning of the word.



c - moves forward end of the word.

w - moves forward to beginning of the word.

Repeat factor can be used on these words also

### Searching for a Pattern

\* A pattern can be searched in a file using following two commands.

| Command  | Function                                  |
|----------|-------------------------------------------|
| /pattern | Searches forward for pattern in the file. |

### Syntax

/pattern [Enter].

### Repeating the last Pattern Search

The previous search command can be repeated using the following 2 commands.

| Command | Function                                                   |
|---------|------------------------------------------------------------|
| n       | repeats the previous search command in the same direction. |

\* 'n' repeats search in same direction of original search.

### Knowing the user Terminal, Displaying its Characteristics

\* This commands can be used to display certain features of OS running on your machine.

Ex:

```
$uname -s
sunos
```

// name of OS used by Sun solaris

```
$uname -r
5.8
```

// version of OS.



knowing your terminal.

\* this command can be used to know device name of the terminal.

Eg:- \$ tty  
/dev/pts/xyz // here terminal filename is xyz resident in pts directory // which is in /dev directory.

### Displaying and Setting Terminal Characteristics.

\* this command can be used to display and set various terminal attributes.

\* a options can be used to display the current settings.

\$ stty -a  
speed 38400 baud rows 24; columns 116;  
par=1c; quit=11; erase=1?; kill=1U;

### Changing the setting

\* To remove the backspacing, we can use the following command

\$ stty -echo

\* To remove echo command to work, we can use the following

\$ stty -echo

### Changing the Interrupt key (intr)

\* To change the interrupt setting we can use following command.

\$ stty intr 1c

### Changing the end of file key (eof)

\* To change the end of file key setting, we can use following command.

\$ stty eof 1a.



When everything else fails (same)

\* Stty also provides another argument to set the terminal

Characteristics to value that will work on most terminals.

\* Use the word same as a single argument to the command

\$ Stty same // restores sanity to the terminal.

Managing the non-uniform behaviour of Terminals and keyboards.

\* Terminal and keyboards have no uniform behavioural patterns.

\* Terminal setting directly impact the keyboard operation.

\* The following table list keyboard commands to try when

things go wrong:

Keystroke or  
Command

Function.

[Ctrl-h] Erases text.

[Ctrl-c] and Interrupts a command.

Delete

[Ctrl-D] Terminates login session.

[Ctrl-S] Stops scrolling of screen output and locks keyboard.

[Ctrl-Q] Resumes scrolling of screen output and unlocks keyboard.

[Ctrl-U] kills command line without executing it.

[Ctrl-I] kills running programme but creates a core file containing the memory image of the program.

[Ctrl-Z] Suspend process & return shell prompt.

[Ctrl-J] or [Ctrl-M] Alternative to enter.

Stty same Restores terminal to normal status.



## Types of Accounts

There are 2 types of accounts on a Unix system:

### 1) Root Account

- \* The system administrator is known as superuser or root user.
- \* The superuser has complete control of the system.

### 2) User Account

- \* User accounts provide interactive access to the system for users and group of users.
- \* General users are typically assigned to these accounts and usually have limited access to critical system files and directories.

### The Root login:-

- \* The system administrator is known as superuser or root user.
- \* The superuser has complete control of the system.
- \* The superuser can run any commands without any restriction.
- \* The job of superuser includes:
  - Maintaining user accounts.
  - providing security.
  - Managing disk space.
  - performing backups.
- \* The root account doesn't need to be separately created but comes with every system.
- \* The root account's password is generally set at the time of installation of the system and has to be used on logging in.

login: root

password: \* \* \* \* \* [enter]

# -

\* The command prompt of root is hash (#).

\* Once you login, you are placed in root's home directory "/".

\* /sbin and /usr/sbin contains administrative commands of the system.

### Su: Becoming Super User

\* Any user can acquire superuser status with the su command if they know the root password.

\* For eg, the user "kumar" becomes a superuser in this way:

```
$ su
```

```
password: ****
```

```
pwd
```

```
/home/kumar
```

\* Though the current directory doesn't change, the # prompt indicates that kumar now has powers of a superuser.

\* To be in root's home directory on superuser login, use

```
su -
```

### Creating a User's Environment?

\* users often rush to the administrator with the complaint that a program has stopped running

\* The administrator first tries running it in a simulated environment.

\* The su command with a - (hyphen) can be used to recreate the user's environment without the login password.

```
su - kumar
```



- \* This sequence executes kumar's profile and temporarily creates kumar's environment.
- \* Su runs in a separate sub-shell, so this mode is terminated by hitting [ctrl-d] or using exit.

### /etc/passwd and /etc/shadow Files

- \* There are four main user administration files:
  - 1) /etc/passwd: Keeps the user account and password information. This file holds the majority of information about accounts on the Unix system.
  - 2) /etc/shadow: Holds the encrypted password of the corresponding account.
  - 3) /etc/group: This file contains the group information for each account.
  - 4) /etc/gshadow: This file contains secure group account information.

### 1) /etc/passwd:

- \* This file contains following user account information:

#### 1) username

→ This is the name used for logging into a Unix system.

#### 2) password

→ This stores the encrypted password which looks like \*\*\*\*\*.

#### 3) UID

→ This is user's numerical identification.

→ No two users can have the same UID.

#### 4) GID

→ This is user's numerical group identification.

### 5) Comments or GICOS:

- This is contains user details or name address.
- This name is used at the front of the email address for this user.

### 6) Home Directory

- The directory where the user ends up on logging in.
- The login program reads this field to set the variable HOME.

### 7) Login Shell

- This is the first program executed after logging in.
- This is usually the shell (/bin/ksh).
- The login program reads this field to set the variable SHELL.

### 8) /etc/shadow :

- \* This file contains encrypted password of the corresponding account.
- \* This is the control file used by passwd to verify the correctness of a user's password.
- \* For every line in /etc/passwd, there's a corresponding entry in /etc/shadow.
- \* As a regular user, you do not have read or write access to this file for security reasons.
- \* Only Superuser can access this file.



# Managing Users and groups

## groupadd

- \* This command can be used to create a new group.
- \* We need to create groups before creating any account.

Syntax: `groupadd -g gid groupname`

Where `-g GID` → The numerical value of the group's ID

`groupname` → Actual group name to be created.

Eg:- `groupadd -g 199 ISE` // 199 is the GID for ISE group

- \* `/etc/group`: This file contains the group information for each account.

- \* A group usually has more than one member with a different set of privileges.

- \* Creating a group involves defining the following parameters:

- 1) A user identification number (UID) and username.
- 2) A group identification number (GID) and groupname
- 3) The home directory.
- 4) The login shell
- 5) The mailbox in `/var/mail`

- \* `/etc/passwd`: This file contains above 5 user account information.

## Commands to Add, Modify and Delete Users

- \* Following commands can be used to create and manage user accounts.

1) `useradd`    2) `usermod`    3) `userdel`

## useradd :-

\* useradd command can be used to create a new user.

### Syntax :-

```
useradd -u username -g groupname -d homedir -s shell -m accountname
```

Where -u username → you can specify a username for this account

-g groupname → specifies a group account

-d homedir → specifies home directory

-s shell → specifies the default shell

-m → creates the home directory if it doesn't exist.

accountname → Actual account name to be created.

Eg :-

```
useradd -u username -g groupname -d homedir -s shell -m accountname
```

```
useradd -u 999 -g ISE -d /home/usp -s /bin/ksh -m kumar
```

## usermod :-

\* The usermod command can be used to make changes to an existing account.

\* This command uses the same options as the useradd command.

```
usermod -s /bin/bash kumar // to set current shell to bash shell
```

```
usermod -g CSE kumar // to set current group to CSE
```

## userdel :-

\* The userdel command can be used to delete an existing user and his account.

```
userdel kumar // delete user "kumar"
```

Eg :- # useradd -u 999 -g ISE -d /home/usp -s /bin/ksh -m kumar



## \*~\* UNIX FILE \*~\*

\* The file system in UNIX is one of its simple and conceptually clean features. It lets users access other files not belonging to them, but it also offers an adequate security mechanism so outsiders are not able to tamper with a file's contents. In this chapter, you'll learn to create directories, move around within the system, and list filenames in these directories. We'll deal with file attributes, including the ones related to security, in

## \*~\* THE FILE'S \*~\*

The file is a container for storing information. As a first approximation, we can treat it simply as a sequence of characters. If you name a file foo and write three characters a, b & c into it, then foo will contain only the string abc and nothing else. Unlike the old DOS files, a UNIX file doesn't contain the eof (end-of-file) mark. A file's size is not stored in the file, nor even its name. All file attributes are kept in a separate area of the hard disk, not directly accessible to humans, but only to kernel.

UNIX treats directories and devices as files as well. A directory is simply a folder where you store filenames and other directories. All physical devices like the hard disk, memory, CD-ROM, printer and modem are treated as files. The shell is also file, and so is the kernel. And if you are wondering how UNIX treats the main memory in your system, it's a file too!

So we have already divided files into three categories:

- \* Ordinary file - Also known as that a directory is regular file. It contains only data as a stream of characters.
- \* Directory file - It's commonly said that a directory contains files and other directories, but strictly speaking, it contains their names and a number associated with each name.
- \* Device file - All devices and peripherals are represented by files. To read or write a device, you have to perform these operations on its associated file.

There are other types of files, but we'll stick to these three for the time being. The reason why we make this distinction between file types is that the significance quite different from that for a directory. Moreover, you can't directly put something into a directory file, and a device file isn't really a stream of characters. While many commands work with all types of files, some don't. For a proper understanding of the file system you must understand the significance of these files.

### Ordinary (Regular) File

An Ordinary file or regular file is the most common file type. All programs you write belong to this type. An ordinary file itself can be divided into two types.

- Text file.
- Binary file.

A Text file contains only printable characters, and you can often view the contents and make sense out of them. All C and Java program sources, shell and perl scripts are text files.



identifies a device from its attributes and then to operate the device.

The term "file" will often be used in this book to refer to any of these types, ~~how~~ though it will mostly be used to mean an ordinary file. The real meaning of the term should be evident from its context.

## Basic File Types/Categories

On most UNIX systems today, a filename can consist of up to 255 characters, permitted to use control characters or other unprintable characters in a filename. The following are valid filenames in UNIX:

last\_time list. ^V^B^D-++bcd -{[[] @#\$%\*abcd a.b.c.d.e

The third filename contains three control characters ([tab-v] being the first). These characters should definitely be avoided in filename. Moreover, since the UNIX system has a special treatment for characters like \$, -, ?, \*, & among others it is recommended that only the following characters be used in filenames:

- Alphabetic characters and numerals.
- The period (.), hyphen (-) and underscore (\_).
- A file can have as many dots embedded in its name; a.b.c.d.e is a perfectly valid filename. A filename can also begin with a dot or end with one.
- UNIX is sensitive to case; chap01, chap01 and three different filenames, and it's possible for them to coexist in the same directory.

## The Parent-Child Relationship

All files in UNIX are "related" to one another. The file system in UNIX is a collection of all of these related files organized in a



A text file contains lines of characters where every line is terminated with the newline character, also known as line feed (LF). When you press [Enter] while inserting text, the LF character is appended to every line. You won't see this character normally, but there is a command (od) which can make it visible.

A binary file, on the other hand, contains both printable and unprintable characters that cover the entire ASCII range (0 to 255). Most UNIX commands are binary files, and the object code and executables that you produce by compiling a program are also binary files. Picture, sound and video files are binary files as well. Displaying such files with a simple cat command produces unreadable output and may even disturb your terminal's settings.

## Directory File :- \*

A directory contains no data, but keeps some details of the files and subdirectories that it contains. The UNIX file system is organized with a number of directories and subdirectories.

A directory file contains an entry for every file and subdirectory that it houses. If you have 20 files in a directory, there will be 20 entries in the directory. Each entry has two components:

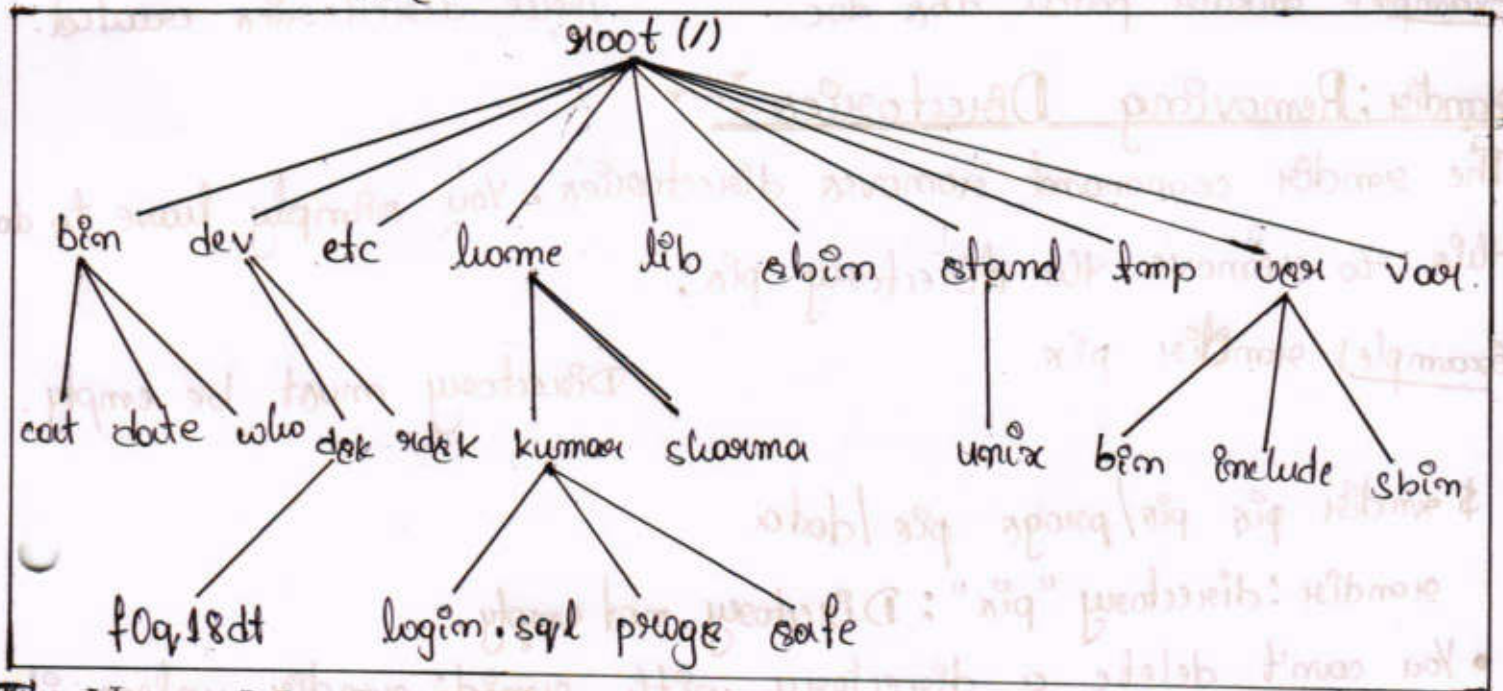
- The filename.
- A unique identification number for the file or directory.

