Windows OS

Purpose	Command
To view Setting	utilman
To open Programs and Features Wizard	appwiz.cpl
To open windows features wizard	**optionalfeatures**
Device Manager	devmgmt.msc
Firewall Control Panel	firewall.cpl
Firewall Control Panel with Advance Security	wf.msc
To export Advance Firewall with security wizard	netsh advfirewall export 'C:\ advfirewallpolicy.wfw'
To see the “AnyDesk” firewall rule	Get-NetFirewallRule -DisplayName *AnyDesk*
To see control panel Item	Get-ControlPanelItem
To open a specific Item of control panel	Show-ControlPanelItem -Name "AutoPlay"
CPU information	Wmic cpu get name
System Information	`msinfo32`
System Properties	`sysdm.cpl`
Local Security Policy	`secpol.msc`
Local Users and Groups	`lusrmgr.msc`
Log out	`logoff`
Shut Down Windows	`shutdown`
To forcefully shutdown	`Shutdown -p`
To restart machine	`Shutdown -r`
Windows Version (About Windows)	`winver`

To open network connection window	ncpa.cpl
IP Configuration	ipconfig
IP Configuration with extra info eg. DNS,DHCP	ipconfig /all
To release IP info	Ipconfig /release
To renew IP info	Ipconfig /renew
Shared Folder Wizard	shrpubw
Shared Folders	fsmgmt.msc
Windows Firewall	WF.msc
Task Manager	taskmgr
To see the use accounts	Net user
To create an user	Net user /add mushfiq
To set password of an user(mushfiq)	Net user mushfiq *
To create an user(itbd) with password	Net user /add itbd *
To Kill any Task by Process ID	Taskkill /PID 3248
To Kill any process by name	Taskkill /F /IM cmd.exe
To format the D drive by renaming “Test”	FORMAT D: /FS=NTFS /V:Test
To Copy any file from one drive to another drive Here, Test.txt is source file	copy I:\Test.txt H:\peresi.txt
Shutdown the system	Shutdown -s
Restart the system	Shutdown -r
Shutdown the system forcefully	Shutdown -p
Restart the system forcefully	Shutdown -r -t 0

To add static Route into system	`Route add 10.0.0.0 mask 255.0.0.0 10.1.1.1`
To add Permanent static Route into system	`Route add -p 10.0.0.0 mask 255.255.255.0  10.1.1.1`
To show route	`Route print`
To shrink a partition in Windows PowerShell	`Resize-Partition –DiskNumber 0 -PartitionNumber 2 –Size 50GB`
To create a VHD in Windows PowerShell	`New-VHD –Path D:\VHD\ MyDynamicDisk.vhdx -SizeBytes 100GB –Dynamic`
To mount virtual hard disks	`Mount-VHD -Path c:\test\ testvhdx.vhdx`

Windows Server

Distributed File Service Mgmt	**DFSmgmt.msc**
Disk Manager	**DiskMgmt.msc**
DNS Manager	**DNSmgmt.msc**
AD Domains and Trusts	**Domain.msc**

sAD Users and Computers	DSA.msc
AD Sites and Services	DSsite.msc
SID of an User	Wmic useraccount get name,sid
To install DNS Role	Add-WindowsFeature DNS -IncludeManagementTools
Enable Remote Management Connecting a Remote System	Enable-PSRemoting –Force Enter-PSSession –ComputerName *host*
To see all Virtual Machine	Get-VM
To Stop Running-virtual Machine	Stop-VM -Name "Name-of-VM"
To Enable virtual Machine	Start-VM -Name "Name-of-VM"
Enable Hyper-V Manager into Virtual Machine	Set-VMProcessor -VMName "NAME-OF-VM" -ExposeVirtualizationExtensions $true
To See The History Of Executed Command	History
To Save the Executed Command File	History > c:\<NAME-OF-THE-FILE>.txt
To Open the Saved File	Notepad <NAME-OF-THE-FILE>.txt
To Rename server Name	Netdom renamecomputer <Current-computer-

	name> /newname: new-computer-name
Domain join to a Member Server	Netdom join <computer-name> /domain: <domain-name-want-to-Join> /userd: <username> /passwordd:<*******>
To Check Integration Service of a Machine	Get-VMIntegrationService -VMName "Name-OF-VM"
To Get Network Adapter Information	Get-VMNetworkAdapter -VMName "Name-OF-VM"
To install just the Hyper-V Manager	install-windowsfeature -name hyper-v-tools
To install just the Hyper-V PowerShell module	install-windowsfeature -name hyper-v-powershell
To install the management tools	install-windowsfeature -name rsat-hyper-v-tools

Creating and Managing Organizational Unit(OU) with Powershell:

==========

1. **Create an OU:**

New-ADOrganizationalUnit "OU-1"

2. **Create a Child OU into a parent OU:**

New-ADOrganizationalUnit "CHILDOU-2" -Path "OU=OU-1,DC=itbd,DC=local"

3. **Apply a feature onto an OU:**

Set-ADOrganizationalUnit -Identity "OU=CHILDOU-1,OU=OU-1,DC=itbd,DC=local" -ProtectedFromAccidentalDeletion:$false

4. **Remove an OU:**

Remove-ADOrganizationalUnit -Identity "OU=CHILDOU-1,OU=OU-1,DC=itbd,DC=local" -Confirm:$FALSE

IP Addressing

1. Set IP Address:

New-NetIPAddress -InterfaceIndex 4 -IPAddress 192.168.1.2 -AddressFamily IPv4 -PrefixLength 24

2. To see the Address:

Get-NewIPAddress

3. To remove that IP Address:

Remove-NetIPAdrees -InterfaceIndex 4 -IPAddress 192.168.1.2 -AddressFamily IPv4 -PrefixLength 24

Renaming Computer:

==========

Rename-computer -NewName coredc -Restart

Basic Commands:

==========

Get-Command * To see all commands
Get-Help Get-localuser * To get help for any command
Update-help * For updating help command
Get-Command *user* * To see user related commands
Get-Command *Partition* * To see partition related commands

[About local users]

Get-LocalUser
Get-LocalUser -Name administrator
*Here Get-LocalUser=Command , Name=Parameter , administrator=Object
New-LocalUser -Name admin1
*Creating a new local user
Remove-LocalUser -Name admin1

To see the Roles & Feature :
==========

Get-WindowsFeature
Get-WindowsFeature *domain*

Nested virtual Machines:

==========

Set-VMProcessor -VMName Server1 -ExposeVirtualizationExtensions $true

To see,start,stop,enter VMs:

==========

get-vm
start-VM -VMName DC * DC= VM Name
stop-VM -VMName DC

Enter-PSSession -VMName DC [To enter Virtual Machine's Powershell]

To install ADDS:

==========

-Install-WindowsFeature -Name ad-domain-services -IncludeAllSubFeature -IncludeManagementTools

-Install-ADDSForest -DomainName itbd.local -DomainMode Default -FoestMode Default -DomainNetbiosName ITBD -InstallDns

have to give DSRM Password

After restarting , you have to change the DNS IP.

To see the output:

==========

Get-ADDomain

Get-ADForest

For Managing Remote Server(Standalone Server):

set-item WSMan:\localhost\client\TrustedHosts -value *

{We can use specific IP instead of * , then specific that IP will be managed only}

To create Organizational Unit :

New-ADOrganizationalUnit -Name ITDepartment
New-ADOrganizationalUnit -Name HRDepartment
New-ADOrganizationalUnit -Name MGTDepartment
New-ADOrganizationalUnit -Name AdminDepartment

To see OU:

Get-ADOrganizationUnit
Get-ADOrganizationUnit -Filter * | fl name

To create User:

New-ADUser -Name mushfiq -SamAccountName Mushfiq -UserPrincipalName mushfiq@itbd.local -Path "OU=ITDepartment,DC=itbd,DC=local" -AccountPassword (Read-Host -AsSecureString typethepassword) -Enabled $true

or

New-ADUser -Name mushfiq -SamAccountName Mushfiq -UserPrincipalName mushfiq@itbd.local -Path "OU=ITDepartment,DC=itbd,DC=local" -AccountPassword (ConvertTo-SecureString "abcd1234" -AsPlaintText -Force) -Enabled $true

To see all users:

Get-ADUser -Filter *
Get-ADUser -Filter * | fl name

To create a CSV file :

1. Open a text file in c drive and write:
name,samaccountname,userprincipalname,path
it1,it1,it1@itbd.local,"ou=itdepartment,dc=itbd,dc=local"
it2,it2,it2@itbd.local,"ou=itdepartment,dc=itbd,dc=local"
it3,it3,it3@itbd.local,"ou=itdepartment,dc=itbd,dc=local"

2. Save it as a users.csv extension.
3.type in powershell following command
Import-Csv -Path c:\users.csv | New-ADUser -AccountPasword (ConvertTo-SecureString "abcd1234" -AsPlainText -Force) -Enabled $true

Trust Relationship :

Test-ComputerSecureChannel -Server ServerGui -Repair -Credential administrator@itbd.local

Install Active Directory

==========

1. **Install-WindowsFeature -name ad-domain-services -includemanagementtools**
2. **Install-ADDSForest -DomainName itbd.local -DomainMode Default -ForestMode Default -DomainNetbiosName ITBD -InstallDNS**

Uninstall ADDS:

==========

1. **Uninstall-ADDSDomainController -LastDomainControllerInDomain -RemoveApplicationPartitions**
2. **Uninstall-WindowsFetaure -Name DNS**
3. **Uninstall-WIndowsFeature -Name ad-domain-services**

Configuring IPv4

Purpose	Command
To get Adapter details	*Get-NetAdapter*

To set IP	*Netsh interface ipv4 set address name="Ethernet" source=static addr=192.168.17.1 mask=255.255.240.0 gateway=192.168.31.254* [Ethernet= network adapter name] [address, mask, gateway= as per your requirement]
To get IP Address	*Get-NetIPAddress*
To change IP Address	*Set-NetIPAddress –InterfaceAlias Ethernet –IPAddress 192.168.17.1*
To set DNS	*netsh interface ip set dns name="Ethernet" static 192.168.16.1*

Sharing file

To share a file	`net share Docs=E:\Documents /grant:everyone,FULL`
To share with specific user	`net share Docs=E:\Documents /grant:everyone,FULL  /users:10`
List of shared file/folder	`net share`
To delete shared file/folder	`net share docs /delete`
To delete shared on remote PC	`net share sharename \\remotepc /delete`

Disable/Enable the Firewall

Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled false
Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled true

Start / Stop the service

net start RpcSs

net stop RpcSs

To Install Nano-Server

New-NanoServerImage -MediaPath D:\Soft\Microsoft\Windows-Server-2016\DVD -BasePath .\Base -TargetPath .\Nano1\Nano2.vhd -ComputerName Nano1 –Packages 'Microsoft-NanoServer-Compute-Package','Microsoft-NanoServer-OEM-Drivers-Package','Microsoft-NanoServer-Guest-Package', 'Microsoft-NanoServer-Storage-Package' –Language en-us

Hyper-V

install-windowsfeature –name hyper-v -includemanagementtools –restart or,
dism /online /enable-feature /featurename:microsoft-hyper-v

IPAM

Get information about all forward zones

```
Get-IpamDnsZone -ZoneType "Forward"
```

Get information about the forward zones in a specific DNS zone

```
Get-IpamDnsZone -ZoneType "Forward" -ZoneName "abc.itbd.local"
```

Get information about all IPv4 scopes

```
Get-IpamDhcpScope -AddressFamily "IPv4"
```

Get information about the IPv4 scopes on a specific server

```
Get-IpamDhcpScope -AddressFamily "IPv4" -ServerFqdn "abc.itbd.local"
```

Group Policy Object

```
Get-GPO -all -Server TIBDC1
```

Create a GPO in the domain of the user

```
PS C:\> New-GPO -Name TestGPO -Comment "This is a test GPO."
```

Remove a GPO by name

```
Remove-GPO -Name "TestGPO"
```

Rename a GPO

```
Rename-GPO -Name "SampleGPO" -TargetName "SecurityGPO"
```

Backup a GPO to a specific directory

```
Backup-Gpo -Name TestGPO -Path C:\GpoBackups -Comment "Weekly Backup"
```

Restore a GPO from a directory

```
Restore-GPO -Name "TestGPO" -Path "\\Server1\Backups"
```

গ্রুপ পলিসি এবং প্রেফারেন্সের মধ্যে পার্থক্য

বৈশিষ্ট্য	গ্রুপ পলিসি	প্রেফারেন্স
প্রয়োগ	একাধিক কম্পিউটারে	একক কম্পিউটারে
নিয়ন্ত্রণ	কেন্দ্রীয়ভাবে	ব্যক্তিগতভাবে
লক্ষ্য	সংস্থার নিরাপত্তা এবং সুসংগতি নিশ্চিত করা	ব্যবহারকারীর কাজের পরিবেশকে আরামদায়ক করা
পরিবর্তন	সিস্টেম এডমিনিস্ট্রেটর কর্তৃক	ব্যবহারকারী কর্তৃক

গ্রুপ পলিসি উদাহরণ:

- সফটওয়্যার ইনস্টলেশন: কোন সফটওয়্যার সব কম্পিউটারে ইনস্টল থাকবে তা নির্ধারণ করা।
- সুরক্ষা সেটিংস: ফায়ারওয়াল, ভাইরাস স্ক্যানার ইত্যাদি সব কম্পিউটারে একইভাবে কনফিগার করা।
- নেটওয়ার্ক সেটিংস: ইন্টারনেট অ্যাক্সেস, প্রিন্টার শেয়ারিং ইত্যাদি নিয়ন্ত্রণ করা।

প্রেফারেন্স কী?

প্রেফারেন্স হল ব্যক্তিগত সেটিংস যা একজন ব্যবহারকারী নিজের কম্পিউটারে পরিবর্তন করতে পারে। এটি ব্যবহারকারীকে তার কাজের পরিবেশকে তার পছন্দ অনুযায়ী কাস্টমাইজ করার অনুমতি দেয়।

উদাহরণ:

- ডেস্কটপের ব্যাকগ্রাউন্ড: ব্যবহারকারী নিজের পছন্দমতো ছবি ব্যাকগ্রাউন্ড হিসেবে সেট করতে পারে।
- স্ক্রিন রেজোলিউশন: ব্যবহারকারী নিজের চোখের স্বাচ্ছন্দ্য অনুযায়ী স্ক্রিন রেজোলিউশন পরিবর্তন করতে পারে।
- ব্রাউজার বুকমার্ক: ব্যবহারকারী নিজের প্রিয় ওয়েবসাইটগুলো বুকমার্ক করতে পারে।

PowerShell Desired State Configuration

DSC হল একটি শক্তিশালী টুল যা আপনার কম্পিউটার বা সার্ভারকে একই অবস্থায় রাখতে সাহায্য করে। স্বয়ংক্রিয়ভাবে কনফিগার করতে সাহায্য করে। এটি সময় বাঁচাতে, ত্রুটি কমাতে এবং আপনার সিস্টেমগুলিকে আরো সুসংগত করতে সাহায্য করতে পারে।

উদাহরণ:
ধরুন, আপনি চান যে, আপনার সার্ভারে একই সফটওয়্যার এবং ফাইল সবসময় থাকবে। আপনি **DSC** ব্যবহার করে একবার সেটিংস দিয়ে দিন, তারপর **DSC** আপনার সার্ভারকে সবসময় সেই সেটিংস অনুযায়ী রাখবে।

সহজ ভাষায়:
আপনি একবার নির্দেশনা দেবেন: "এটি থাকতে হবে, এটি থাকতে হবে না।" পরে, **DSC** স্বয়ংক্রিয়ভাবে সেই নির্দেশনা অনুযায়ী সব কিছু ঠিক রাখবে, যদি কিছু পরিবর্তন হয়।

কেন এটা দরকার?
- আপনার সার্ভারের কনফিগারেশন সবসময় একই থাকবে, ভুল হবার সম্ভাবনা কম।

সংক্ষেপে: **DSC** মানে স্বয়ংক্রিয়ভাবে সিস্টেম ঠিক রাখা।

এভাবে ভাবুন, আপনি যেমন ফাইলের কপি পেস্ট করতে পারেন, **DSC** তেমনই আপনার সিস্টেমের "কনফিগারেশন কপি" করে রাখে!

Windows NT Based Domain Controller	Windows Active Directory based Domain Controller
একটি centralized authentication system, যা শুধুমাত্র primary domain controller (PDC) এবং backup domain controller (BDC) ব্যবহার করত। Backup Domain Controller (BDC) এবং Additional Domain Controller (ADC) একি জিনিস নয়। - Windows NT 4.0 তে, ডেটা র replication মূ লত PDC থে কে BDC তে করা হতো, কিন্তু এটি Active Directory-এর মতো full replication বা distributed database সি স্টে ম ছি ল না ।	Active Directory অনেক বেশি distributed(**), যেখানে একাধিক domain controllers এর মধ্যে full replication হয়, অর্থাৎ প্রত্যেকটি domain controller একই ডেটাবেসের কপি রাখে এবং সেগুলি একে অপরের সাথে সিঙ্ক্রোনাইজ হয়। এতে global availability আসে এবং ডেটা কোথাও হারিয়ে যাওয়ার কোনো সম্ভাবনা থাকে না।
শুধুমাত্র domains ছি ল, এবং একক domain এর মধ্যে user accounts এবং computers ম্যানেজ করা হত। এখানে কোনো organizational units (OU) বা forest এর ধারণা ছি ল না ।	Active Directory-তে hierarchical structure এ domains, organizational units (OU), trees এবং forests ব্যবহৃত হয়। এটি আরও উন্নত এবং স্কেলযোগ্য, যেখানে আপনি অনেক বড় এবং জটিল নেটওয়ার্ক ম্যানেজ করতে পারেন।
Windows NT 4.0 তে শুধু user accounts এবং groups ব্যবস্থাপনা করা হতো, কিন্তু এটি একটি পূর্ণাঙ্গ Directory Service ছি ল না ।	একটি fully-featured directory service যা user accounts, group policies, security policies, network resources ইত্যাদি সব কিছু centralized ভাবে ম্যানেজ করে , এবং এটি একটি distributed database হিসেবে কাজ করে ।
Windows NT 4.0 Domain Controller তে Global Catalog ছি ল না ।	কিন্তু Active Directory তে Global Catalog থা কে, যা directory searches দ্রুত করতে সাহায্য করে এবং আপনাকে multi-domain environment-এ সহজে কাজ করতে দেয়।