## Device File :- \*

Device filenames are generally found inside a single directory structure, /dev. A device file is indeed special; it's not really a stream of characters. Operation of a device is entirely governed by the attributes of its associated file. The kernel



hierarchical structure. The implicit feature of every UNIX file system is that there is a top, which serves as the reference point for all files.



### The Home Variable: The Home Directory

UNIX automatically places you in a directory called the home directory. It is created by the system when a user account is opened. If you log in using the login name kumar you'll land up in a directory that could have the path name. The shell variable HOME knows your home directory.

Example:  $\rightarrow$  \$ echo \$HOME.  
/home/kumar.

First/represents the root directory.

### pwd: Checking Your Current Directory

Move around from one directory to another, but at any point of time, you are located in only one directory. This directory is known as your current directory.

Example: \$ pwd  
/home/kumar.

### Making Directories

Directories are created with the mkdir command. The command is followed by names of the directories to be created.



Example mkdir patch  
You can create a number of subdirectories with one mkdir command:

Example mkdir patch dbx doc Three directories created.

## rmmdir: Removing Directories

The rmdir command removes directories. You simply have to do this to remove the directory pi8:

Example rmdir pi8

Directory must be empty.

```
$ rmdir pi8 pi8/progs pi8/data
```

```
rmdir: directory "pi8": Directory not empty.
```

- You can't delete a directory with rmdir unless it is empty; in this case, the pi8 directory couldn't be removed because of the existence of the subdirectories, progs and data, under it.

- You can't remove a subdirectory unless you are placed in a directory which is hierarchically above the one you have chosen to remove.

```
$ cd progs
```

```
$ pwd
```

```
/home/kumar/pi8/progs
```

```
$ rmdir /home/kumar/pi8/progs
```

```
rmdir: directory "/home/kumar/pi8/progs": Directory does not exist.
```

```
$ cd /home/kumar/pi8
```

```
$ pwd
```

```
/home/kumar/pi8
```

```
$ rmdir /home/kumar/progs
```



## • MANIPULATING THE PATH:

"As its name implies, the path class is a programmatic representation of a path in the file system. A path object contains the file name and directory list used to construct the path, and is used to examine, locate, and manipulate files. A path instance reflects the underlying platform. A path is not system independent".

Path manipulation arise when user-controllable data is placed into a file (or) URL path that is used on the server to access local resources, which may be within or outside the web root".

## • ABSOLUTE PATHNAMES:

If the first character of a pathname is, the file's location must be determined with respect to root. Such a pathname, as ~~the~~ is called an absolute pathname.

Ex: `cat /home/kumar/login.sql`

No two files in a UNIX system can have identical absolute pathnames. You can have two files with the same name, but in different directories; their pathnames will also be different. Thus, the file `home/kumar/progs/c2f.pl` can coexist with the file `/home/kumar/safe/c2f.pl`.

## • USING THE ABSOLUTE PATHNAME FOR A COMMAND:

The UNIX Command runs by executing its disk file. when you specify the date command, the system has to locate the file date from a list of directories specified in the PATH variable. Since date resides in /bin and also use the absolute pathname;

\$ /bin/date ↵

Thu Sep 1 09:30:49 IST 2005.

For any command that resides in the directories specified in the PATH variables.

If executes the programs residing in some other directory that isn't in PATH, the absolute pathname then needs to be specified.

Ex: /usr/local/bin/less.

Frequently accessing programs in a certain directory, it's better to include the directory itself in PATH.

### • RELATIVE PATHNAMES:

Absolute pathname didn't use an `cd` to move to the directory `progs`. Now we use one as an argument to `cat`:

Ex: `cd progs`  
`cat login.sql`

Here, both `progs` and `login.sql` are presumed to exist in the current directory. If `progs` also contains a directory `scripts` under it, you still won't need an absolute pathname to change to that directory:

`cd progs/scripts`.

But it is not an absolute pathname because it doesn't begin with a `.`. In this example, we used a rudimentary form of relative pathnames though they are generally not labelled as such.



## • Using . and .. in RELATIVE PATHNAMES:

In a preceding example, you changed your directory from /home/kumar/pis/progs to its parent directory by using cd with an absolute pathname;

ex: cd /home /kumar /pis.

Navigation often becomes easier by using a common ancestor as reference. UNIX offers a shortcut - the relative pathname - that uses either the current or parent directory as reference, and specifies the path relative to it.

A relative pathname uses one of these cryptic symbols:

- (\*) . (a single dot) - This represents the current directory.
- (\*) .. (two dots) - This represents the parent directory.

Assuming that you are placed in /home/kumar/progs/data/text, you can use .. as an argument to cd to move to the parent directory,

```
$ pwd
/home/kumar/progs/data/text
$ cd ..
$ pwd
/home/kumar/progs/data
```

The command cd.. translates to this: "change your directory to the parent of the current directory".

For instance, to move to home also used cd/home. Alternatively, use a relative pathname:

```
$ pwd
/home/kumar/pis
$ cd .. | ...
$ pwd
/home
```

The cp command which also uses a directory as the last argument can be used with a dot:

cp .. /sharma / . profile .

This copies the file . profile to the current directory(.).  
for instance, cd prog is the same as cd ./prog.

- pwd tells you the current directory, and cd is used to change it. when used by itself, it switches to the home directory. The home directory is available in the shell variable HOME.

Ex: \$HOME /foo @ ~ /foo.

- mkdir and rmdir are used to create or remove directories. To remove a directory base with rmdir, base must be empty and you must be positioned above base.

#### • LISTING , DIRECTORY , CONTENTS :

We already used "ls" command to obtain a list of all filenames in the current directory.

\$ ls ↵

08- Packets .html

Toc .sh

Calendar

cp odos .sh

dept .lst

emp .lst

{ Numerals first  
Upper case next  
Then lowercase

It includes directories also, and if you are using Linux, you would probably see the directories and ordinary files in different colors.



Directories often contain many files, and may simply be interested in only knowing whether a particular file is available.

```
$ ls calendar
```

```
calendar
```

and if perl isn't available, the system clearly says so:

```
$ ls perl
```

```
perl: No such file or directory
```

ls can also be used with multiple filenames, and has options that list most of the file attributes.

#### • OPTIONS:

Output `mv` in Multiple columns (`-x`). It's better to display the filenames in multiple columns. Modern version of `ls` do that by default but if that doesn't happen on your system, use the `-x` option to produce a multicolumnar output:

```
$ ls -x
```

```
OS_Packets.html Toc.sh calendar Cptodos.sh
dept.lst emp.lst helpdir progs
Usdsk06x Usdsk07x Usdsk08x vx2nd06.
```

If system needs to use the `-x` option to display multicolumnar output.

And also `-f` option should be used. Combining `-x` produces a multicolumnar output

```
$ ls -fx
```

```
OS-Packets.html Toc.sh* calendar* Cptodos.sh*
dept.lst emp.lst helpdir / progs /
```

And showing hidden files also -a.

\$ ls -axf

./ .. .ixrc .kshrc  
= profile = rhosts = sh-history = xdtcup check  
.....

The file .profile contains a set of instructions that are performed when a user logs in.

The first two files (. and ..) are special directories, we used same symbols in relative pathnames to represent the current and directories.

### • Table:

| <u>Option</u> | <u>Description</u>                                                           |
|---------------|------------------------------------------------------------------------------|
| 1. -x         | Multicolumnar output                                                         |
| 2. -F         | Marks executables with *, directories with / and symbolic links with @       |
| 3. -a         | Shows all filenames beginning with a dot . and ..                            |
| 4. -R         | Recursive list                                                               |
| 5. -r         | Sorts filenames in reverse order                                             |
| 6. -l         | Long listing in ASCII collating sequence showing seven attributes of a file. |
| 7. -d dirname | Lists only dirname .                                                         |
| 8. -t         | Sorts filenames by last modification time                                    |
| 9. -lt        | Sorts listing by last modification time                                      |
| 10. -u        | Sorts filenames by last access time                                          |
| 11. -lut      | As sorted by last access time                                                |
| 12. -i        | Displays inode numbers.                                                      |



## File related commands :

1) cat : Displaying and creating files.

cat is one of the most well-known commands of the UNIX system. It is mainly used to display the contents of a small file on the terminal:

It also accepts more than one filename as arguments.

```
$ cat chap 01 chap 02
```

• cat - concatenates the two files.

2) cp : copying a file.

The cp command copies a file or a group of files. It creates an exact image of the file on disk with a different name. The syntax requires at least two filenames. The first is copied to second.

```
$ cp chap 01 unit 1
```

3) rm : Deleting files.

The rm (remove) command deletes one or more files.

• The following command deletes three files.

```
$ rm chap 01 chap 02 chap 03
```

4) mv : renaming files.

The mv command renames (moves) files. It has two distinct functions:

- It moves a group of files to a different directory.

- It renames a files (or directory).

mv doesn't create a copy of the file, it merely rename it.

```
mv chap 01 man 01
```



5) more : paging output

The man command displays its output on pages. UNIX offers the more pager as internal commands. To view the file chap 01, enter command with filename.  
\$ more chap 01.

6) file :- Knowing the file types.

UNIX provides the file command to determine the type of file, especially of an ordinary file. file correctly identifies the basic file types (regular, directory or device).

7) wc : counting lines, words and characters

Using wc command we can count lines, words and characters in an existing file.

We can use cat command to view its contents.

```
$ wc infile ↵
```

```
3 20 103 infile
```

wc counts 3 lines, 20 words, and 103 characters.

8) od : Displaying data in octal.

The od command makes nonprinting characters to be displayed / visible by displaying the ASCII octal value of its input (i.e. file). It makes newline character visible too.

9) cmp : comparing two files.

Using cmp command we can check or compare two files whether they are identical or not, so one of them can be deleted. If two files are identical, cmp displays no message.

```
$ cmp chap 01 chap 02
```



b) comm: This command is used to display the difference between the two files. It requires two sorted files, and lists the differing entries in different columns.

```
$ cat file 1 ↵
C.K. Shukla
Sumit
das

$ cat file 2 ↵
das
C.K. Shukla
Sumit
```

\*) diff: converting one file to other.  
diff is another command that can be used to display file differences. It also tells you which lines in the file have to be changed to make the two files identical.

COMPRESSING AND ARCHIVING FILES:

When sending a large file as an email attachment, it's good etiquette to compress the file first. UNIX system comes with some or all following compression and decompression utilities.

\*) gzip and gunzip (.gz): compressing and decompressing files.

gzip is a very popular program, that works with one or more filenames. It provides the extension, gz to the compressed filename and removes the original file.

```
$ gzip libc.html
```

\* gzip options: To restore the original and uncompressed file, we have two options: use either gzip -d or gunzip with one or more filenames as arguments, the .gz extension is optional.

Recursive compression (-r): Like cp, we can also descend a directory structure and compress all files found in subdirectories. We need -r option, and the arguments to gzip must comprise at least one directory.

```
$ gzip -r prog
```



\* tar: the archival program.

For creating a disk archive that contains a group of files or an entire directory structure, we need tar. The keys/options to create a disk archive,

-c → create an archive ; -x → extract files from archive  
-t → Display files in archive ; -f arch → Specify the archive arch.

\* Creating an archive (-c): To create an archive, we need to specify the name of the archive (with -f), the copy or write operation (-c) and the filenames as arguments. We can use (-v) (verbose) option to display the progress while tar works.

\* Extracting files from archive (-x): 'tar' uses the -x option to extract files from an archive. tar file is one we just used to archive these directories.  
\$tar -xvf prog.tar

\* Viewing the archive (-t): To view the contents of the archive, use the -t (table of contents) option. It doesn't extract files, but simply shows their attributes.

• Both tar and gzip can be made to behave like filters (a group of programs whose i/p and o/p are quite flexible).

iii) zip AND unzip: Compressing and Archiving Together.

Phil Katz's popular PKZIP & PKUNZIP programs are now available as zip and unzip on UNIX. Zip generally doesn't compress as much as bzip2 but it combines the compressing function of gzip with the archival function of tar.

zip is used to compress a directory structure. It requires the first argument to be compressed filename, the remaining arguments are interpreted as files and directories to be compressed.

Recursive Compression (-r): For recursive behavior, zip uses the -r option. It descends the tree structure in same way tar does



except that it also compresses files.

Using unzip: Files are restored with the unzip command, which in its simplest form, uses the compressed filename as argument. It does a nonintuitive restoration if it doesn't overwrite any existing files.

viewing the archive (-v): We can view compressed archive with the -v option. The list shows both the compressed and uncompressed size of each file in archive.

BASIC FILE ATTRIBUTES:

\* The ls command with options:-

» ls -l: listing file attributes.

This option displays most attributes of a file like its permissions, size and ownership details. The option in UNIX lingo is often referred to as the listing. Sometimes, we combine this option with other options for displaying other attributes or ordering the list in a different sequence.

```
$ ls -l
total 72.
```

|            |   |       |       |       |              |        |
|------------|---|-------|-------|-------|--------------|--------|
| -rw-r--r-- | 1 | kumar | metal | 19514 | May 10 13:45 | chap01 |
| -rw-r--r-- | 1 | kumar | metal | 4174  | May 10 15:01 | chap02 |
| drwxr-xr-x | 2 | kumar | metal | 512   | May 9 10:31  | progs  |

The first column shows the types and permissions associated with each file. The second column indicates the number of links associated with the file. The third column shows the owner of files whereas fourth column represents the group owner of the file. The fifth column shows the size of file in bytes. The sixth, seventh and eighth columns indicate the last modification time of the file, which is stored to the nearest second.

2) The -d option: Listing directory attributes.