*** **Distributed Database** বলতে এমন একটি ডেটাবেস সিস্টেমকে বোঝানো হয় যা একাধিক অবস্থানে বা সার্ভারে ছড়িয়ে থাকে। এতে একাধিক **physical locations** এ ডেটা সংরক্ষিত থাকে, কিন্তু ব্যবহারকারীর জন্য এটি একটি একক ডেটাবেস সিস্টেম হিসেবে কাজ করে।

সহজ ভাষায়:

ধরা যাক, আপনার তথ্য একটি সেন্ট্রাল সার্ভারে থাকলে সেটা একটি **centralized database**। কিন্তু যদি আপনার ডেটা বিভিন্ন স্থানে, বিভিন্ন সার্ভারে ভাগ করা থাকে এবং সেগুলোর মধ্যে যোগাযোগ থাকে, তখন সেটি একটি **distributed database**।

উদাহরণ:

- Google Drive, Dropbox, এবং **Facebook-**এর মত ক্লাউড স্টোরেজ সিস্টেমগুলির মধ্যে ব্যবহারকারীর তথ্য এবং ফাইলগুলি একাধিক সার্ভারে ছড়িয়ে থাকে। তবে ব্যবহারকারী অনুভব করেন যে তার সব তথ্য একটি একক সিস্টেমে পাওয়া যাচ্ছে। এটি একটি **distributed database**।

Distributed Database এর উপকারিতা:

1. High Availability: যদি একটি সার্ভার ডাউন হয়ে যায়, অন্য সার্ভারে থাকা কপি থেকে ডেটা পাওয়া যাবে।

2. Fault Tolerance: একাধিক জায়গায় ডেটা থাকার ফলে, যদি এক জায়গার সার্ভার সমস্যা হয়, তবুও সিস্টেম চলমান থা কে।

3. Load Balancing: ডেটা বিভিন্ন সার্ভারে বিভক্ত থাকলে, সার্ভারের উপর চাপ কমানো যায় এবং সিস্টেমের পারফরম্যান্স বাড়ানো যায়।

উদাহরণ:

- Active Directory: এটি একটি **distributed database**। এটি বিভিন্ন **domain controllers** এ ছড়িয়ে থাকে এবং একে অপরের সাথে সিঙ্ক্রোনাইজড থাকে, যাতে একটি ডোমেইনের মধ্যে সমস্ত ইউজার এবং কম্পিউটার অ্যাকাউন্ট সবার কাছে আপডেট থাকে।

সংক্ষেপে:

Distributed Database হল এমন একটি ডেটাবেস সিস্টেম যেখানে ডেটা একাধিক সার্ভারে বা অবস্থানে সংরক্ষিত থা কে, কিন্তু এটি ব্যবহারকারীর কাছে একটি একক ডেটাবেসের মত কাজ করে। এটি সিস্টেমের **scalability, availability,** এবং **fault tolerance** উন্নত করতে সহায়তা করে।

Directory Service হলো একটি সিস্টেম যা নেটওয়ার্কে থাকা ব্যবহারকারীদের এবং উপকরণের **(resources)** তথ্য সংগঠিত, সংরক্ষিত এবং অ্যাক্সেসযোগ্য রাখে। এটা এক ধরনের ডেটাবেস, যেখানে তথ্য **searchable** বা খোঁজা যায় এবং **centralized** ভাবে ব্যবস্থাপনা করা হয়। **Directory Service** এমন একটি সিস্টেম যা নেটওয়ার্কে থাকা সমস্ত তথ্য **(**যেমন ইউজার, কম্পিউটার, প্রিন্টার, ইত্যাদি**)** এক জায়গায় সংরক্ষণ ও ব্যবস্থাপনা করে, যাতে এই তথ্যগুলি সহজে অ্যাক্সেস এবং নিরাপদভাবে ব্যবহৃত হয়।

সহজ ভাষায়:

ধরা যাক, আপনার অফিসের সব কম্পিউটার, প্রিন্টার, ইউজার অ্যাকাউন্ট ইত্যাদি যদি একটি একক জায়গায় সংরক্ষিত থা কে, এবং আপনি যেকোনো ডিভাইস বা ইউজারের তথ্য সহজে খুঁজে বের করতে পারেন, তবে সেটিই হলো **Directory Service**।

মূল সুবিধা:
1. Centralized Management: এক জায়গায় সব তথ্য থাকে এবং সেখান থেকে সহজে অ্যাক্সেস করা যায়।
2. Data Organization: তথ্য সহজে খোঁজা যায় এবং কার্যকরভাবে সাজানো থাকে।
3. Security: ইউজারদের বিভিন্ন স্তরের অ্যাক্সেস নিয়ন্ত্রণ করা যায়।

*** পূর্ণাঙ্গ **Directory Service** বললে আমরা বুঝি এমন একটি সিস্টেম যেখানে ডেটা সিঙ্ক্রোনাইজেশন, নিরাপত্তা, এবং স্কেলেবিলিটির মতো গুরুত্বপূর্ণ ফিচার থাকে। **Windows NT 4.0** এর **Directory Service** ছিল প্রাথমিক এবং সীমিত, কিন্তু **Active Directory (Windows 2000** এবং পরবর্তী সংস্করণে) এ পূর্ণাঙ্গ **Directory Service** যুক্ত হয় যা আধুনিক ও উন্নত ব্যবস্থাপনা সমর্থন করে।

Windows NT 4.0 তে:
- শুধু **user accounts** এবং **groups** ব্যবস্থাপনা করা হতো।
- ডেটা সঠিকভাবে **replicate (**পুনরুদ্ধার**)** বা **distribute (**বিভাগীকৃত**)** হতো না। অর্থাৎ, একাধিক **server** বা **machine** এ তথ্য সিঙ্ক্রোনাইজেশন বা অটোমেটিক আপডেটের সুবিধা ছিল না।
- এটি **centralized** ছিল, কিন্তু **distributed** বা **scalable** ছিল না।

Second Level Address Translation (SLAT) হল একটি প্রযুক্তি যা **Windows** অপারেটিং সিস্টেম এবং অন্যান্য ভার্চুয়ালাইজেশন প্ল্যাটফর্মে ব্যবহৃত হয়। এটি মূলত **CPU** এর মধ্যে অন্তর্নিহিত কিছু প্রযুক্তি যা **virtualization (**যেমন **Hyper-V** বা অন্য ভার্চুয়াল মেশিন পরিচালনা করার সিস্টেম**)** কে আরও দ্রুত এবং কার্যকরভাবে কাজ করতে সাহায্য করে।

সহজ ভাষায় ব্যাখ্যা:

যখন আপনি একটি ভার্চুয়াল মেশিন **(VM)** চালান, তখন সেই **VM-**এর জন্য অপারেটিং সিস্টেম ও প্রোগ্রামগুলি একটি ভিন্ন ঠিকানা ব্যবহার করে কাজ করে। কিন্তু মূল হার্ডওয়্যার বা সিস্টেম **Physical Memory Address** এর সাথে ভার্চুয়াল মেমরি ঠিকানা মানিয়ে নিয়ে কাজ করে। একে বলে **Address Translation**।

এখন, **Second Level Address Translation (SLAT)** এর মাধ্যমে **CPU** আরও দ্রুত এই ভার্চুয়াল মেমরি ঠিকানা এবং শারীরিক ঠিকানা **(Physical Address)** এর মধ্যে **translation (**অর্থাৎ, মানানসই ঠিকানা নির্ধারণ**)** করতে সক্ষম হয়। এর ফলে ভার্চুয়াল মেশিনে সিস্টেমের পারফরম্যান্স উন্নত হয়, বিশেষ করে যখন একাধিক **VM** একসাথে চলতে থাকে।

উদাহরণ**:**
ধরা যাক, আপনার কাছে দুটি ভার্চুয়াল মেশিন চলছে। একটির জন্য **VM1** এবং অন্যটির জন্য **VM2**। যদি **SLAT** প্রযুক্তি না থা কে, তাহলে প্রতি **VM** এর জন্য **address translation** করতে **CPU**কে **extra work** করতে হয়। কিন্তু **SLAT** এর মাধ্যমে **CPU** এই কাজটি আরও দ্রুত এবং দক্ষভাবে করতে পারে, যা **overall system performance** বাড়ায়।

মোটা মু টি **:SLAT** হল**hardware-level support** যা ভার্চুয়াল মেশিনের **memory management** কে আরও দ্রুত এবং দক্ষ করে তোলে।
- এটি **Hyper-V** বা অন্য ভার্চুয়ালাইজেশন সফটওয়্যারকে দ্রুত এবং সঠিকভাবে কাজ করতে সাহায্য করে।

Registry Editor হলো**Windows** অপারেটিং সিস্টেমের একটি টুল যা আপনাকে **Windows Registry-**তে সরা সরি পরিবর্তন করতে দেয়। **Windows Registry** হল একটি ডাটাবেস যেখানে সিস্টেমের কনফিগারেশন সেটিংস, অপারেটিং সিস্টেম এবং ইনস্টল করা প্রোগ্রা মগুলোর তথ্য সংরক্ষিত থাকে।

কখন **Registry Editor** ব্যবহার করা উচিত**?**

Registry Editor ব্যবহার করার প্রয়োজন হতে পারে যখন**:**

1. অ্যাপ্লিকেশন বা সিস্টেম সেটিংস পরিবর্তন করতে হয়**:**
- কিছু সফটওয়্যার এবং সিস্টেম কনফিগারেশন শুধুমাত্র **Registry-**তে পরিবর্তন করে অ্যাক্সেসযোগ্য হয়।
- উদাহরণস্বরূপ, **Windows-**এ কিছু ফিচার বা সিস্টেম সেটিংস **(**যেমন স্টার্ট মেনু, ডেস্কটপ কাস্টমাইজেশন, এবং বিভিন্ন টু লস**)** শুধুমাত্র **Registry-**তে পরিবর্তন করে কনফিগার করা যেতে পারে।

2. অ্যাপ্লিকেশন বা সিস্টেম সমস্যা সমাধান**:**
- কখনও কখনও, কিছু অ্যাপ্লিকেশন বা সিস্টেম সমস্যা **Registry** পরিবর্তন বা মুছে ফেলার মাধ্যমে সমাধান করা যায়।

- উদাহরণস্বরূপ, যদি কোনও সফটওয়্যার ঠিকভাবে কাজ না করে, তবে তা র **Registry-**তে কিছু সেটিংস পরিবর্তন করে তা পুনরুদ্ধার করা যেতে পারে।

3. প্রস্তুতিপর্ব বা **Advanced Configurations:**

- কিছু **advanced** কনফিগারেশন যেমন নিরাপত্তা সেটিংস, পলিসি সেটিংস, প্রিন্টার এবং নেটওয়ার্ক কনফিগারেশনের পরিবর্তন, সাধারণত **Registry-**এর মাধ্যমে করা হয়।

4. বুট টুথ বা ড্রাইভার কনফিগারেশন**:**

- কিছু হার্ডওয়্যার ড্রাইভার এবং তাদের সেটিংস **Registry-**তে সংরক্ষিত থাকে, এবং সেগুলিকে কাস্টমাইজ করতে **Registry Editor** ব্যবহার করা যেতে পারে।

Data Execution Prevention (DEP) হল একটি নিরাপত্তা প্রযুক্তি যা কম্পিউটারে সিস্টেম এবং প্রোগ্রামের নিরাপত্তা বাড়ানোর জন্য ডিজাইন করা হয়েছে। এটি বিশেষভাবে **memory** এর মধ্যে **malicious code (**যেমন **viruses** বা **malware)** চালানো বন্ধ করার জন্য ব্যবহৃত হয়।

ধরা যাক, আপনি একটি প্রোগ্রাম চালাচ্ছেন, এবং কোনো ভাইরাস প্রোগ্রামটি আপনার কম্পিউটারের **data segment-**এ কোড চালানোর চেষ্টা করছে। **DEP** এর মাধ্যমে এটি ধরা পড়বে এবং সেই কোডটি চলতে দেওয়া হবে না, ফলে ভাইরাসটি আপনার কম্পিউটারকে ক্ষতিগ্রস্ত করতে পারবে না।

সংক্ষেপে**:**
DEP একটি নিরাপত্তা ফিচার যা মেমরি থেকে ম্যালওয়্যার চালানো বন্ধ করে এবং **buffer overflow** বা **malicious code execution** এর মতো আক্রমণ থেকে সিস্টেমকে রক্ষা করে।

AD Class, Objects এবং Attributes:

- **Classes:** একটি শ্রেণি বা ধরনের ডাটা সংজ্ঞায়িত করে, যেমন **User** ক্লাস যা সমস্ত ইউজার অবজেক্টের জন্য গঠিত।**Class** হলো **Object** এর ধরন বা শ্রেণী। অব্জেক্টটি **User** হবে নাকি অব্জেক্টটি একটি **Computer** হবে সেটা ডিফাইন করে **Class**।
- **Objects** হলো ডিরেক্টরির সমস্ত এক্সটেনশান বা প্রকারের এন্ট্রি যেমন ইউজার, কম্পিউটার, গ্রুপ, প্রিন্টার, এবং সার্ভিস।
- **Attributes** হলো সেই অবজেক্টের বৈশিষ্ট্য বা তথ্য। যেমন একটি ইউজারের জন্য **First Name, Last Name, Email,** ইত্যাদি।

সহজভাবে বললে, **Class** হলো এমন একটি "শ্রেণী" বা "ক্যাটাগরি", যার অধীনে অনেক **objects** রাখা যায়, এবং প্রতিটি **object** এর কিছু নির্দিষ্ট **attributes** থা কে।

উদাহরণ দিয়ে বুঝানো যাক:

1. Class: User

- **Object:** এটি একটি **User object** হতে পারে, যেমন একজন আসল ব্যবহারকারী: **John Doe**।
- **Attributes:** এর মধ্যে থাকবে **First Name, Last Name, Email Address, Phone Number,** ইত্যাদি।

2. Class: Computer

- **Object:** এটি একটি **Computer object** হতে পারে, যেমন একটি কম্পিউটার: **Computer01.**
- **Attributes:** এর মধ্যে থাকবে **Computer Name, IP Address, Operating System,** ইত্যাদি।

Schema কী?

একটি সহজ উদাহরণ দিয়ে শুরু করা যাক। ধরুন, আপনি একটি নতুন ডাটাবেজ তৈরি করছেন। এই ডাটাবেজে আপনি বিভিন্ন ধরনের তথ্য সংরক্ষণ করবেন, যেমন ব্যক্তির নাম, বয়স, ঠিকানা ইত্যাদি। এই তথ্যগুলোকে কীভাবে সংরক্ষণ করবে ন, তা নির্ধারণ করার জন্য আপনাকে একটি স্ট্রাকচার তৈরি করতে হবে। এই স্ট্রাকচারকেই **Schema** বলা হয়।

Windows Server-এর ক্ষেত্রে Schema কী?

Windows Server-এ একটি **Active Directory** ডোমেইন থাকে। এই ডোমেইনে **Computer, User, Group** ইত্যাদি বিভিন্ন অবজেক্ট থাকে। এই অবজেক্টগুলোর সম্পর্ক, তাদের গঠন এবং তাদের মধ্যে কী ধরনের তথ্য রাখা যাবে, তা নির্ধারণ করে একটি **Schema**।

Schema কেন গুরুত্বপূর্ণ?

- স্ট্রাকচার প্রদান করে: **Schema** একটি সুনির্দিষ্ট স্ট্রাকচার / কাঠামো প্রদান করে, যার ফলে ডোমেইনে সবকিছু সুশৃঙ্খলভাবে থাকে।
- সমন্বয় রক্ষা করে: বিভিন্ন অবজেক্টের মধ্যে সমন্বয় রক্ষা করে।
- নতুন অবজেক্ট তৈরি করা সহজ করে: নতুন অবজেক্ট তৈরি করার সময় **Schema** একটি গাইডলাইন হিসেবে কাজ করে ।
- ডাটা সুরক্ষা: **Schema** ডাটা সুরক্ষায় সাহায্য করে।

একে সুনির্দিষ্ট কাঠামো বলার কারণ:

- নির্দিষ্ট ধরনের তথ্য: **Schema** নির্দিষ্ট করে যে কোন অবজেক্টে কী ধরনের তথ্য রাখা যাবে। উদাহরণস্বরূপ, একটি ব্যবহারকারী অবজেক্টে নাম, ইমেইল, ফোন নাম্বর ইত্যাদি থাকতে পারে, কিন্তু তার জন্মতারিখ রাখা যাবে না।

- সঠিক সম্পর্ক: **Schema** নির্ধারণ করে যে কোন অবজেক্ট অন্য কোন অবজেক্টের সাথে কী ধরনের সম্পর্ক থাকতে পারে। উদাহরণস্বরূপ, একটি ব্যবহারকারী একটি গ্রুপের সদস্য হতে পারে, কিন্তু একটি কম্পিউটারের সাথে সরাসরি কোন সম্পর্ক থাকতে পারে না।
- নির্দিষ্ট গঠন: **Schema** অবজেক্টের গঠন নির্ধারণ করে। উদাহরণস্বরূপ, একটি ব্যবহারকারী অবজেক্টে নাম একটি বাধ্যতামূলক তথ্য হতে পারে, কিন্তু ফোন নাম্বর একটি ঐচ্ছিক তথ্য।
- সুসংগতি: **Schema** নিশ্চিত করে যে ডোমেইনের সকল তথ্য একই ধরনের কাঠামো অনুসরণ করে। এটি ডাটাবেজের সুসংগতি বজায় রাখতে সাহায্য করে।

বিভিন্ন অবজেক্টের মধ্যে সমন্বয় রক্ষা করে - বলার কারণ:

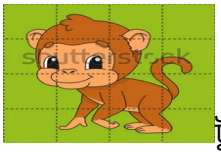
Active Directory ডোমেইনে অনেক ধরনের অবজেক্ট থাকে, যেমন ব্যবহারকারী, কম্পিউটার, গ্রুপ, পরিষেবা ইত্যাদি। এই অবজেক্টগুলোর মধ্যে একটি নির্দিষ্ট সম্পর্ক থাকে। উদাহরণস্বরূপ, একটি ব্যবহারকারী একাধিক গ্রুপের সদস্য হতে পারে, একটি কম্পিউটার একটি নির্দিষ্ট **Organizational Unit**-এর অধীনে থাকতে পারে ইত্যাদি। **Schema** এই সম্পর্কগুলোকে সংজ্ঞায়িত করে। এটি নির্দিষ্ট করে যে কোন অবজেক্ট অন্য কোন অবজেক্টের সাথে কী ধরনের সম্পর্ক স্থাপন করতে পারে।