To force ls to list the attributes of a directory, rather than its contents, we need '-d' (directory) option.



```
$ ls -ld helpdir prog <
```

```
drwxr-xr-x 2 kumar metal 512 May 9 10:31 helpdir
drwxr-xr-x 2 kumar metal 512 May 9 09:57 prog
```

Directories are easily identified in the listing by the first character of the first column, which here shows `ad`. For ordinary files, this slot always shows a `-` (hyphen) & for device files. The significance of the attributes of a directory differ a good deal from an ordinary file.

File ownership:- When you create a file, your username shows up in the third column of the file's listing; you are the owner of the file. Your group is the group owner of the file.

When the system administrator creates a user account, he has to assign these parameters to the user:

- The user-id (UID) - both its name and numeric representation.
- The group-id (GID) - both its name and numeric representation.

### FILE PERMISSIONS:-

UNIX has a simple and well-defined system of assigning permissions to files.

```
$ ls -l Chap 02 dept.lst dateval.sh
```

```
-rwxr-xr-- 1 kumar metal 20500 May 10 19:21 chap 02
-rwxr-xr-x 1 kumar metal 890 Jan 29 23:17 dateval.sh
```

First column that represents file permissions. UNIX follows a three-tiered file protection system that determines a file's access rights. Each group here represents a category and contains three slots, representing the read, write and execute permissions of the file in that order. `r` indicates read permission, which means cat can display the file, `w` indicates write permission, you can edit such a file with an editor. `x` indicates execute permission, the `-` shows the absence of the corresponding permission.



## chmod: Changing File Permissions

A file or directory is created with a default set of Permissions, and this default is determined by a simple Setting called umask.

Create a file xstart:

```
$ cat /usr/bin/startx > xstart
```

```
$ ls -l xstart
```

```
-rw-r--r-- 1 kumar metal 1906 Sep 5 23:38 xstart
```

The chmod (change mode) command is used to set the permission of one or more files for all three categories of users (users, group, and others). It can be run only by the user (the owner) and the superuser.

The command can be used in two ways

- \* In a relative manner by specifying the changes to the current permissions.
- \* In an absolute manner by specifying the final permissions.

### Relative Permission

When changing permissions in a relative manner, chmod only changes the permissions specified in the command line and leaves the other permissions unchanged.

### Syntax

chmod category operations permission filename(s)



chmod takes as its arguments an expression comprising some letters and symbols that completely describe the user category and the type of permissions being assigned or removed. The expression contains three components:

- User category (user, group, others)
- The operation to be performed (assign or remove a permission)
- The type of permission (read, write, execute)

To assign execute permission to the user of the file xstart, we need to frame a suitable expression

```
$ chmod u+x xstart
```

```
$ ls -l xstart
```

```
-rwxr--r-- 1 Kumar metal 1906 May 10 20:30 xstart
```

Permissions are removed with the - operator

```
$ ls -l xstart
```

```
-rwxr-xr-x 1 Kumar metal 1906 May 10 20:30 xstart
```

```
$ chmod go-r xstart; ls -l xstart
```

```
-rwx--x--x 1 Kumar metal 1906 May 10 20:30 xstart
```

Abbreviations Used by chmod

| Category      | Operation                       | Permission             |
|---------------|---------------------------------|------------------------|
| u - user      | + - Assigns permission          | r - Read permission    |
| g - group     | - - Removes permission          | w - write permission   |
| o - others    | = - Assigns absolute permission | x - Execute permission |
| a - all (ugo) |                                 |                        |



## Absolute Permissions:

The expression used by `chmod` here is a string of three Octal numbers.

Octal numbers use the base 8, and octal digits have the values 0 to 7. This means that a set of three bits can represent one octal digit.

- Read permission - 4 (Octal 100)
- Write permission - 2 (Octal 010)
- Execute permission - 1 (Octal 001)

| Binary | Octal | Permissions | Significance                    |
|--------|-------|-------------|---------------------------------|
| 000    | 0     | ---         | No permissions                  |
| 001    | 1     | --X         | Executable only                 |
| 010    | 2     | -W          | writable only                   |
| 011    | 3     | -WX         | writable and executable         |
| 100    | 4     | r--         | Readable only                   |
| 101    | 5     | r-X         | Readable and executable         |
| 110    | 6     | rW-         | Readable and writable           |
| 111    | 7     | rWX         | Readable, writable & executable |



## The Security Implications:

To understand the security implications behind these Permissions and the role played by chmod,

```
-rw-r--r-- 1Kumar metal 1906 May 10 20:30 xstart
```

These permissions are fairly safe; only the user can edit the file.

## Using chmod Recursively (-R)

It's possible to make chmod descend a directory hierarchy and apply the expression to every file and subdirectory.

This is done with the -R

```
chmod -R a+x shell-scripts
```

This makes all files and subdirectories found in the tree-walk executable by all users. You can provide multiple directory and filenames, and if you want to use chmod on your home directory tree, then "cd" to it.

```
chmod -R 755.
```

```
chmod -R a+x*
```

## Directory permissions:

Directories also have their own permissions and the significance of these permissions differ a great deal from those of ordinary files.

```
$ mkdir c-progs; ls -ld c-progs
```

```
drwxr-xr-x 2Kumar metal 512 May 9 09:57 c-progs
```

A directory must never be writable by group and others.



## Changing file ownership

File ownership is a feature often ignored by many users. There are two commands meant to change the ownership of a file or directory -

- chown
- chgrp

UNIX systems differ in the way they restrict the usage of these two commands.

- \* Only the system administrator's can change a file's owner with chown.
- \* The restrictions are less severe when it comes to changing groups with chgrp. On other systems only the owner can change both.

### chown: changing the file owner:

Consider the behaviour of BSD-based system chown that has been adopted by many systems. The command is used in this way.

chown options owner[:group] file(s)

chown transfers ownership of a file to a user, and it seems that it can optionally change the group as well. The command requires the user-id (UID) of the recipient, followed by one or more filenames. Changing ownership requires superuser permissions

\$ su

Password: \*\*\*\*

# -



## chgrp: changing group owner.

The group owner of a file is the group to which the owner belongs. The `chgrp` (change group) command changes a file's group owner.

`chgrp` shares a similar syntax with `chown`.  
for ex:

```
$ ls -l dept.lst
```

```
-rw-r--r-- 1 Kumar metal 139 June 8 16:43 dept.lst
```

```
$ chgrp dba dept.lst ; ls -l dept.lst
```

```
-rw-r--r-- 1 Kumar dba 139 Jun 8 16:43 dept.lst
```

This command will work on a BSD-based system.

## chown: changing the file owner.

Consider the behavior of BSD-based system `chown` that has been adopted by many systems. The command is used in this way:

`chown options owner[:group] files`

`chown` transfers ownership of a file to user, and it seems that it can optionally change the group as well. The command requires the user-id (UID) of the recipient, followed by one or more file names. Changing ownership requires superuser permission.

```
$ su
Password: *****
#
```



# The vi Editor

Module 3

## The vi editor basics

Let us invoke vi with the filename sometext;  
vi sometext

- The vi presents a full screen with the filename shown at the bottom with the qualifier [Newfile].
- The cursor is positioned at the top and all remaining lines of the screen (except the last) show a~.
- The last line is reserved for commands that we can enter to act on the text.
- We are now in the command mode, one of the three modes of vi editor. This is the mode where we can pass commands to act on text.
- Moving cursor to next line, deleting a line can be done in command mode.
- In input mode, for text editing, first press the key marked i to input text. The key entered will then show off on the screen as text input.
- After text entry is complete, cursor is moved to current line, [backspace] and [esc] key can be used to wipe out and revert to command mode.
- The entered text is stored in temporary storage called buffer and to save it we switch to ex mode from the command mode by entering a:(coln), enter x and press [Enter].  
:x[Enter]  
"some text" 6 lines, 232 characters  
\$- // quits editor - back to shell prompt.
- The file is saved on disk and vi returns the shell prompt. To modify this file, we'll have to invoke vi sometext again.



## The .exrc File

- The default behavior of vi is adequate for novices, but as we get comfortable with it, we'll feel the need to customize it to behave in a way that makes writing programs and documents easier.
- vi reads the file `$HOME/.exrc` on startup. If it doesn't show up this file in our home directory then we can create or copy one.
- Many ex mode commands can be placed in the file so they are available in every session and we can redefine keys to behave differently.

## Different modes of vi

- (i) Command mode - It is the default mode of the editor where every key pressed is interpreted as a command to run on text. We have to be in this mode to copy & delete text.
  - (ii) Input mode - Every key pressed after switching to this mode actually shows up as text. This mode is invoked by pressing one of the keys.  
Pressing [Esc] in this mode takes vi to command mode.
  - (iii) Ex mode) or Last Line Mode - The mode used to handle files (like for saving and perform substitutions).  
Pressing a: in the command mode invokes this mode. You then enter an ex mode command by [Enter].
- Thus, command mode, input mode and ex mode are the three modes of vi.



## Input mode commands

### - Insertion of Text (i and a)

The command `[i]` inserts text to left of cursor (existing text shifted right)

The command `[a]` appends text to right of cursor.  
With `i` and `a`, you can append several lines of text.

### - Insertion of Text at Line Extremes (I and A)

`[I]` inserts text at beginning of line while `[A]` appends text at end of line.

### - Opening a New Line (o and O)

`[o]` opens line below whereas `[O]` opens line above

### - Replacing Text (r, s, R and S)

`[r]` replaces single character under cursor with `ch` and  
`[R]` replaces text from cursor to right (existing text overwritten)  
`[s]` replaces single character under cursor with any number of characters while `[S]` replaces entire line.

## The ex mode commands

### - Saving your work

`[:w]` command saves file and remains in editing mode

### - Saving and Quitting

`[:x]` saves file and quits editing mode. `[:wq]` also does the same. `[:w n2w.p1]` is like save As... in MS Windows

`[:w! n2w.p1]` it is also same, but overwrites existing file

### - Aborting editing

`[:q]` quits editing mode when no changes are made to file

`[:q!]` quits editing mode but after abandoning changes



## - Writing selected lines

`:n1, n2w build.sql` writes lines n1 to n2 to file build.sql

`:.w build.sql` writes current line to file build.sql

`:$w build.sql` writes last line to file build.sql

`!:cmd` Runs cmd command and returns to command mode

## - Escape to the UNIX shell (:sh and [Ctrl-Z])

`:sh` escapes to UNIX shell

## - Recovering from a Crash (:recover and -r)

`:recover` recovers file from a crash

## Navigation

### - Movement in the four directions (h, j, k and l)

`k` moves cursor up      `j` moves cursor down

`h` moves cursor left    `l` moves cursor right

### - Word navigation

`b` Moves back to beginning of word

`e` Moves forward to end of word

`w` Moves forward to beginning of word

### - Moving to Line Extremes

`0` (zero) or `1`    `30l` moves cursor to column 30

`$` moves to end of line

### - Scrolling

`[Ctrl-F]` scrolls forward

`[Ctrl-B]` scrolls backward

`[Ctrl-d]` scrolls half page forward

`[Ctrl-u]` scrolls half page backward

### - Absolute Movement (G)

`G` goes to the end of file



## Repeating the last command (.)

- The dot (.) command is used for repeating both input and Command mode commands that perform editing tasks.
- The . command can be used to repeat only the most recent editing operation - be it insertion, deletion or any other action that modifies the buffer.

## Searching for a pattern (/ and ?)

/pattern [Enter] searches forward when one is looking for the string pattern.

?pattern [Enter] searches backward for the most previous instance of the pattern.

n repeats search in same direction along which previous search was made while N repeats search in direction opposite to that along which previous search was made

## Search and Replace (:s)

We can perform search and replace in execute mode using :s. Its syntax is,

: address / source\_pattern / target\_pattern / flags

: 1, \$s / director / member / g can also use % instead of 1, \$

: 1, 50s / unsigned / g deletes unsigned between in lines 1 to 50

: 3, 10s / director / member / g substitute lines 3 through 10

: s / director / member / g only the current line

: \$s / director / member / g only the last line



Interactive substitution: sometimes you may like to selectively replace a string. In that case, add the `c` parameter as the flag at the end:

```
: I, $s/director/member/gc
```

Each line is selected in turn, followed by a sequence of prompts in the next line, just below the pattern that requires substitution. The cursor is positioned at the end of this prompt sequence, waiting for your response.

The `ex` mode is also used for substitution. Both search and replace operations also use regular expressions for matching multiple patterns.

### The map and abbr commands

`:map` followed by a space, a key, another space, and a macro definition maps a complex command or series of commands to that key.

```
Example:- :map @ I/* <ctrl-v> Esc>A
```

### Insert mode abbreviations (:ab)

`:ab` followed by a space, a short abbreviation (no spaces in the abbreviation), another space, and a definition for the abbreviation to make a macro that will self expand when typed in insert mode.

```
Example:- :ab ilv I Love Vi followed by hitting
[Enter] will cause ilv to be a macro which self expands to the phrase I Love Vi when typed in insert mode.
```



## ★ The SHELL'S INTERPRETIVE CYCLE

when you log on to a Unix Machine you 1<sup>st</sup> see a prompt Unix Command is in fact running at the terminal.

But this Command is special; it's with you all the time and never terminates unless you log Out. This is the SHELL.

If you provide this Input in the form of a ps Cmd

## ★ Summarized form of activities of Shell

- The shell issues the prompt and waits for you to enter a Command.
- After a Command is Entered, the shell scans the Command for Metacharacters and expands abbreviation to recreate simplified Command line.
- It then passes on the Command line to the Kernel for Execution.
- The shell waits for the Command to complete and normally can't do any work while the Command is Running.
- After Command Execution is complete the prompt reappears and the shell returns to its waiting role to start the next Cycle.

## ★ Pattern Matching - The WILD-CARDS

The metacharacters that are used to construct the generalised pattern for matching file name belong to category called Wild Cards.



## The shell's wild-Cards,

### wild Cards,

|        | <u>Matches,</u>                                                                            |
|--------|--------------------------------------------------------------------------------------------|
| *      | → Any No. & Character Including none                                                       |
| ?      | → A Single Character.                                                                      |
| [ijk]  | → A Single Character either i, j or k                                                      |
| [x-z]  | → A single Character that is written within the ASCII Range of Character x and Character z |
| [!ijk] | → A single Character that is not an i, j or k.                                             |
| [!x-z] | → A single Character that is not within ASCII Range of Character x and z (not inside)      |

## Redirection: The Three Standard files,

\* Before we commence our discussion on Redirection let's understand the term "terminal" - It is a generic name that represents the screen, display on keyboard. Just as we refer to a dictionary as a file.

\* The special files are the actually scheme of character which many command see as i/p & o/p. A scheme is simply a sequence of bytes.

\* Each stream is associated with a default device -

→ Standard I/P :- The file representing input which is connected to keyboard.

→ Standard Output :- The file representing output which is connected to the display.

→ Standard Error :- The file representing error messages that emanate from the command or shell. This is also connected to



The Display:  
Standard Input :- we have used the cat and we command have an add to read this files. They read the file representing the standard input.

- \* It can represent three input sources
- The keyboard, the default source.
- A file using redirection with symbol < (a meta character)
- Another program using a pipeline

when you use wc without argument and have no special symbol like < and | in the command line wc obtains its input from the default source (keyboard) and mark the end of i/p with [ctrl-d].

\$ wc  
 standard input can be redirected  
 It can come from file  
 or pipeline  
 [ctrl-d]

Standard Output :- all command displaying output on the terminal actually write to the standard output files as a stream of character, and not directly to the terminal as such

→ there are three possible destinations of the of this stream

- \* The Terminal, the default destination



→ A file using redirection symbol > and >>

→ As input to another program using a pipeline  
(to be taken up later.)

```
$ wc sample.txt > newfile // nothing appears on the
8 14 71 sample.txt screen
```

### How it works

1. On seeing the >, the shell opens the disk file, newfile, for writing.
  2. It unplugs the standard output file from its default destination and assigns to newfile.
  3. wc (and not the shell) opens the file sample.txt for reading.
  4. wc writes to standard output which has earlier been reassigned by the shell to newfile.
- \* Redirection also becomes a useful feature when concatenating the standard o/p of a no. of files

```
$ cat *.c > c-prog-all.txt
```

### Standard Error

Before we proceed any further, you should know that end of the three standard files is represented by a number, called a file description.

The kernel maintains a table of the file description for every process running in the system.



The 1<sup>st</sup> three slots are generally allocated to the three standard streams in this manner:

- 0 - Standard Input
- 1 - Standard Output
- 2 - Standard Error

\* Redirecting std Error Requires use of the 2> symbol  
 \$ cat foo 2> errorfile  
 \$ cat errorfile  
 cat: cannot open

\* Now that you have separate symbol for o/p & error streams, you can redirect them separately  
 foo.sh 1> bar1 2> bar2

PIPES :- Standard Input and Output Constituent two separate streams that can be individually manipulated by the shell.

\* You know the who command produce a list of users  
 One uses few lines 'Let's use redirection to save this o/p in a file':  
 \$ who > users.txt  
 \$ cat users.txt.

\* \$ who | wc -l  
 No. Intermediate file created  
 Hence the output of who has been passed directly to the input wc and who is said to be piped to wc.



\* Tree Command :- It is an External Command and not a feature of the shell. It handles a character stream by duplicating its input.

\* Can be placed anywhere in a pipeline. tree doesn't perform any filtering action on its input.

tree does not perform any filtering action on its input; it gives out exactly what it takes.

The following command sequence uses tree to display the output.

\* Command Substitution

The shell enables one or more command arguments to be obtained from the standard output of another command. This feature is called command substitution.

For Ex : to display today's date with a statement like this:

The date today is Sat Sep 7 19:01:16 IST 2002

The last part of the statement represents the output of the date command, with command substitution its simple matter. Use the expression date as an argument to echo.

```
$ echo The date today is `date`
```

The date today is Sat Sep 7 19:01:56 IST 2002



## MODULE - 3

**grep:** This command searches for pattern in one or more filename on the standard input- if no filename is specified.

Let's use grep to display lines containing the string sales from the file emp.lst, the content of this files are shown.

```
$ grep "sales" emp.lst
```

```
2533 | A.K. Sharma | g.m. | sales | 12/12/52 | 600
```

Because grep is also a filter, it can search its standard input- for the pattern and also save the standard output- in a file.

```
who | grep Kumar > foo
```

When grep is used with multiple filename it displays the filename along with the output.

```
$ grep "director" emp1.lst emp2.lst
```

```
emp1.lst: 1006 | Lalit | director | sales | 03/09/88
```

```
emp2.lst: 9876 | Jai | director | production | 12/03/50
```

### grep Options:

| Option | Significance.                                                                                                             |
|--------|---------------------------------------------------------------------------------------------------------------------------|
| -i     | → Ignores case for matching                                                                                               |
| -v     | → Don't display lines matching expression                                                                                 |
| -n     | → Displays line number along with lines                                                                                   |
| -c     | → Displays count of number of occurrence                                                                                  |
| -l     | → Displays lists of filenames only                                                                                        |
| -e ext | → Specific expression with this option. Can use multiple times. Also used for matching expression beginning with a Hyphen |
| -x     | Matches pattern with entire line.                                                                                         |

|         |                                                  |
|---------|--------------------------------------------------|
| -f file | Take patterns from file, one per line            |
| -E      | Treat patterns as an extended regular expression |
| -F      | Matches multiple fixed strings.                  |

### BASIC REGULAR EXPRESSIONS (BRE) -

| Symbol | Matches                                             |
|--------|-----------------------------------------------------|
| *      | Zero or more occurrences of the previous character. |
| g*     | Nothing or g, gg, ggg etc                           |
| .      | A single character                                  |
| .*     | Nothing or any number of characters                 |
| [pqn]  | A single character p, q or n                        |
| [1-3]  | A digit b/w 1 and 3                                 |
| [^pqn] | A single character which is not a p, q, n           |
| ^pat   | Pattern pat at beginning of line                    |
| pat\$  | Pattern pat at end of line                          |
| ^\$    | Lines containing nothing.                           |

### THE CHARACTER CLASS:

A regular expression let you specify a group of characters enclosed within a pair of non-angled brackets [], in which case the match is performed for single character in the group. This form resembles the one used by Shell's wild-card.

[na] match either an n or an a. The meta-character and can be now used to match Agarwal and agrawal. The following regular expression.

[aA]g[an][an]wal

\$ grep "[aA]g[an][an]wal" emp.lst

3564 | sudhin Agarwal | executive | 07/06/49 | 7500

0110 | vk. agrawal | g.m. | marketing | 12/31/40 | 9000



## THE \*:

The \* (asterisk) refers to the immediately preceding character.

The pattern  $g^*$  matches the <sup>single</sup> character  $g$  on any number of  $g$ 's. It also matches a null string.

$g \quad gg \quad ggg \quad \dots$

Mark the keyword "zero or more occurrences of the previous character" that are used to describe the significance of \*. Observe carefully the regular expression.

$[aA]gg^*[om][am]wal$

$\$grep "[aA]gg^*[om][am]wal" emp.txt$

3456 | Anil aggarwal | manager | 1am | 05/01/99 | 5000

8564 | Sudhin agnawal | g.m. | marketing | 12/31/40 | 9000

## THE DOT:

A  $(.)$  dot matches a single character. The dot along with the  $*$  ( $.^*$ ) constitutes a very useful regular expression.

$\$grep "j.*saxena" emp.txt$

3345 | j.b.saxena | g.m. | marketing | 03/12/45 | 8000

## # EXTENDED REGULAR EXPRESSIONS (ERE) AND egrep.

ERE make it possible to match disjunction pattern with a single expression.  $egrep$  is an extended version of  $grep$  or  $egrep$  is equal to  $grep -E$ .

## THE + and ?

ERE includes 2 special characters  $+$  and  $?$ . They are often used in place of  $*$  to restrict matching scope.

$+$  - Matches one or more occurrences of previous character.

$?$  - Matches zero or one occurrences of previous character.



\$egrep '[A-Z][a-z]\*' emp.txt

9456 | anil aggarwal | manager | sales | 01/05/59 | 5000

8564 | Sudhin Ag<sup>g</sup>arwal | executive | personnel | 06/07/47 | 7500

The + is a pretty useful character too. when you are looking for a multiword string like #include <stdio.h>, but don't know how many spaces separate the #include and <stdio.h> you can use the expression #include + <stdio.h>

## # Matching Multiple Patterns (|, and)

The | is the delimiter of multiple patterns. Using it we can locate both sengupta and dasgupta from file.

\$egrep 'sengupta|dasgupta' emp.txt

9365 | barun sengupta | director | personnel | 11/05/47 | 7500

1265 | s.m. dasgupta | manager | sales | 12/09/63 | 5600

The GRE thus handles the problem easily but offers an even better alternative. The characters (and), let you group patterns, and when you use the | inside the parenthesis you can frame an even more compact pattern

\$egrep '(Sen|das)gupta' emp.txt

9365 | barun sengupta | director | personnel | 11/05/47 | 7500

1265 | s.m. dasgupta | manager | sales | 12/09/63 | 5600

## # BASIC REGULAR EXPRESSIONS REVIEWED.

The 3 types of expressions are:

- The Repeated pattern: This uses a single ~~word~~ symbol and to make the entire source pattern appear at the destination also.
- The Interval regular expression (IRE) - This expression encloses the character { and } with a single or a pair of numbers between them.
- The tagged regular expression (TRE) - This expression groups patterns with ( and ) and represents them at the destination.



## # THE REPEATED PATTERN (&)

We sometimes encounter situation when the source pattern also occurs at the destination. We can then use the special character & to represent it. All of these commands do the same things.

```
sed 's/director/executive director/' emp.lst
```

```
sed 's/director/executive &/' emp.lst
```

```
sed 's/director/s/executive &/' emp.lst
```

All of these commands replace director with executive director. The & known as the repeated pattern.

## # INTEGER REGULAR EXPRESSION (IRE):

We have matched a pattern at the beginning and end of a line. But what about matching it at any specific location. Sed and grep also use the IRE that uses an integer to specify the number of characters preceding a pattern. The IRE uses an escaped pair of curly braces and takes three forms:

→  $ch\{n\}$  - The metacharacter  $ch$  can occur  $n$  times.

→  $ch\{m,n\}$  - Here,  $ch$  can occur between  $m$  and  $n$  times.

→  $ch\{m,\}$  - Here,  $ch$  can occur at least  $m$  times.

Let's consider this small telephone directory where the person has either a wired phone (8 digit) or mobile (10 digit)

```
$ cat teledir.txt
```

```
Jai Sharma 25853670
```

```
Anil Aggarwal 9876543210
```

Let's use grep to select who have mobile phones.

```
$ grep '[0-9]\{10\}' teledir.txt
```

```
Anil Aggarwal 9876543210
```

Let's now consider the 2nd form of IRE. Since this matches a pattern within a "zone" we can display the listing for those files







# Shell Programming

module-04

## Shell scripts:-

When a group of commands have to be executed regularly, they should be stored in a file, & the file itself executed as a shell script (or) shell program. Though it's not mandatory, we normally use the .sh extension for shell scripts. This makes it easy to match them with wild-cards.

Shell scripts are executed in a separate child shell process, & this sub-shell need not be of the same type as your login shell. In other words, even if your login shell is Bourne, you can use a Korn sub-shell to run your script. By default, the child & parent shells belong to the same type, but you can provide a special interpreter line in the first line of the script to specify a different shell for your script.

Use your vi editor to create the shell script. sh - The script runs three echo commands & shows the use of variable evaluation & command substitution. It also prints the calendar of the current month.

Use your vi editor to create the shell script. sh - The script runs three echo commands & shows the use of variable evaluation & command substitution. It also prints the calendar of the current month.

Use your vi editor to create the shell script. sh - The script runs three echo commands & shows the use of variable evaluation & command substitution. It also prints the calendar of the current month.

```
#!/bin/sh
```

```
script.sh: Sample shell script
```

```
echo "Today's date: `date`"
```

```
echo "This month's calendar:"
```

```
cal `date` +%m %Y
```

```
echo "my shell: $SHELL"
```



Note the comment character (#) that can be placed anywhere in a line; the shell ignores all characters placed on its right. However, this doesn't apply to the first line which also begins with a #. This is the interpretation line that was mentioned previously.

To run the script, make it executable first & then invoke the script name:

```
$ chmod +x script.sh
```

```
$./script.sh
```

today's date: Mon Jan 6 10:02:42 IST 2003

this month's calendar:

January 2003

| Su | Mo | Tu | We | Th | Fri | Sa |
|----|----|----|----|----|-----|----|
|    |    |    | 1  | 2  | 3   | 4  |
| 5  | 6  | 7  | 8  | 9  | 10  | 11 |
| 12 | 13 | 14 | 15 | 16 | 17  | 18 |
| 19 | 20 | 21 | 22 | 23 | 24  | 25 |
| 26 | 27 | 28 | 29 | 30 | 31  |    |

my shell: /bin/sh

The script is too simple to need any explanation -- no inputs, no command line arguments & no control structures. We'll be progressively adding these features to our future scripts.

When used in this way, the Bourne sub-shell opens the file but ignores the interpreter line. The script doesn't need to have execute permission either. We'll make it a practice to use the interpreter line in all our scripts.



## read: Making scripts Interactive

The read statement is the shell's internal tool for taking input from the user, i.e., making scripts interactive. It is used with one or more variables. Input supplied through the standard input is read into these variables, when you use a statement like,

```
read name.
```

The script, `emp1.sh`, uses read to take a search string & file name from the terminal.

Shell scripts accept comments prefixed by # anywhere in a line, you know what the sequence \c does. Run the script & specify the input when the script pauses twice:

```
!/bin/sh
```

```
emp1.sh; Interactive version - uses read to take 2 inputs
```

```
echo "Enter the pattern to be searched: \c"
```

```
read pname
```

```
echo "Enter the file to be used: \c"
```

```
read flname
```

```
echo "searching for $pname from file $flname"
```

```
grep "$pname" $flname
```

```
echo "Selected records shown above"
```

```
$ emp1.sh
```

```
Enter the pattern to be searched: director
```

```
Enter the file to be used: emp1.txt
```

```
searching for director from file emp1.txt
```

```
9876 | jai sharma | director | production | 12/03/54
```

```
8365 | harun sengupta | director | personnel | 11/05/47
```

```
selected records shown above.
```



A single read statement can be used with one or more variables to let you enter multiple arguments: read pname fl name.

## Using Command Line Arguments

Like UNIX commands, shell scripts also accept arguments from the command line. They can, therefore, run noninteractively & be used with redirection & pipelines. When arguments are specified with a shell script, they are assigned to certain special "variables" - rather positional parameters.

$\$*$  → It stores the complete set of positional parameters as a single string.

$\$\#$  → It is set to the no of arguments specified. This lets you design scripts that check whether the right no of arguments have been entered.