উদাহরণ:

ধরুন, একটি কোম্পানির **Active Directory** ডোমেইনে একটি নতুন বিভাগ যোগ করা হল। এই ক্ষেত্রে, **Schema**-তে একটি নতুন **Organizational Unit** তৈরি করা হবে। এই নতুন **Organizational Unit**-এর সাথে বিভাগের সকল কর্মচারীকে যুক্ত করা হবে। এভাবে **Schema** নিশ্চিত করে যে সকল কর্মচারী তাদের নিজ নিজ বিভাগের সাথে যুক্ত থাকবে।

তাহলে সহজ করে এটাকে এভাবে বলা যেতে পারে:

এটা সবকিছুকে একসাথে জোড়া রাখে। এটা নিশ্চিত করে যে সবকিছু নিজের জায়গায় আছে এবং একে অপরের সাথে ঠিকভাবে কাজ করছে।এটা তেমন, যেমন কিনা একটি পাজলের **(Puzzle)**



টুকরোগুলোকে একসাথে মিলিয়ে একটি সম্পূর্ণ ছবি তৈরি করে।

অর্থাৎ, **Schema** সবকিছুকে একটি নির্দিষ্ট ক্রমে সাজিয়ে রাখে এবং একে অপরের সাথে সম্পর্কিত করে।

আরও কিছু উদাহরণ:

- একটি লাইব্রেরির বইয়ের তাকের মতো। প্রতিটি বই নিজস্ব জায়গায় থাকে এবং বিষয় অনুযায়ী সাজানো থাকে।
- একটি বাড়ির নকশার মতো। প্রতিটি ঘর নিজস্ব কাজের জন্য নির্ধারিত এবং একে অপরের সাথে সংযুক্ত।

Active Directory Schema Master:

Active Directory Schema Master হল একটি বিশেষ ধরনের **Domain Controller** যার প্রধান কাজ হল **Active Directory** ডোমেইনের মূল কাঠামো বা **Schema** পরিচালনা করা।

Active Directory Schema Master হল একটি বিশেষ **FSMO role** যা **Active Directory**-র **Schema** পরিবর্তন বা আপডেট করার কাজটি পরিচালনা করে। এটি এক ধরনের "নিয়ন্ত্রক" বা "মা স্টা র" যা **Active Directory**-তে **Schema**-তে কোনো ধরনের কোনো পরিবর্তন বা আপডেট করলে তা পুরো **forest**-এ প্রভাব ফেলে।

বাস্তব উদাহরণ:

ধরা যাক, আপনার কোম্পানির নাম **ABC Ltd.** এবং আপনার ৫০০ জন কর্মচারী আছেন। আপনি **"Employee ID"** না মকএকটি নতু ন **attribute** তৈরি করতে চান, যা প্রত্যেক কর্মচারীর জন্য একটি ইউনিক আইডি সংরক্ষণ করবে।

1. Schema Master-এ আপডেট:

- আপনি **Active Directory**-র **Schema Master**-এ গিয়ে নতুন **"Employee ID" attribute** যোগ করেন।

2. সকল **Domain Controllers**-এ ছড়িয়ে পড়া:

- এখন, যেহেতু আপনি **Schema Master**-এ এই পরিবর্তন করেছেন, এটি পুরো **ABC Ltd.** সংস্থার সমস্ত **Domain Controllers**-এ ছড়িয়ে পড়বে, এবং **Employee ID attribute** আগামীতে কোনো নতুন বা বিদ্যমান ব্যবহারকারীর জন্য সক্রিয় হবে।

3. কোনো ভুল বা অসঙ্গতি ঘটলে:

- যদি কেউ অন্য **Domain Controller**-এ**Schema** পরিবর্তন করার চেষ্টা করে, তা **Schema Master**-এ অনুমোদিত না হওয়া পর্যন্ত কোনো পরিবর্তন করা সম্ভব হবে না। এইভাবে, **Schema Master** নিশ্চিত করে যে **Schema** পরিবর্তনগুলি সঠিকভাবে ও সুরক্ষিতভাবে পরিচালিত হয় এবং পুরো **Active Directory environment**-এ সামঞ্জস্য থা কে।

সারাংশ: এভাবে **Schema Master** নিশ্চিত করে যে **Active Directory Schema**-তে কোনো পরিবর্তন সঠিকভাবে প্রয়োগ করা হচ্ছে এবং সেগুলো পুরো **forest**-এ সমন্বিতভাবে কাজ করছে।

Active Directory-র **Schema Master**-এ গিয়ে নতুন **attribute** যেমন **"Employee ID"** যোগ করার জন্য আপনাকে কিছু নির্দিষ্ট পদক্ষেপ অনুসরণ করতে হবে। **Schema**-তে পরিবর্তন করা একটু জটিল কাজ হতে পারে, কারণ **Schema**-এর পরিবর্তনগুলি অনেক গুরুত্বপূর্ণ এবং এগুলি পুরো **Active Directory forest**-এ প্রভাব ফেলে।

এটি করতে **Active Directory Schema snap-in** এবং কিছু **PowerShell** কমান্ড ব্যবহার করতে হয়। এই কাজটি করতে হলে আপনার কাছে **Schema Master**-এ অ্যাডমিনিস্ট্রেটিভ অ্যাক্সেস থাকতে হবে এবং আপনাকে কিছু গুরুত্বপূর্ণ সতর্কতা মেনে চলতে হবে, কারণ **Schema** পরিবর্তন করলে তা সবার উপর প্রভাব ফেলতে পারে।

নিচে যেভাবে যেভাবে বলেছি সেটা অনুসরণ করে ঠিক সেইভাবে সেইভাবে সার্ভারে কাজ করুন।

১. **Schema Management Snap-in** ইনস্টল করা

Schema snap-in ডিফল্টভাবে **Windows Server**-এ ইনস্টল করা থাকে না। প্রথমে এটি ইনস্টল করতে হবে।

1. Run উইন্ডো খোলার জন্য **Windows + R** চাপুন, তারপর টাইপ করুন **regsvr32 schmmgmt.dll** এবং এন্টার চাপুন।

2. এটি স্কিমা ম্যানেজমেন্ট টুলটিকে সিস্টেমে নিবন্ধন করবে। তারপর, **MMC (Microsoft Management Console)** খুলতে হবে।

২. **MMC** খুলে **Schema Management Snap-in** যুক্ত করা

1. Run উইন্ডো **(Windows + R)** খুলুন এবং টাইপ করুন **mmc** এবং **Enter** চাপুন।

2. MMC টু লে **File** মে নু থে কে **Add/Remove Snap-in** নির্বাচন করুন।

3. Add বা টনে ক্লি ককরু ন, তারপর **Active Directory Schema** নির্বাচন করুন এবং **Add** ক্লি ককরু ন।

4. এবা র **OK** বা টনে ক্লি ককরু ন।

৩. **Schema Master**-এ পরিবর্তন করা

এখন আপনি **Schema Master**-এ গিয়ে নতুন **attribute** যোগ করতে পারবেন।

1. Active Directory Schema স্ন্যাপ-ইনটি খোলার পর, **Attributes** বিভাগে যান।

2. ডা ন ক্লি ক করে **Create Attribute** নির্বাচন করুন।

3. একটি নতু ন **"Employee ID" attribute** তৈরি করতে কিছু ইনফরমেশন পূর্ণ করুন, যেমন:

- **Common Name: Employee ID**
- **LDAP Display Name: employeeID**
- **Syntax: String** (বা আপনার প্রয়োজন অনুযায়ী অন্য কিছু)
- **Length:** যেমন **10** (যদি এটি একটি নির্দিষ্ট দৈর্ঘ্যের আইডি হয়)

4. এটি তৈরি করার পরে, আপনি এটিকে **User class** বা অন্য কোন **Object class-**এ যুক্ত করতে পারবেন।

৪. New Attribute Add করা Object Class-এ

আপনি যদি **"Employee ID" attribute** টি **User object class-**এ যুক্ত করতে চান, তা হলে:

1. Object Classes বিভাগে যান এবং **User** ক্লাসটি খুঁজুন।

2. সেখানে ক্লিক করে **Add** বাটনে ক্লিক করুন এবং আপনার নতুন **"Employee ID" attribute** যোগ করুন।

৫. পরিবর্তন করা সম্পূর্ণ হলে

যখন আপনি **"Employee ID" attribute** তৈরি এবং যুক্ত করবেন, তখন এটি **Active Directory forest-**এর প্রতিটি **Domain Controller-**এ আপডেট হবে। এই **attribute-**টি পরবর্তীতে ব্যবহারকারীর তথ্যের অংশ হিসেবে যোগ করা যাবে এবং বিভিন্ন অ্যাপ্লিকেশন বা স্ক্রিপ্টে ব্যবহৃত হতে পারবে।

৬. PowerShell ব্যবহার করে Attribute যোগ করা

PowerShell ব্যবহার করেও আপনি **Active Directory Schema-**তে **attribute** যোগ করতে পারেন। উদাহরণস্বরূপ, নতু ন **"Employee ID" attribute** যোগ করার জন্য **PowerShell** কমান্ড ব্যবহার করা যেতে পারে।