$\#\! / \text{bin} / \text{sh}$

$\# \text{cmd} \& . \text{sh}$ ; Non-interactive version - uses command line arguments.

$\#$

echo "Program:  $\$0$   
The no of arguments specified is  $\$\#$   
The arguments are  $\$*$ "

grep " $\$1$ "  $\$2$

echo "\n Job over"

When arguments are specified in this way, the 1st word is assigned to  $\$0$ , the 2<sup>nd</sup> word to  $\$1$ , & the 3<sup>rd</sup> word to  $\$2$ . You can use more positional parameters in this way up to  $\$9$ . All assignments to positional & special parameters are made by the shell. You can't really tamper with their values, except in an indirect fashion, but you can use them to great advantage in several ways. They will be used over & over again in shell scripts, & are listed Table.



## The Logical Operators && And ||

### Conditional Execution

The script `emp1.sh` has no logic to prevent display of the message, selected lines shown above when the pattern search fails. That is because we didn't use `grep`'s exit status to control the flow of the program. The shell provides 2 operators that allow conditional execution - the `&&` & `||`, which typically have this syntax:

```
cmd1 && cmd2
```

```
cmd1 || cmd2
```

The `&&` delimits 2 commands; the command `cmd2` is executed only when `cmd1` succeeds. You can use it with `grep` in this way:

```
$ grep 'director' emp.lst && echo "pattern found
in file"
```

```
1006 | chanchal singh wil director | date 10 3/09/23
```

```
$ grep 'manager' emp.lst || echo "Pattern not
found"
```

```
Pattern not found.
```

This segment makes rudimentary decisions which the previous scripts couldn't. In fact, the `&&` & `||` operators are recommended for making simple decisions. When complex decision making is involved, they have to make way for the `if` statement.



## Shell Parameters

## Significance

\$1, \$2, ...

Positional parameters representing command line arguments

##

No of arguments specified in command line.

\$0

Name of executed command.

\*\$\*

Complete set of positional parameters as a single string.

"\$@"

Each quoted string treated as a separate argument.

\$\_

Exit status of last command.

\$\$

PID of the current shell.

\$\_

PID of the last background job.

## Exit & Exit status of Command.

C programs & shell scripts have a lot in common & one of them is that they both use the same command to terminate a program. It has the name `exit` in the shell & `exit()` in C. We'll take up the `exit` func in Part II of this book, but in this section, we'll examine the shell's `exit` command. The command is generally run with a numeric argument:

`exit 0`

used when everything went fine

`exit 1`

used when something went wrong.

These are a very common exit values. You don't need to place this statement at the end of every shell script because the shell will do so when script execution is complete. Rather it's quite often used with a command when it fails.



# UNIX AND SHELL PROGRAMMING P-4

FILE System :- every file system has a separate free links directory structure, each with its own root directory. if you have three file systems on one ~~different~~ hard disk, then they will have three separate root directories.

## Hard links

A file can have multiple filenames. also; then it is said that the file has more than one links. The link count is displayed in the second column of the listing. This count is normally 1, but the following files have two links.

```
~$ ll -lx -xr -xr -d. kumar metal 103 Jul 13 01:36 karnot.gm
- rwxr-xr-x -xr -d kumar metal 103 Jul 13 01:36 yestoff.gm
```

## Soft links (symbolic links)

unlike the hard link, a symbolic link doesn't have the file's contents, but simply provides the path name of the file that actually has the contents.

```
$ ln -s note note.lnk
```

```
$ ll -li note note.lnk
```

```
lrwxr-xr-x 1 kumar group 80 Aug 16 14:50 note
```

```
lrwxr-xr-x 1 kumar group 4 Aug 16 15:07 note.lnk -> note
```

where  $\pi$  indicates symbolic link file category  $\rightarrow$  indicates note.lnk containing the path name for the file name note. size of symbolic link is only 4 bytes.

## Simple filters

Filters are the commands which accept data from standard input, manipulate it and write the results to standard output.

## Few simple filters

o head :- displays the beginning of the file

```
$ head cmd.lst
```

we can specify -n for the line count.

\$ head -n 3 emp.lst

will display the first three lines of the file.

tail ! - displays the end of a file.

\$ tail . displays

\$ tail . emp.lst -

specifies -n for the line count

\$ tail -n 3 emp.lst.

displays the last three lines of the file.

cut ! - splitting a file vertically

it is used for splitting the file vertically

\$ head -n 5 emp.lst | tee short.lst.

will select the first five lines of emp.lst and save it to

short.lst

we can cut by using -c option with the list of column numbers, delimited by a comma.

Paste !. Pasting files.

when we cut with cut, it can be pasted back with the paste command, vertically rather than horizontally

\$ paste cut.lst 1 cut.lst 2

we can specify one or more delimiters with -d.

\$ paste -d " | " cut.lst 1 cut.lst 2.

Joining Files (-s)

let us consider that the file address book containing the details of three persons.

\$ paste -s address book

- to print in one line only

\$ paste -s -d " | \n " address book

- are used in a circular manner



sort  
ordering - a file.

Sortness is the ordering of data in ascending or descending sequence. The sort command orders a file and by default the entire line is sorted.

By default, sortness sequence can be altered by using certain options.

We can also sort one or more keys (fields) or use a different ordering rule.

The important sort options are

description

Options

- t char - uses delimiter char to identify
- k n - sorts on <sup>field</sup> nth field
- k m, n - starts sort on mth field and ends sort on nth field.
- k m, n - starts sort on nm column of mth field
- u - removes repeated lines
- n - sorts numerically
- r - reverses sort order
- f - reverses fold, lowercase to equivalent uppercase.
- m - list merges sorted file in list
- c - check if file is sorted
- o filename - places output in file name

ex.

To sort on the second field i.e., name, use -k2 in sort command

\$ sort -t "|" -k2 shortlist

|    |              |          |           |            |       |
|----|--------------|----------|-----------|------------|-------|
| 20 | A. K. Shukla | manager  | sales     | 10/10/2015 | 6000  |
| 15 | Jai          | director | marketing | 2/2/2016   | 7000  |
| 18 | hemish       | chairman | sales     | 1/1/2015   | 15000 |

Sort order can be reversed with the r-option.  
 \$ sort -t "|" -r -k2 sortlist or \$ sort -t "|" -k2 sortlist  
 18 | Kumar | chairman | sales | 11/1/2015 | 15000  
 15 | Jain | director | marketing | 2/1/2016 | 7000  
 20 | A.K. Shukla | manager | sales | 12/1/2015 | 6000

\$ umask !- file

default, file and directories. Permissions  
 when we create files and directories, the permissions  
 assigned to them depend on the system's default settings.  
 The unix system has the following default permissions

for all files and directories  
 rw - rw - rw (octal 666) for regular files  
 rw rwxrwx (octal 777) for directories

\$ umask 022.

this becomes 644 (666 - 022) for ordinary files and  
 755 (777 - 022) for directories

\$ umask 000. this indicates we are not restricting  
 anything and the default permissions will remain  
 unchanged.



FILE INODE

The inode is a data structure in a Unix-style file system that describes a filesystem object such as a file or a directory. Each inode stores the attributes and disk block location of the object data. Filesystem object attributes may include metadata as well as owner and permission data.

An inode is a data structure on a Unix/Linux file system. An inode stores basic information about a regular file, directory, or other file system object. You can use following two commands to display an inode.

- a) ls Command: List directory contents.
- b) stat Command: Display file or file system status.

a) ls Command:

Type ls command with option: `$ ls -i /etc/passwd.`

output: 752010 /etc/passwd.

∴ 752010 is index number (inode) for /etc/passwd file.

by STAT Command:

\$ stat /etc/passwd.

We can use inode numbers to delete or search a file.

The internal representation of a file is called an "inode", which contains all the required information and description of the file data and its layout on disk.

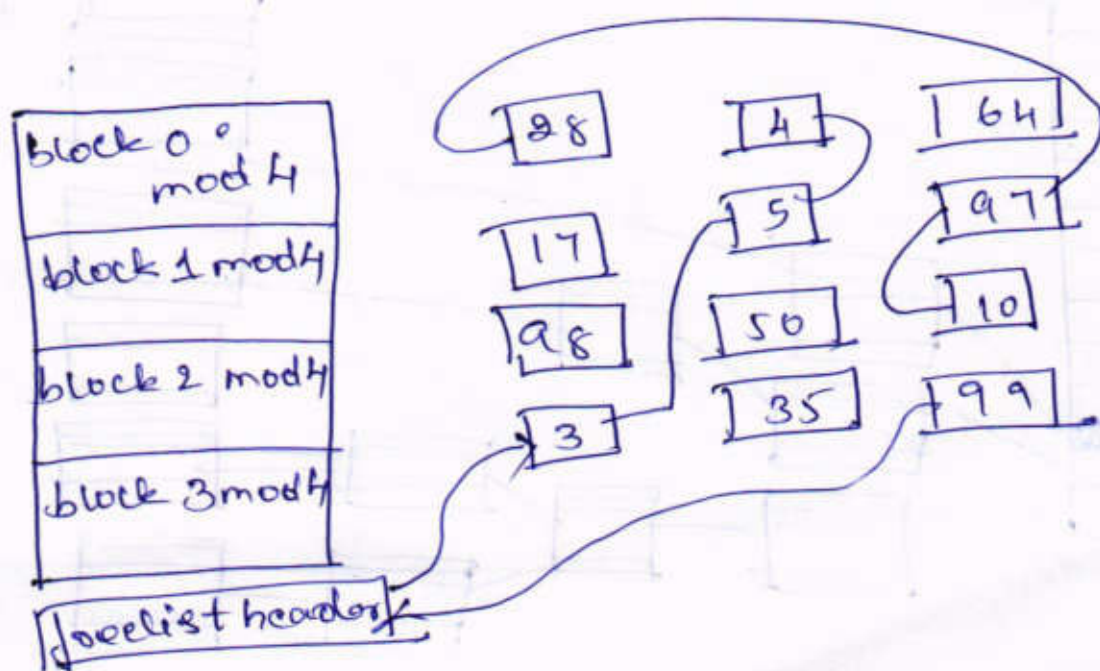
Inodes resides on the disk and the kernel reads them into an in-memory which we can call as in-core inodes. Disk inodes contains the following information:

- \* File access permission and time.
- \* File ownership information.
- \* Type of the file.
- \* Number of links to the file.
- \* File size and organization on disk.



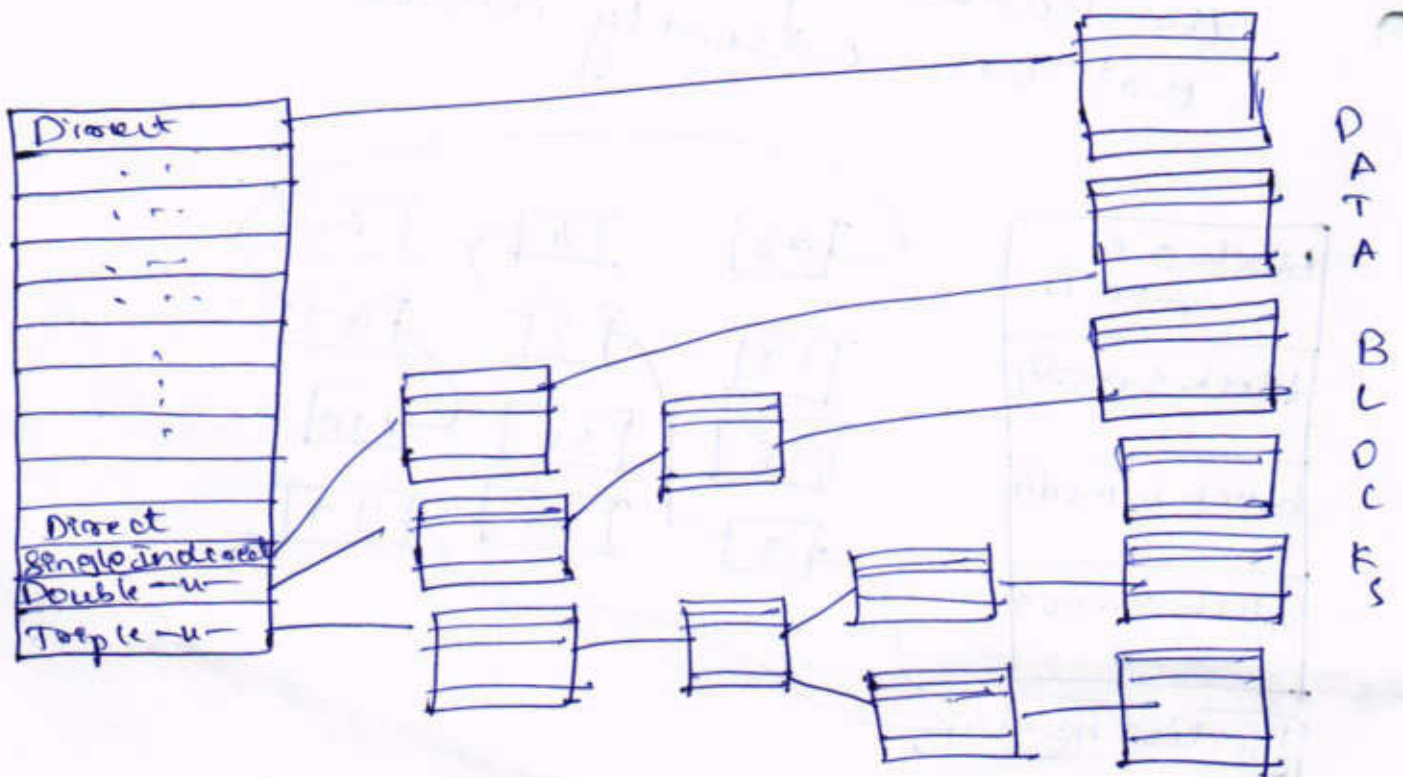
The in-core copy of nodes contains all of the above information, but it also contains the following additional information:

- \* Status (locked / process is waiting for it to become unlocked).
- \* Logical device number of the file system.
- \* Inode Number: Since inodes are stored in a sequential manner on the disk, the kernel uses an identifier of that array to refer to its in-core copy.
- \* pointer to other in-core inodes.
- \* Reference count which indicates the number of instances of the file that are currently active.



# STRUCTURE OF INODE.

- \* structure of a regular file on disk.  
 As stated previously inodes contains the table of content of the file data on disk. As each disk block can be referenced by a number, the table of content is nothing but a sequence of disk block numbers. The file data may not always be stored in contiguous memory locations, hence we will need to keep track of all the block numbers on the disk. The system Unix system have the following 13 entries as the table of contents.





The blocks marked "direct" can refer to a single disk block that contains the real data. "single indirect" block contains the number of a disk block which in itself contains a list of block numbers that we can reference and they have the real data. Going on the same line, we have "double indirect" and "triple indirect" blocks.

Let's now try to get an estimate of the max limit on the size of a file that Unix file system can handle.

Assume one block is of 4 kbytes and a block number is an integer of 4 bytes (32 bits). Thus a block can have 256 block numbers.

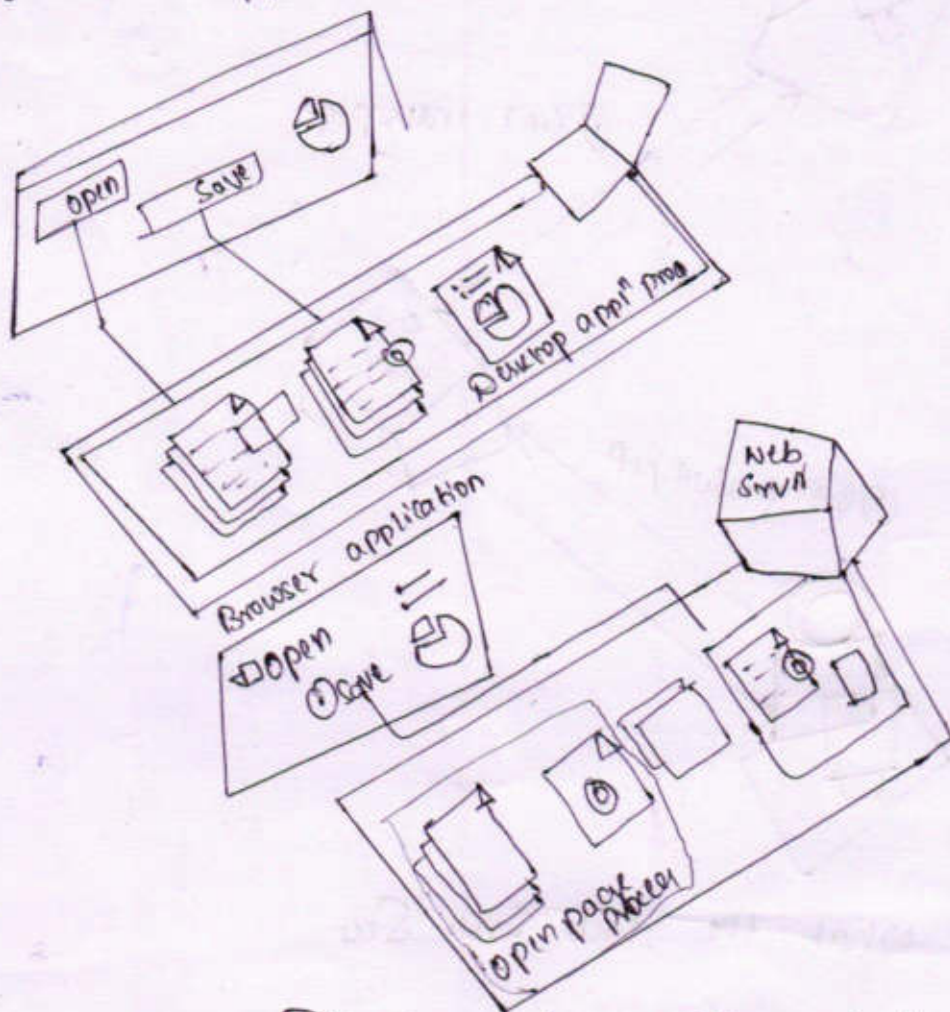
## MODULE-5

### Managing State

#### The problem of state in web applications

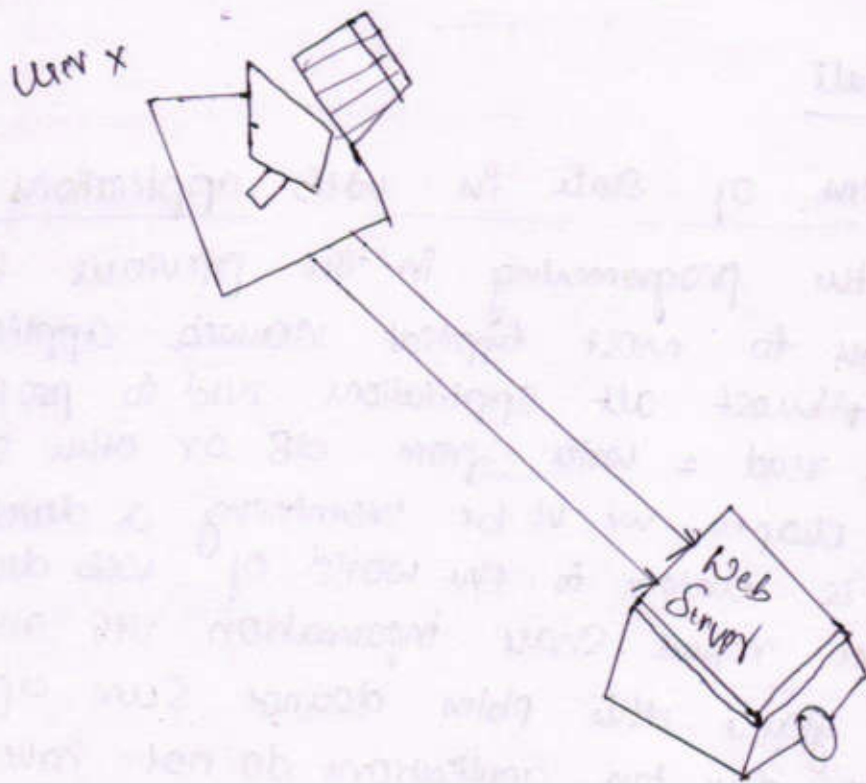
Much of the programming in the previous several chapters has analogies to most typical nonweb application programming. Almost all applications need to process user i/p's. Old info, & read & write from db or other storage media. But in this chapter we will be examining a development problem that is unique to the world of web development: how can one request share information with another request.

At first glance this problem doesn't seem especially formidable. Single user desktop applications do not have this challenge at all as the program information for the user is stored in memory & can thus be easily accessed throughout the application.



Desktop application versus web application





Is for the server not really any different than...

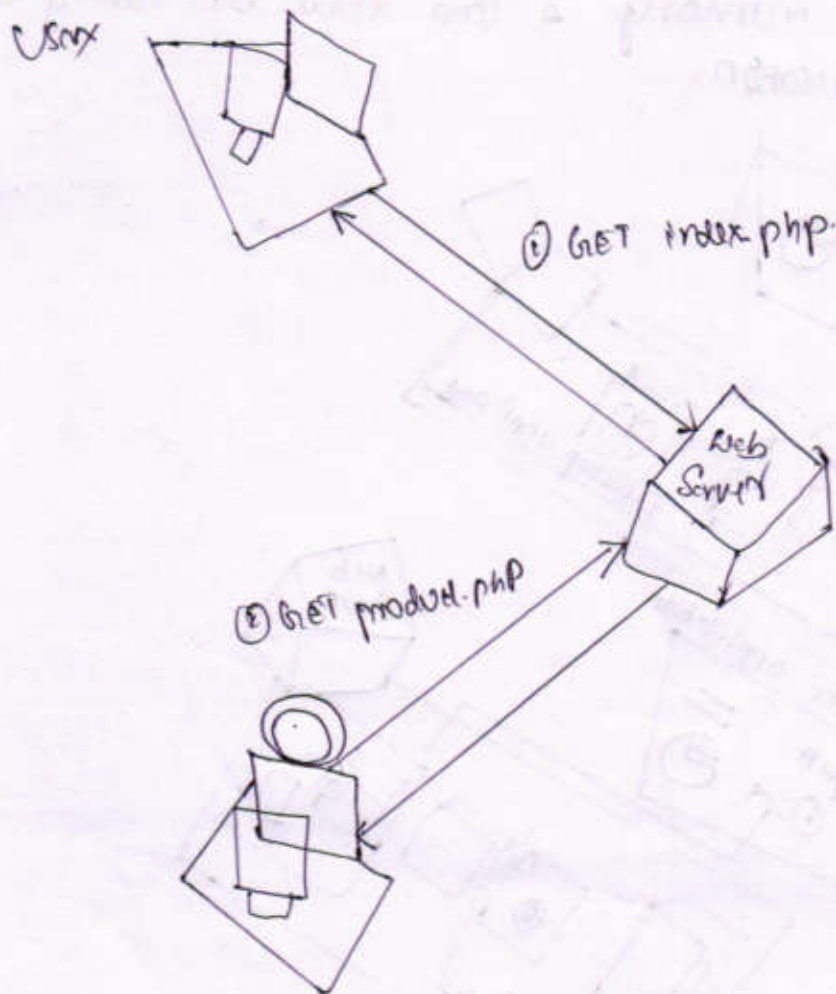


fig: what the web server sees.

The starting point will be to examine the somewhat simpler problem of how does one web page pass info to another page? That is, what mechanisms are available within HTTP to pass info to the server in our request? As we have already seen. In HTTP we can pass information

String using:

- Query string
- Cookie

### 13.2 Passing Information Via Query Strings

As you will recall from earlier chapters, a web page can pass query string information from the browser to the server using one of the two methods: a query string within the URL (GET) & a query string within the HTTP header (POST).



Browser

← →

Artist:

Year:

Nationality:

<form method="GET" action="proces.php">

GET proces.php?artist=picasso&year=1906&nation=spain http/1.1

Query string

<form method="POST" action="proces.php">

```
POST proces.php http/1.1
Date: Sun, 20 May 2012 23:59:59 GMT
Host: www.mysite.com
User-Agent: Mozilla/4.0
Content-Type: application/x-www-form-urlencoded
artist=picasso&year=1906&nation=spain
```

- HTTP header

Query string

fig: Recap of GET Versus POST

## Passing information via the URL path:-

While query strings are a vital way to pass information from one page to another, they do have a drawback.

The URLs that result can be long & complicated.

URLs do provide some benefits with search engine.

result rankings

Dynamic URLs essential part of the web development.

This process is commonly called URL rewriting. The four Commerce-related results for the search term "reproduction Raphael portrait La donna velata" are shown along with their URLs.

Let us begin with the following its query string information

`www.somedomain.com/DisplayArtist.php?artist=16`

One typical alternate approach would be to rewrite the URL to:  
`www.somedomain.com/artists/16.php`

Notice that the query string name & value have been turned into path names. One could improve this to make it more SEO friendly.

using the following

`www.somedomain.com/artist/leonardo-cassatt`

## URL Rewriting in Apache and Linux:-

Depending on your web development platform, there are diff. ways to implement URL rewriting. On web servers.

Running Apache, the solution typically involve using the

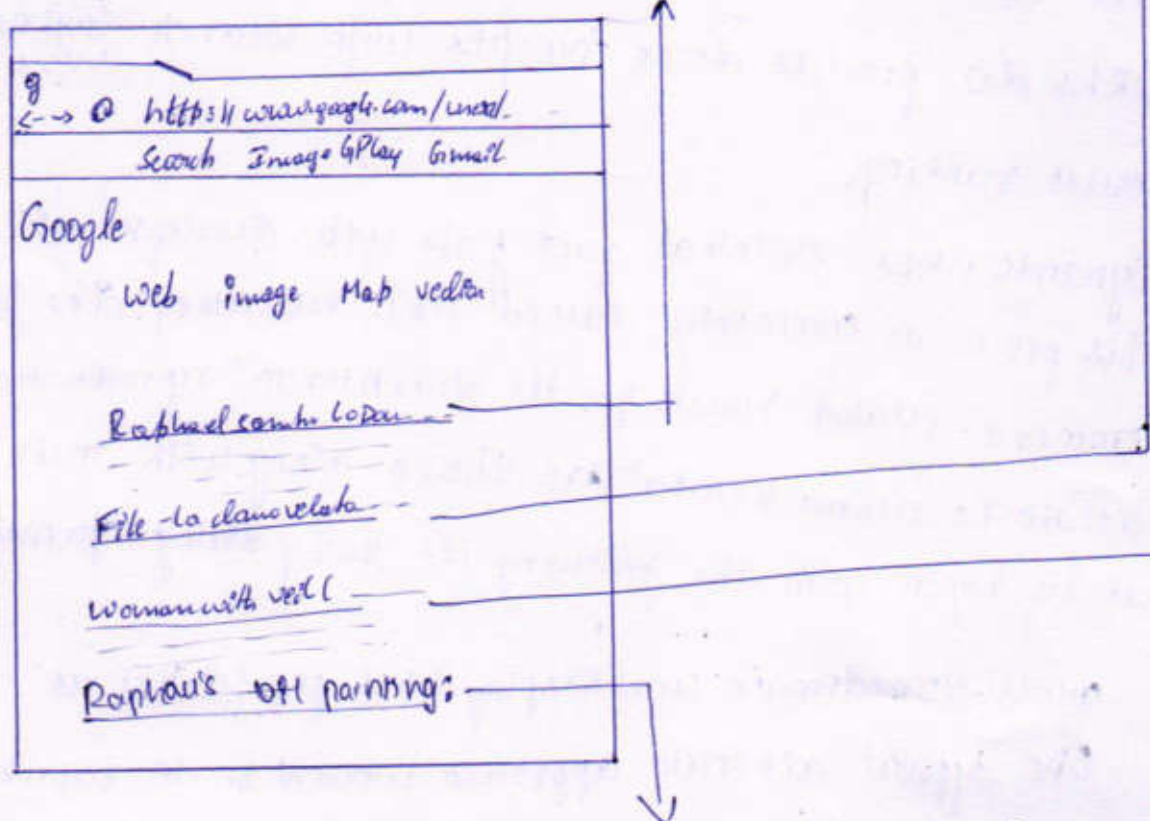
mod\_rewrite module in Apache along with the .htaccess file.



<http://www.1st-art-gallery.com/Raphael/La-Donna-Velata-1416.html> ←

<http://www.paintingall.com/raphael-Sanzio-Woman-with-a-Velata-donna-velata.html> ↑

<http://www.artshaven.com/raphael-la-donna-velata.html>



<http://www.paintingsofholwaler.com/detail.asp?Vcode=6mvd7emriagqilad&fith=La+Donna+Velata>

Cookies: figure: URIs within a Search engine result page

There are few things in the world of web development so reviled & misunderstood as HTTP cookie. Cookies are a

Client-side approach for persisting state information.