Powershell

নতু ন **attribute** তৈরি করার জন্য **PowerShell** কমা ন্ড

```
New-ADSchemaAttribute -Name "employeeID" -Syntax "String" -Length 10
```

সতর্ক তা :

- Schema পরিবর্তন খুবই গুরুত্বপূর্ণ: **Active Directory-**র **Schema** পরিবর্তন করলে তার প্রভাব পুরো **forest-**এ পড়ে। তাই, **Schema** পরিবর্তন করার আগে অবশ্যই ব্যাকআপ নিন এবং কাজটি যত্নসহকারে করুন।

- Schema Master-এর একক ভূমিকা: **Schema Master-**এর ভূমিকা একমাত্র **Domain Controller-**এ থা কে, এবং এটি **Schema** পরিবর্তন করতে পারে। এটি একটি অত্যন্ত শক্তিশালী ভূমিকা, তাই এই রোলের অধিকারী **Domain Controller-**এ কাজ করার সময় সতর্ক থাকুন।

সারাংশ:

Active Directory Schema Master-এনতু ন **attribute** যোগ করতে হলে, আপনাকে **Schema Management snap-in** ব্যবহার করতে হবে, এবং নতুন **attribute** তৈরি করে সেটি প্রয়োজনে **Object Class-**এ যুক্ত করতে হবে। তবে, **Schema** পরিবর্তন খুবই ঝুঁকিপূর্ণ হতে পারে, তাই সঠিক পরিকল্পনা, ব্যাকআপ এবং সতর্কতার সাথে কাজ করা জরুরি।

ডোমেইন নেমিং মাস্টার রোলের প্রধান দায়িত্ব হল:

- ডোমেইন যোগ/অপসারণ: ডিরেক্টরি থেকে ডোমেইন যোগ করা বা অপসারণ করা।
- ক্রস রেফারেন্স পরিচালনা: অন্যান্য ডিরেক্টরির সাথে ক্রস রেফারেন্স যোগ করা বা অপসারণ করা।

⇒ **Domain Naming Master** হল **Active Directory-**তে একটি বিশেষ ভূমিকা **(FSMO role)**, যা মূলত ডোমেইন এবং ফরেস্ট নামের ব্যবস্থাপনা করে। এটি নির্দিষ্ট করে যে, নতুন ডোমেইন যুক্ত করা বা পুরনো ডোমেইন মুছে ফেলা হবে কি না ।

Domain Naming Master- এর কাজ:

Domain Naming Master মূলত নতুন ডোমেইন তৈরি বা পুরনো ডোমেইন মুছে ফেলা সময় একক নিয়ন্ত্রক হিসেবে কাজ করে। এটি নিশ্চিত করে যে, একই নামের দুটি ডোমেইন **Active Directory-**তে না থাকে এবং ডোমেইন সংক্রান্ত নামের সমস্যাগুলো এড়ানো যায়।

সহজ ভাষায়:
ধরা যাক, আপনার একটি **Active Directory Forest** আছে এবং আপনি এতে নতুন একটি ডোমেইন যুক্ত করতে চান। **Domain Naming Master** এই নতুন ডোমেইনটি যোগ করার অনুমতি দেয়, এবং একে নিশ্চিত করে যে নামের কোনো দ্বন্দ্ব বা কনফ্লিক্ট হচ্ছে না। এটি কোনও একটি ডোমেইন নাম পরিবর্তন বা মুছে ফেলা হলে, সেই কাজটিও সম্পন্ন করে।

উদাহরণ:

ধরা যাক, আপনার প্রতিষ্ঠানের নাম **"ABC Corp."** এবং বর্তমানে আপনার **Active Directory forest** এ একটি ডোমেইন আছে যার নাম **"corp.abc"**। এখন আপনি **"sales.abc"** নামে একটি নতুন ডোমেইন তৈরি করতে চান।

1. Domain Naming Master-এ নতুন ডোমেইন যোগ করার কাজ**:**

- আপনি **"sales.abc"** নামে একটি নতুন ডোমেইন তৈরির চেষ্টা করবেন।

- এই ক্ষেত্রে **Domain Naming Master** এই কাজটি অনুমোদন করবে এবং নিশ্চিত করবে যে, **"sales.abc"** না মটি অন্য কোনো ডোমেইন **(**যেমন, **"corp.abc")** এর সা থে মি লছে না ।

- নতুন ডোমেইন **"sales.abc" Active Directory forest-**এ সফলভাবে যুক্ত হবে।

2. Domain Naming Master-এ ডোমেইন মুছে ফেলা**:**

- ধরা যাক, আপনি **"sales.abc"** ডোমেইনটি আর ব্যবহার করতে চান না এবং এটি মুছে ফেলতে চান।

- এই ক্ষেত্রে, **Domain Naming Master-**ই অনুমতি দেবে এবং ডোমেইনটি **Active Directory** থেকে পুরোপুরি মুছে ফেলবে।

Domain Naming Master-র ভূমিকা কেন গুরুত্বপূর্ণ**?**

- নামের দ্বন্দ্ব এড়ানো**:** যদি দুটি ডোমেইনের একই নাম থাকে, তা **Active Directory-**তে সমস্যা তৈরি করতে পারে। **Domain Naming Master** এই দ্বন্দ্ব এড়াতে সাহায্য করে।

- নতুন ডোমেইন যোগ করা**:** নতুন ডোমেইন যুক্ত করার জন্য এটি একমাত্র জায়গা যেখানে অনুমতি দেওয়া হয়।

- ডোমেইন মুছে ফেলা**:** যখন কোনো ডোমেইন মুছে ফেলা হয়, তখন সেটি পুরো **forest** থেকে নিরাপদভাবে বাদ দেয়।

সারাংশ**:**

Domain Naming Master রোলহল**Active Directory-**তে নতুন ডোমেইন তৈরি এবং পুরনো ডোমেইন মুছে ফেলার জন্য দায়িত্বশীল। এটি নিশ্চিত করে যে, ডোমেইন নামের মধ্যে কোনো দ্বন্দ্ব বা কনফ্লিক্ট নেই এবং ডোমেইন সংক্রান্ত সমস্ত কার্যক্রম সঠিকভাবে পরিচালিত হচ্ছে।

সারাংশ**:**

Domain Naming Master রোলহল**Active Directory-**তে নতুন ডোমেইন তৈরি এবং পুরনো ডোমেইন মুছে ফেলার জন্য দায়িত্বশীল। এটি নিশ্চিত করে যে, ডোমেইন নামের মধ্যে কোনো দ্বন্দ্ব বা কনফ্লিক্ট নেই এবং ডোমেইন সংক্রান্ত সমস্ত কার্যক্রম সঠিকভাবে পরিচালিত হচ্ছে।

⇒ Cross References কি?

Cross reference হলো এক ধরনের রেফারেন্স যা একটি ডোমেইন থেকে অন্য ডোমেইন বা ডিরেক্টরি-এ অবজেক্ট বা ডেটার সম্পর্ক স্থাপন করে। এটি বিশেষত তখন প্রয়োজন হয় যখন দুটি আলাদা ডোমেইন বা ডিরেক্টরির মধ্যে অবজেক্টের তথ্য সম্পর্কিত থাকে, যেমন**:**

- এক ডোমেইনে থাকা একটি ইউজার, যা অন্য ডোমেইনে একটি গ্রুপের সদস্য।

- একটি ডোমেইন থেকে অন্য ডোমেইনে অবজেক্টের সম্পর্ক ব্যবহার করতে বা প্রত্যক্ষ করতে **cross reference** প্রয়োজন।

প্রকৃত উদাহরণ**:**

ধরা যাক, আপনার **Active Directory-**তে দুটি ডোমেইন রয়েছে**:**

- domainA.com
- domainB.com

এখন, যদি আপনি **domainA.com** ডোমেইন থেকে **domainB.com** ডোমেইন**-**এর কিছু অবজেক্টে অ্যাক্সেস করতে চান, যেমন **domainA.com-**এর ইউজারকে **domainB.com-**এ একটি গ্রুপের সদস্য বানাতে চান, তা হলে **Domain Naming Master** এই **cross reference** তৈরি করবে।

সারাংশ**:**

Domain Naming Master রোলটি **cross references** যোগ বা মুছে ফেলা এবং নতুন ডোমেইন যোগ করার জন্য দায়িত্বশীল। এটি অন্য ডিরেক্টরি বা অন্য ডোমেইন থেকে রেফারেন্স বা সম্পর্ক স্থাপন করার সময় ব্যবহার হয়, যা **Active Directory-**র মধ্যে বিভিন্ন ডোমেইন বা ডিরেক্টরির মধ্যে সম্পর্ক স্থাপন করে।

PDC Emulator (Primary Domain Controller Emulator)

হল একটি FSMO role (Flexible Single Master Operations role), যা Active Directory-এর একটি গুরুত্বপূর্ণ ভূমিকা। এটি মূলত ডোমেইন কন্ট্রোলার এর মধ্যে সিঙ্ক্রোনাইজেশন এবং নেটওয়ার্কের প্রাথমিক ক্ষমতা বজায় রাখার দায়িত্বে থাকে।

PDC Emulator- এর কাজ:

1. Backwards Compatibility:
   - PDC Emulator কাজ করে এমন ভাবে যে পুরনো Windows NT 4.0-based domain controllers এর সা থে নতু ন domain controllers সামঞ্জস্যপূর্ণভাবে কাজ করতে পারে।

2. Password Changes (পাসওয়ার্ড পরিবর্তন):
   - যখন ব্যবহারকারী তার পাসওয়ার্ড পরিবর্তন করে, PDC Emulator-এ সেই পরিবর্তনটি প্রথমে রেকর্ড করা হয় এবং পরে তা বাকি ডোমেইন কন্ট্রোলার-গুলিতে সিঙ্ক্রোনাইজ হয়। অন্য ডোমেইন কন্ট্রোলারগুলি পাসওয়ার্ড পরিবর্তনের তথ্য তখন PDC Emulator থেকে গ্রহণ করে।

3. Time Synchronization (সময়ের সিঙ্ক্রোনাইজেশন):

- PDC Emulator নেটওয়ার্কে সময়ের সিঙ্ক্রোনাইজেশন নিশ্চিত করে। এটি ডোমেইন কন্ট্রোলারের মধ্যে একটি সময়সীমা নির্ধারণ করে, যাতে সমস্ত কন্ট্রোলার এবং ক্লায়েন্ট মেশিনগুলির সময় সঠিক থাকে। এটি বিশেষ করে Kerberos Authentication এর জন্য গুরুত্বপূর্ণ, যেহেতু সঠিক সময় ছাড়া Kerberos কাজ করবে না।

4. Group Policy Changes (গ্রুপ পলিসি পরিবর্তন):
- PDC Emulator Group Policy Objects (GPO) এর কোনো পরিবর্তন প্রথমে গ্রহণ করে এবং অন্য ডোমেইন কন্ট্রোলারগুলিতে সেগুলি পুশ করে। এটি Group Policy updates-এর সিঙ্ক্রোনাইজেশন নিশ্চিত করে।

5. Lockout (Account Lockout):
- যদি কোনো ব্যবহারকারী ভুল পাসওয়ার্ড দিয়ে লগইন করার চেষ্টা করে এবং তার অ্যাকাউন্ট লক হয়ে যায়, তা হলে PDC Emulator-ই মূলত অ্যাকাউন্ট লকআউট তথ্য আপডেট করবে এবং বাকি ডোমেইন কন্ট্রোলারগুলিতে এই তথ্য সিঙ্ক্রোনাইজ হবে।

সারাংশ:
PDC Emulator রোলটি মূলত পাসওয়ার্ড পরিবর্তন, সময় সিঙ্ক্রোনাইজেশন, গ্রুপ পলিসি আপডেট, অ্যাকাউন্ট লকআউট এবং অন্য কিছু গুরুত্বপূর্ণ সিঙ্ক্রোনাইজেশন কাজ পরিচালনা করে। এটি একটি খুব গুরুত্বপূর্ণ ভূমিকা, কারণ এটি Active Directory ডোমেইন কন্ট্রোলারের মধ্যে সঠিক তথ্য সমন্বয় এবং সিঙ্ক্রোনাইজেশন নিশ্চিত করে।

একটি সহজ উদাহরণ দিয়ে বোঝানো:

ধরা যাক, আপনার একটি domain আছে যেখানে অনেক domain controllers রয়েছে। এখন, যদি কোনো ব্যবহারকারী তাদের password পরিবর্তন করে, PDC Emulator নিশ্চিত করে যে সেই পরিবর্তনটি সঠিকভাবে domain controllers এর মধ্যে সিঙ্ক্রোনাইজ হবে এবং সেই ব্যবহারকারী যেন অন্য কোনো domain controller থেকে লগইন করতে গিয়ে পুরানো password দিয়ে প্রবেশ না করতে পারে।

সংক্ষেপে:
- PDC Emulator হলো Active Directory-এর একটি বিশেষ FSMO role, যা password changes, time synchronization, এবং backward compatibility নিশ্চিত করার জন্য ব্যবহৃত হয়।
- এটি domain controllers এর মধ্যে সঠিক সমন্বয় এবং সিঙ্ক্রোনাইজেশন নিশ্চিত করে।

RID Master (Relative Identifier Master) হল Active Directory-র একটি FSMO role (Flexible Single Master Operations role), যা Active Directory domain এর মধ্যে রিড (RID) বরাদ্দ এবং সমন্বয় করে।

RID Master Role এর কাজ:

1. RID (Relative Identifier) বরাদ্দ করা:
- RID Master হল সেই Domain Controller (DC) যা Active Directory ডোমেইনের জন্য RID pools তৈরি এবং বরাদ্দ করে।
- RID হল একটি আনকোয়ালিফাইড সঠিক চিহ্ন যা ডোমেইন এর মধ্যে ইউজার, কম্পিউটার, গ্রুপ এবং অন্যান্য অবজেক্টের জন্য একে অপর থেকে আলাদা করা যায়।
- যখন একটি নতুন অবজেক্ট (যেমন ইউজার, গ্রুপ, কম্পিউটার) তৈরি হয়, তখন RID প্রয়োজন হয় তাকে সুনির্দিষ্টভাবে চিহ্নিত করার জন্য। RID Master এর দায়িত্ব হল RID pool এর বরাদ্দ।

2. RID Pools বরাদ্দ করা:
- একটি Domain Controller যখন নতুন অবজেক্ট তৈরি করে, তখন RID Master তা কে RID pool থে কে RID বরাদ্দ করে ।
- RID Master একাধিক ডোমেইন কন্ট্রোলারকে RID pool বরাদ্দ করে এবং তাদের মধ্যে সঠিকভাবে বরাদ্দ করা RID গুলি সরবরাহ করে, যাতে প্রতিটি ডোমেইন কন্ট্রোলার সঠিকভাবে একটি অবজেক্ট তৈরি করতে পারে।
- প্রতিটি Domain Controller-এ কিছু সীমিত RID pool থা কে, এবং যখন ওই pool শেষ হয়ে যায়, তখন RID Master-এর কাছ থেকে আরেকটি pool বরাদ্দ করতে হয়।

3. RID Pool Exhaustion (RID pool শেষ হয়ে যাওয়া):
- যদি RID pool শেষ হয়ে যায় এবং RID Master এর কা ছ থে কে নতু ন RID pool না পাওয়া যায়, তা হলে নতু ন কোনো অবজেক্ট (যেমন নতুন ইউজার, গ্রুপ, কম্পিউটার) তৈরি করা যাবে না। এই পরিস্থিতি এড়ানোর জন্য RID Master নিয়মিত RID pool বরাদ্দ করে থাকে।

4. একটি Domain Controller থেকে অন্য Domain Controller-এ RID Pool-এর স্থা না ন্ত র:
- RID Master মূলত বিশ্বস্ত এবং একমাত্র RID Pool বরাদ্দকারী। যদি RID Master ডোমেইনে অপ্রাপ্য হয়, তবে অন্য Domain Controller-কে নতু ন RID pool বরাদ্দ করার জন্য সক্ষম হতে হয় না।

5. RID Master Failover:
- যদি RID Master কোনো কারণে অসামর্থ্য হয়ে যায়, তবে আপনাকে FSMO role স্থানান্তর করে অন্য Domain Controller-কে RID Master হিসেবে নিয়োগ করতে হবে। তবে, এই স্থানান্তর শুধুমাত্র একটি পদ্ধতিতে করা যাবে, যা সমস্ত RID pool বরাদ্দের কাজ চালিয়ে যাবে।

RID Master রোলের গুরুত্ব:
- Active Directory-তে RID গুরুত্বপূর্ণ কারণ এটি প্রত্যেকটি অবজেক্টের জন্য একটি একক এবং পৃথক পরিচিতি (ID) প্রদান করে।
- RID Master এই RID বরাদ্দের কাজ সঠিকভাবে সমন্বয় করে, এবং এটি নতুন ইউজার, কম্পিউটার, গ্রুপ, বা অন্য কোনো অবজেক্ট তৈরি করার জন্য অপরিহার্য।

ধরা যাক, আপনার ডোমেইনে নতুন একটি ইউজার অ্যাকাউন্ট তৈরি করতে হবে। তখন RID Master এই ইউজার অ্যাকাউন্টের জন্য একটি নতুন RID বরাদ্দ করবে। RID বরাদ্দের পর, ইউজার অ্যাকাউন্টটি সিস্টেমে তৈরি হবে এবং প্রতিটি ডোমেইন কন্ট্রোলার RID-টি সমন্বয় করতে পারবে।

সারাংশ:
RID Master রোলটি RID pool বরাদ্দ ও সমন্বয় করার জন্য দায়ী, যা Active Directory-এর মধ্যে নতুন অবজেক্ট তৈরির জন্য প্রয়োজনীয়। RID Master ছাড়া, নতুন অবজেক্ট তৈরি করা সম্ভব নয়, কারণ RID একটি গুরুত্বপূর্ণ উপাদান যা প্রতি অবজেক্টের জন্য আলাদা এবং পৃথক চিহ্ন সরবরাহ করে।