They are name=value pairs that are saved within one

or more text files that are managed by the browser.

These pairs accompany both server requests &

responses within the HTTP header.

Cookies were intended to be long-term state mechanism. They provide web site authors with a mechanism for persisting user-related information that can be stored on that domain's cookies are not transported to a diff domain.

While cookies can be used for any state-related purpose they are principally used as a way of maintaining continuity over time in a web application.

How do cookies work?

While cookie information is stored & retrieved by the browser, the information is a cookie travels within the HTTP header.

There are limitations to the amount of information that can be stored in a cookie & to the number of cookies for a domain.

Browsers will delete cookies that are beyond their expiry date specified. The browser will delete it when the browser closes for this reason.

Some commentators will say that there are 2 types of cookie session cookies & persistent cookies.

A session cookies has no expiry date & thus will be deleted at the end of the user browsing.

Session persistent cookies have a expiry date specified.



they will persist the browser's cookie file until the expiry date occurs, after which they are deleted.

Using Cookies:-

PHP provides mechanisms for writing & reading cookies.

Cookies in PHP are created using `setcookie()` function.

& are retrieved using the `$_COOKIE` super global associative array, which works

```
<?php.
```

```
// add 2 day to the current time for expiry time.
```

```
$expirytime = time() + 60 * 60 * 24;
```

```
// create a persistent cookie.
```

```
$name = "username";
```

```
$value = "Ricardo";
```

```
setcookie($name, $value, $expirytime);
```

```
?>
```

~~PHP~~:- writing a cookie.

The `setcookie` function also supports several more parameters, which further customize the new cookie.

```
&?php
```

```
if (isset($_COOKIE["username"])) {
```

```
// no valid cookie found.
```

```
} else {
```

```
 echo "The username retrieved from the cookie is:";
```

```
 echo $_COOKIE['username'];
```

```
}
```

```
>
```

Many sites provide a "Remember Me" check box on login forms, which refers to the use of a persistent cookie.

This login cookie would contain the user's username but not the password.

Every time the user logs in, a new token would be generated & stored in the database & cookie.

For instance, some sites allow the user to choose their preferred color scheme or their country origin, or site language.

Another common use of cookies is to track a user's browsing behavior on a site.

### Serialization:-

It is a process of taking a complicated object & reducing it down to zero & ones for either

storage or transmission.



# interface serializable

1# methods /

public function serializy(1);

public function unserializy(\$Serialized);

;

↳ : the serializable interface.

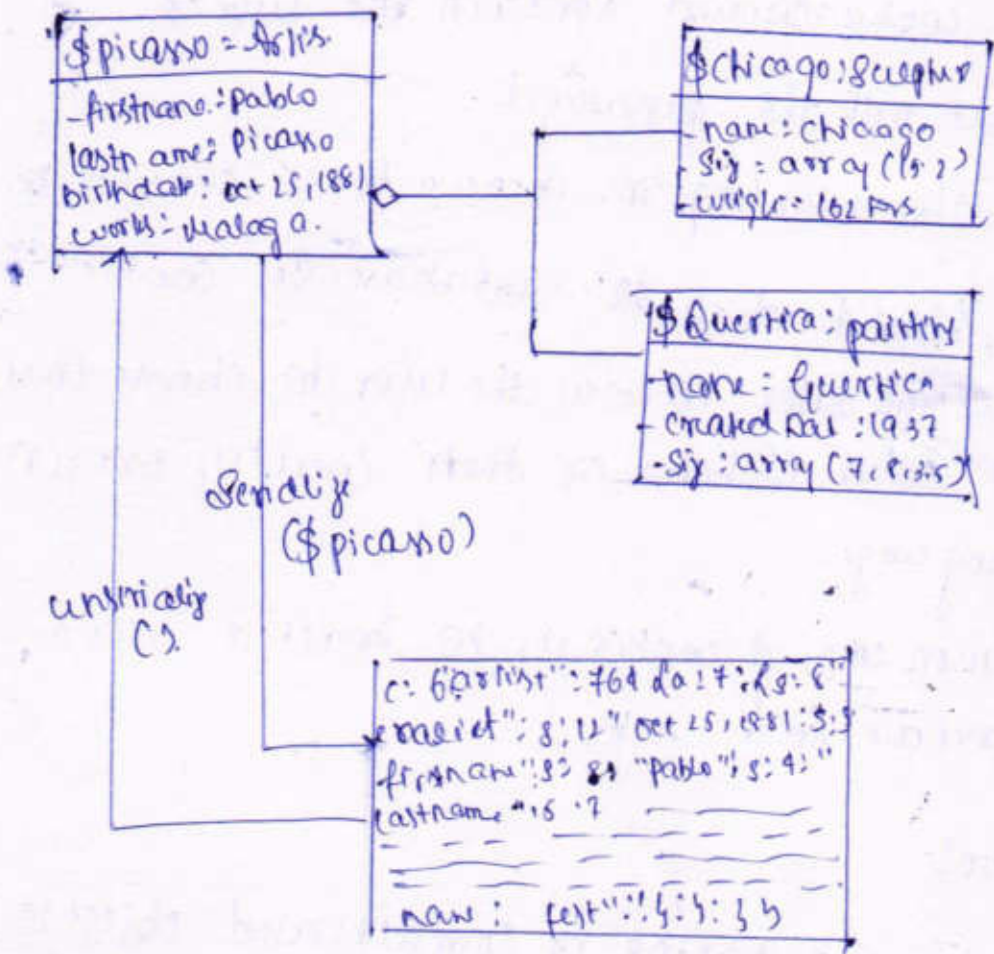


fig: - serialization & deserialization.

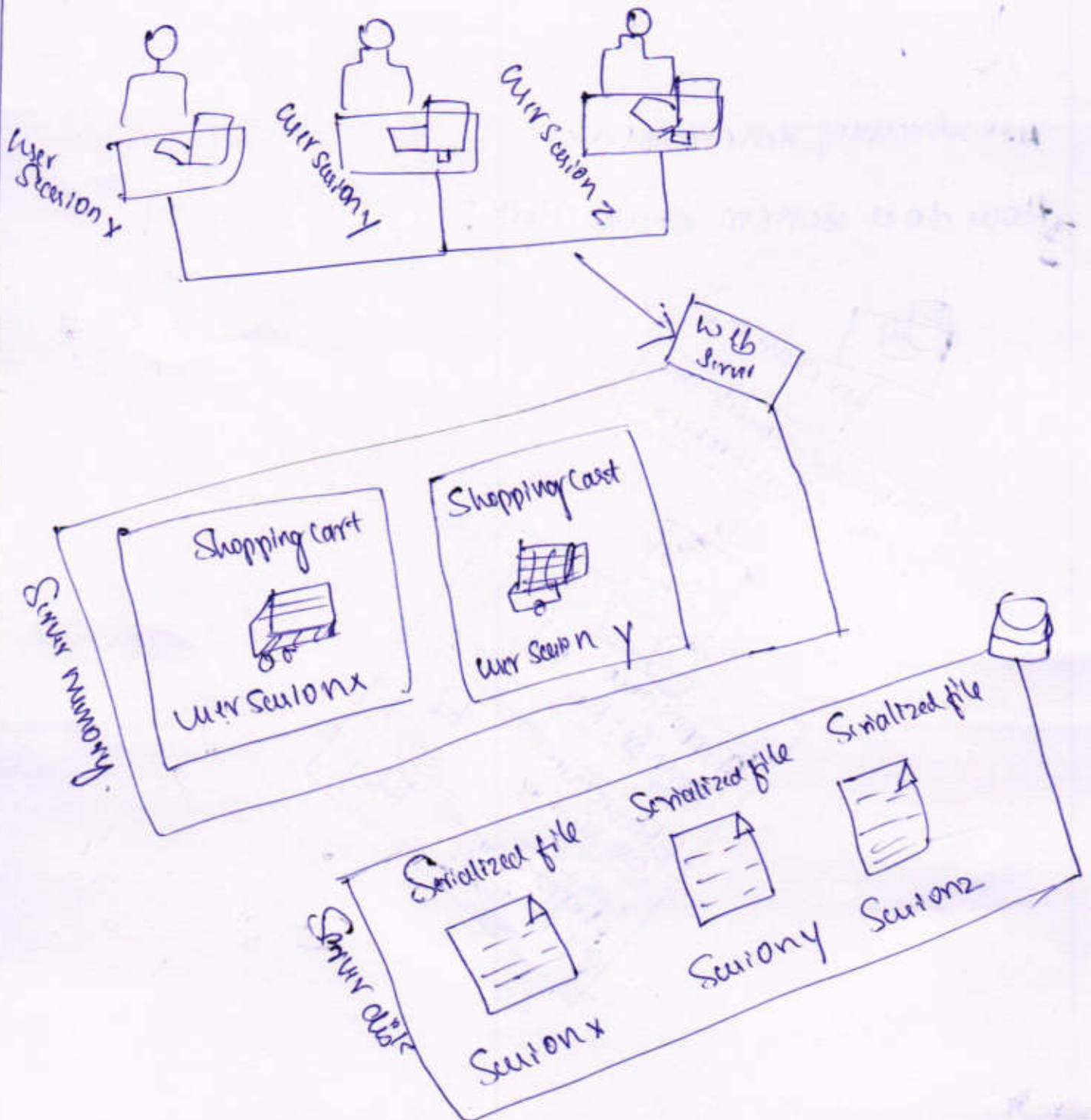
It shows how the artist class must be modified to implement the serializable interface by adding the implements keyword to the class definition & adding

implementation of 2 methods.

implementation of 2 methods.

# Session state.

All modern web development environments provide some type of Session State mechanism. Session State is a Server-based state mechanism that lets web applications store & retrieve objects of any type for each unique user session. That is, each browser session has its own session state stored as a Serialized file on the Server.





To use Session in a Script `session_start()` function

```
<?php
```

```
session_start();
```

```
if (isset($_SESSION['usr'])) {
```

```
 //usr is logged in
```

```
 }
```

```
else {
```

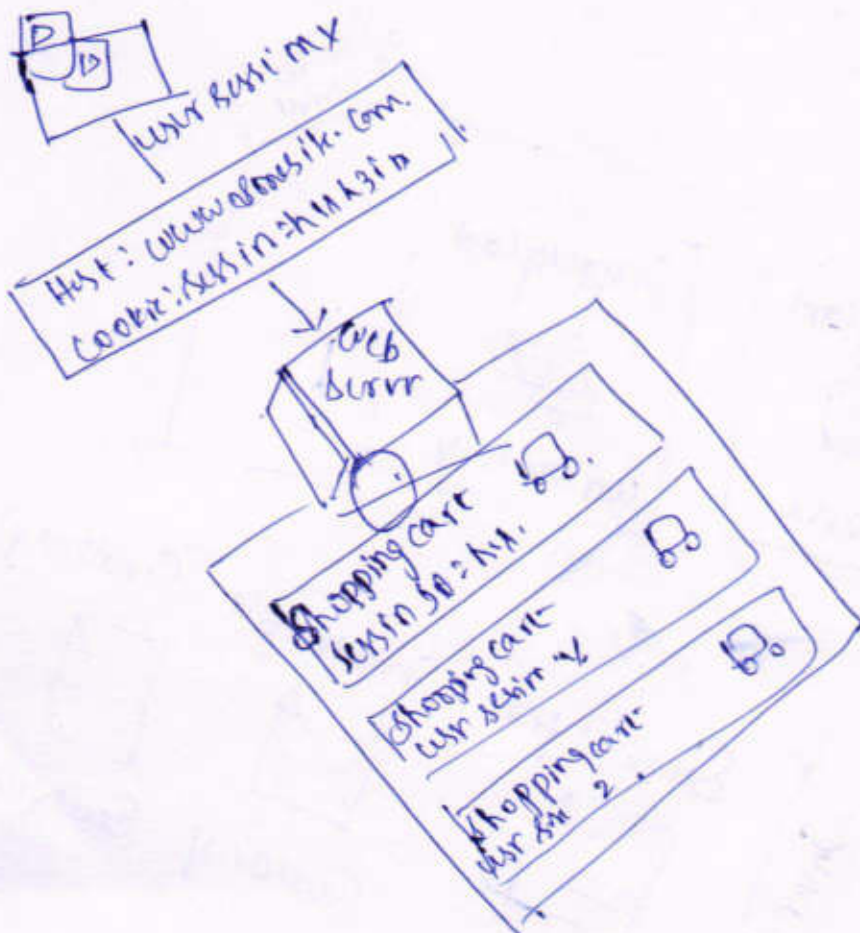
```
 //no one is logged in (guest)
```

```
 }
```

```
};
```

Hint: Accessing session state.

How does session state work?



## HTML5 Web Storage

web storage is a new JavaScript-only API introduced in HTML5.

### Using Web Storage.

As demonstrated that the process of reading from web storage is equally straightforward. The difference between SessionStorage and localStorage in this example is that if we close the browser after writing and then run the code in, only the localStorage item will still contain a value.

```
<form id="formal" runat="server">
 <h1> Web Storage Reader </h1>
 <script language="javascript" type="text/javascript">
 if (typeof (localStorage) === "undefined" ||
 typeof (sessionStorage) === "undefined") {
 alert ("web storage is not supported on this browser..");
 }
 else {
 document.write ("date saved=" + today);
 document.write ("
 favorite artist=" + artist);
 document.write ("
 user name=" + user);
 }
 </script>
</form>
```

### Why Would We Use Web Server Storage?

A better way to think about web storage is not as a cookie replacement but as a local cache for relatively static items available to JavaScript.

### Caching:-

Caching is a vital way to improve the performance of web applications.



## Page Output Caching

There are two models for page caching: full page caching & partial page caching. In full page caching, the entire contents of a page are cached.

Page caching is not included in PHP by default, which has allowed a market place for free and commercial third-party cache add-ons such as Alternative PHP Cache and Zend to flourish.

## Application Data Caching

An alternate strategy is to use application data caching in which a page will programmatically place commonly used collections of data that require time-intensive queries from the database or web server into cache memory.

```
$memcache = new Memcache;
$memcache->connect('localhost', 11211)
or die("could not connect to memcache server");
$cachekey = 'topcountries';

if (!isset($countries = $memcache->get($cachekey)))
$cgate = new CountryTableGateway($dbAdapter);
$countries = $cgate->getMostPopular();
or die("failed to save cache data at the server");
}
displayCountryList($countries);
?>
```

## \* jQuery Attributes: HTML Attributes:

The core set of attributes related to DOM elements are the ones specified in the HTML tags described.

Attributes like the href attribute of an <a> tag, the src attribute of an <img>, or the class attribute of most elements.

```
Ex: var link = $("a").attr("href")
 $("a").attr("href", "http://funweblog.com");
 $("img").attr("class", "fancy");
```

\* jQuery Listeners: Java jQuery supports creation and management of listeners/handlers for JavaScript events. The usage of these events is conceptually the same as which JavaScript with some minor syntactic difference.

## \* Modifying the DOM:

Creating DOM and textNodes: If you decide to think about your page as a DOM object, then you will want to manipulate the tree structure rather than merely manipulate strings. Thankfully, jQuery is able to convert strings containing valid DOM syntax into DOM objects automatically.

```
var element = document.createElement('div');
```

\* AJAX: Asynchronous JavaScript with XML (AJAX) is a term used to describe a paradigm that allows a web browser to send message back to the server without interrupting the flow of what's being shown in the browser.



## \* jQuery Foundation :-

A library or Framework is software that you can utilize in your own software which provides some common implementation of standard ideas. A web framework can be expected to have features related to the web including HTTP headers, AJAX, authentication, DOM manipulation, cross-browser implementation, and more.

## \* jQuery Selectors :-

Selectors offer the developer a way of accessing and modifying a DOM object from an HTML page in a simple way. Although the advanced `querySelector()` method allows selection of DOM elements based on CSS selectors, it is only implemented in newer browsers. To address this issue, jQuery introduced its own way to select an element, which under the hood supports a myriad of older browsers.

### Basic Selectors :-

- \* `$("*")` Universal selector matches all elements
- \* `$("tag")` Element selector matches all elements with the given element name.
- \* `$(".class")` Class selector matches all elements with the given CSS class
- \* `$("#id")` Id selector matches all elements with a given HTML id attribute.

EX:- `var singleElement = $("#grab");`



\* Making Asynchronous Requests: JQuery provides a family of methods to make asynchronous requests. We will start with the simplest GET request, and work our way up to the more complex usage of AJAX where all variety of control can be exerted.

EX: `$('#timeDiv').load('currentTime.php')`

\* Complex Control over AJAX:

It borrows from both the `$.get()` and `$.post()` methods and actually shorthand form for the `$.ajax()` method which allows fine-grained control over HTTP requests. This method allows us to control many more aspects of our asynchronous JavaScript requests including the modification of headers and use of cache controls.

```
$.ajax({url: "vote.php",
 data: $('#voteForm').serialize(),
 async: true,
 type: "post"
});
```

\* Asynchronous File Transmission:

Asynchronous file transmission is one of the most powerful tools for modern web applications.

EX: `<form name="fileUpload" id="fileUpload" enctype="multipart/formdata" method="post" action="upload.php">`

`input name="image" id="images" type="file" multiple />`  
`</form>`



\* Animation: When developers first learn to use jQuery, they are often initially attracted to the easy-to-use animation features. When used appropriately, these animation features can make your web application appear more professional and engaging.

\* Animation Shortcuts:

Animation is no different with a raw `animate()` method and many more easy-to-use shortcuts like `fadeIn()`/`fadeOut()`, `slideUp()`/`slideDown()`.

The `hide()` and `show()` methods can be called with no arguments to perform a default animation. Another version allows two parameters: the duration of the animation and a callback method to execute on completion. Using the fully visible before changing their contents.

\* Raw Animation: The `animate()` method has several variations but the one we will look at has the following form.

• `animate(properties, options);`

The `properties` parameter contains a Plain Object with all the CSS styles of the final state of the animation. The `options` parameter contains another Plain Object with any of the options below set.

\* `always` in the function to be called when the animation completes or stops with a fail condition. This function will always be called.

\* `Done` in a function to be called when the duration of the animation

\* `fail` in the function called if the animation does not complete

\* `Progress` in a function to be called after each step of the animation.



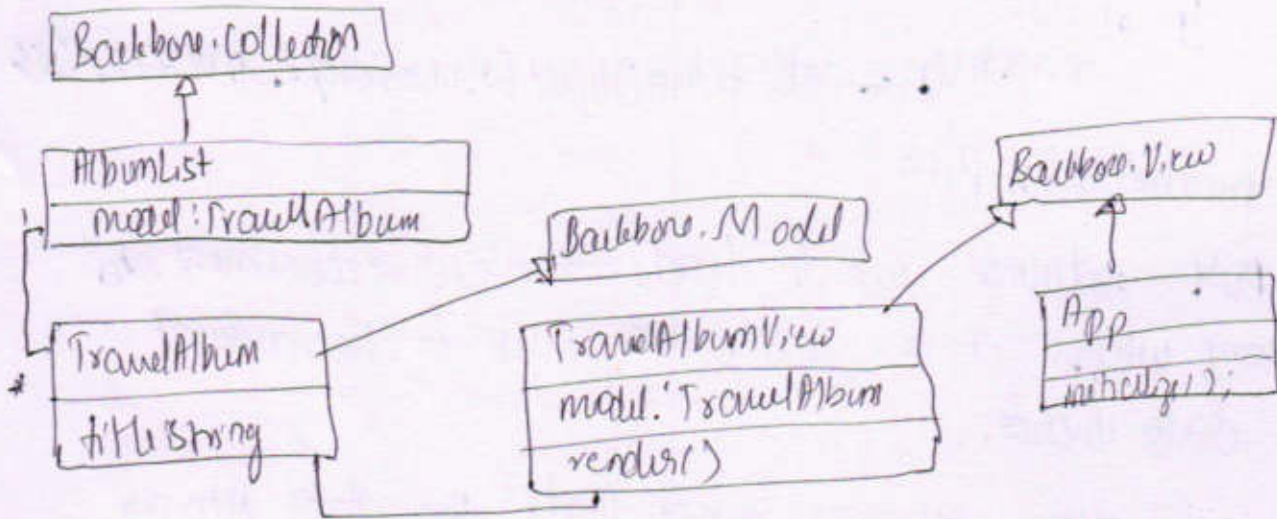
# Backbone MVC framework

## \* Getting started with backbone.js

Backbone is an MVC framework that further abstracts javascript with libraries

```
<script src="backbone-min.js"></script>
```

## \* Backbone Models



## \* Collection

```
var AlbumList = Backbone.Collection.extend({
 model: TravelAlbum
 getChecked: function() {
 return this.where({checked: true});
 }
});
```

Backbone introduces the concept of collection which are normally used to contain lists of Model Objects.

## \* View

Deriving custom view objects of models and collection.

```
var TravelAlbumView = Backbone.View.extend({
 tagName: 'li',
 events: {
 'click': 'toggleAlbum',
 initialize: function() {
 this.listenTo(this.model, 'change', this.render);
 },
 toggleAlbum: function() {
 this.model.toggle();
 }
 }
});
```



## XML processing

### \* XML processing in JavaScript

The code necessary for loading an XML document into an XML object, and it displays the id attribute of <parenting> elements as well.

<script>

```
if (window.XMLHttpRequest)
```

```
xmlhttp = new XMLHttpRequest();
```

```
else
```

```
xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

### \* XML processing in PHP

The DOM extension which loads the entire document into memory where it is transformed into a hierarchical tree data structure.

The simple XML extension which loads the data into an object that allows developer to access via array.

<body>

```
<div id="container">
```

<script>

```
act = '<?xml version="1.0" encoding="ISO-8859-1"?'>';
```

```
xmlDoc = $->parseXML(act);
```

```
$xml = $(xmlDoc);
```

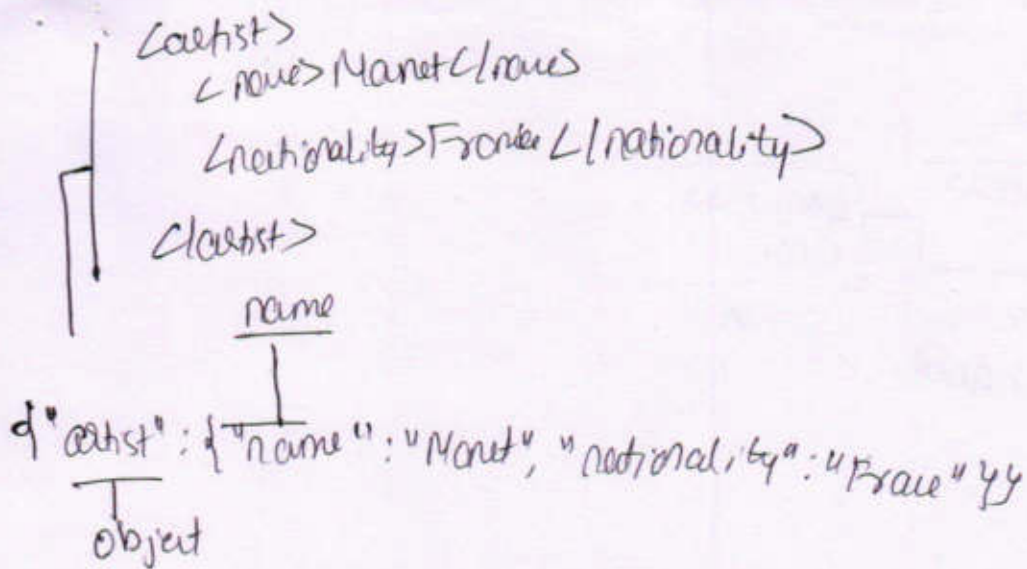
```
$parenting = $xml->find('parenting');
```

```
$parenting->each(function()
```

```
{ ("#container").append($ (this).attr('id') + "-");
```

```
}); </script>
```

## JSON



JSON representation of XML data is a data serialization format.

\* Using JSON in JavaScript

The syntax of JSON is the same used for creating object in JavaScript, it is make use of JSON format

\* Using JSON in PHP

Ex. php

```
$text = '{"artist": {"name": "Monet", "nationality": "France"}}';
```

```
echo $anObject->artist->nationality;
```

```
$anArray = json_decode($text, true);
```

```
echo $anArray['artist']['nationality'];
```

To go to other direction you can use `json_encode()`

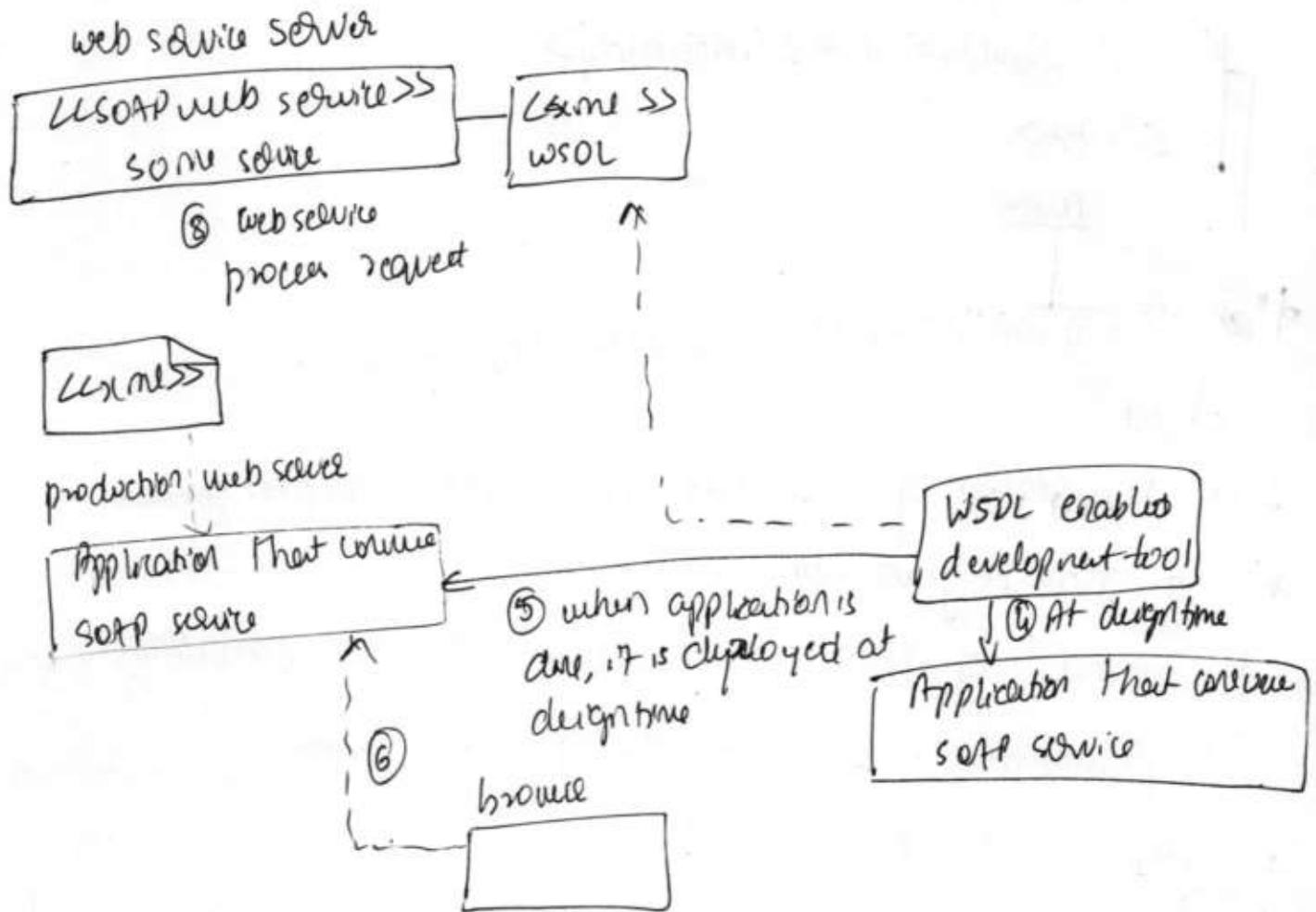
```
$text = json_encode($anObject);
```

JSON data is often coming from an external source, one should always check for parse errors before using it.



# Overview of web services

## \* SOAP service



## \* Identifying and authenticating service requests:-

Identify:- Each web service request must identify who is making the request

authentication:- Each web service request must provide additional evidence that they are who they say they are.

`http://dev.virtual earth.net/REST/v1/locations?O=xml?query=British+20+museum+Great+Russell+Street+London+W1B+3DQ,+UK` & key = [PUTTING API KEY HERE]

Some web services are simply providing information already available on their websites.