***** RID** এবং **SID** সম্পর্কিত সঠিক ব্যাখ্যা**:**

1. SID (Security Identifier):
  - SID হল একটি ইউনিক আইডেন্টিফায়ার যা ডোমেইন বা সিস্টেমে প্রতিটি অবজেক্ট (যেমন ইউজার, গ্রুপ, কম্পিউটার) কে চিহ্নিত করতে ব্যবহৃত হয়।
  - SID যখন একটি অবজেক্ট তৈরি হয়, তখন সেটি সম্পূর্ণ ডোমেইনের জন্য একটি বিশেষ আইডি তৈরি করা হয়। SID-তে একটি অংশ থাকে যা ডোমেইনের ইউনিক আইডেন্টিফায়ার হিসেবে কাজ করে এবং অবজেক্টের জন্য একটি ইউনিক RID (Relative Identifier) থা কে।
  - উদাহরণস্বরূপ, SID দেখতে এমন হতে পারে:
    S-1-5-21-1234567890-1234567890-1234567890-1000
    - এখানে, S-1-5-21 হলো ডোমেইনের unique identifier, আর 1000 হলো RID যা ঐ ইউজারের জন্য নির্দিষ্ট।

2. RID (Relative Identifier):
  - RID হলো SID এর একটি অংশ যা Active Directory ডোমেইনে প্রতিটি নতুন অবজেক্টকে আলাদা চিহ্নিত করার জন্য ব্যবহৃত হয়।
  - প্রতিটি নতুন অবজেক্টের জন্য RID বরাদ্দ করা হয়, এবং এটি ডোমেইন কন্ট্রোলারের মধ্যে সঠিকভাবে বিভাজিত হয়।
  - RID Master রোলটি RID pool বরাদ্দ করে, যাতে নতুন অবজেক্টের জন্য সঠিক RID ব্যবহার করা যায়।

RID এবং SID এর সম্পর্ক:
- SID তৈরি হয়, যার মধ্যে RID থা কে।
- SID-এর মধ্যে RID হল সেই নম্বর যা নির্দিষ্ট অবজেক্টকে আলাদা করে চিহ্নিত করে।

সারাংশ:
- RID Master SID তৈরি করে না। বরং, SID-এর মধ্যে RID থা কে, এবং RID বরাদ্দ করার জন্য RID Master রোলটি কাজ করে।

SID (Security Identifier) তৈরি করে Active Directory বা Windows সিস্টেম যখন কোনো নতুন অবজেক্ট (যেমন ইউজার, গ্রুপ, কম্পিউটার) তৈরি হয়।

SID তৈরি করার প্রক্রিয়া:

1. SID Structure:
- SID একটি ইউনিক আইডেন্টিফায়ার যা সিস্টেমের প্রতিটি অবজেক্টকে আলাদা ভাবে চিহ্নিত করে। একটি SID দুটি মূল অংশের সমন্বয়ে তৈরি হয়:
  - Domain Identifier: যা ডোমেইন বা সিস্টেমের পরিচিতি দেয়।
  - RID (Relative Identifier): যা ডোমেইনের মধ্যে প্রতিটি ইউনিক অবজেক্টকে চিহ্নিত করে।

2. SID Creation Process:
- SID তৈরি করার সময়, RID অংশটি RID Master এর মাধ্যমে বরাদ্দ করা হয়। RID হল অবজেক্টের জন্য নির্দিষ্ট একটি সংখ্যা (যেমন 1000, 1001, ইত্যাদি), যা ডোমেইনের মধ্যে ইউনিক।
  - SID-এর প্রথম অংশ (যেমন S-1-5-21-...) ডোমেইনের ইউনিক আইডেন্টিফায়ার, যা একটি নির্দিষ্ট ডোমেইন বা সিস্টেমের জন্য তৈরি হয়।

3. SID Creation Example:
- SID Example:
  S-1-5-21-1234567890-1234567890-1234567890-1000
  - S-1-5-21: এটি ডোমেইনের ইউনিক পরিচিতি (Domain Identifier)।
  - 1000: এটি RID (Relative Identifier), যা প্রতিটি নতুন অবজেক্টের জন্য ইউনিক থাকে।

SID কে তৈরি করে?
- SID তৈরি হয় যখন নতুন ইউজার, গ্রুপ, কম্পিউটার বা অন্য কোনো অবজেক্ট তৈরি হয়। Windows সিস্টেম বা Active Directory স্বয়ংক্রিয়ভাবে SID তৈরি করে এবং সেই অবজেক্টের সাথে যুক্ত করে।
- RID Master রোল RID বরাদ্দ করে, যা পরে SID অংশ হিসেবে ব্যবহার হয়।

সারাংশ:
- SID তৈরি করে Active Directory বা Windows সিস্টেম যখন নতুন অবজেক্ট তৈরি করা হয়।
- RID Master RID বরাদ্দ করে, যা SID-এর অংশ হয়ে থাকে এবং অবজেক্টের ইউনিক চিহ্ন হিসেবে কাজ করে।

SID তৈরি করা একটি FSMO role এর কাজ নয়। SID (Security Identifier) তৈরি করার কাজ Active Directory বা Windows Operating System স্বয়ংক্রিয়ভাবে করে, যখন কোনো নতুন অবজেক্ট (যেমন ইউজার, গ্রুপ, কম্পিউটার) তৈরি হয়। তবে, FSMO roles (Flexible Single Master Operations roles) কিছু গুরুত্বপূর্ণ পরিচালনামূলক কাজ

করে Active Directory সি স্টে মে , কিন্তু SID তৈরি করা তাদের মধ্যে কোনো রোলের কাজ নয়।- SID দুটি অংশের সমন্বয়ে তৈরি হয়:

- Domain Identifier: যা ডোমেইনের ইউনিক পরিচিতি প্রদান করে।
- RID (Relative Identifier): যা ডোমেইনের মধ্যে ইউনিক থাকে এবং একটি নির্দিষ্ট অবজেক্টকে চিহ্নিত করে। RID Master রোলের মাধ্যমে RID বরাদ্দ করা হয়, তবে SID তৈরির দায়িত্ব Active Directory-র।

Schema Master	Domain Naming Master	PDC Emulator (Primary Domain Controller)	RID Master (Relative Identifier Master)	Infrastructure Master

Distributed system

এমন একটি সিস্টেম যেখানে বিভিন্ন কম্পিউটার বা ডিভাইস একে অপরের সাথে সংযুক্ত থাকে এবং একটি যৌথভাবে কাজ করার জন্য একসাথে কাজ করে, তবে প্রতিটি ডিভাইস বা কম্পিউটার আলাদা এবং পৃথকভাবে কাজ করতে সক্ষম।

Distributed System এর মূল বৈশিষ্ট্য:

1. Multiple Components: এতে একাধিক কম্পিউটার বা ডিভাইস থাকে, যা একে অপরের সাথে যোগাযোগ করে এবং কাজ করে।

2. Independent Nodes: প্রতিটি কম্পিউটার বা নোড **(node)** আলাদা এবং স্বাধীনভাবে কাজ করতে পারে, তবে তা রা একসাথে কাজ করার জন্য সমন্বিত হয়।

3. Communication: নোডগুলো একে অপরের সাথে নেটওয়ার্কের মাধ্যমে যোগাযোগ করে, যেমন ইন্টারনেট বা লোকাল নেটওয়ার্ক **(LAN)**।

4. Scalability: এটি সহজে স্কেল করা যায়, মানে প্রয়োজনমতো নতুন ডিভাইস বা নোড যুক্ত করা সম্ভব।

5. Fault Tolerance: এক বা একাধিক কম্পিউটার বা ডিভাইস বিকল হয়ে গেলে, অন্যান্য কম্পিউটার বা নোড দিয়ে সিস্টেম চালু রাখা যায়।

6. Resource Sharing: বিভিন্ন নোড একে অপরের রিসোর্স যেমন প্রিন্টার, ডা টা , প্রসেসিং পাওয়ার শেয়ার করে।

Distributed System এর উদাহরণ:

- Cloud Computing: যেমন **Google Cloud, Amazon Web Services (AWS)** যেখানে অনেক সার্ভার একে অপরের সাথে কাজ করে।

- **File Servers:** অনেক ফাইল সার্ভার একে অপরের সাথে সংযুক্ত থাকে এবং ডাটা শেয়ারিং করে।
- **Email Systems:** ইমেইল সার্ভিস যেমন **Gmail** বা **Outlook,** যেগুলোর সিস্টেম একাধিক সার্ভার নিয়ে গঠিত।

উদাহরণ:
ধরা যাক, একটি **online shopping** সাইট, যেখানে:
- **Web Server** এবং **Database Server** আলাদা আলাদা কম্পিউটারে কাজ করে।
- **Payment Gateway Server** আরেকটি সিস্টেমের মাধ্যমে কাজ করছে।
- সকল সার্ভার একে অপরের সাথে তথ্য আদান-প্রদান করে এবং পুরো সিস্টেম মিলে কাজ করছে।

এই সমস্ত সিস্টেমগুলো একে অপরের সাথে সিঙ্ক্রোনাইজড হয়ে কাজ করতে থাকে এবং একসাথে তারা **Distributed System** গঠন করে।

সংক্ষেপে:
Distributed system বলতে এমন একটি সিস্টেম বোঝায়, যেখানে একাধিক কম্পিউটার বা ডিভাইস একসাথে কাজ করতে পারে, তবে প্রত্যেকে আলাদা এবং স্বতন্ত্র।

Distributed System এর বিপরীতে যে সিস্টেমটি থাকে তা হলো **Centralized System** ।

Centralized System এর মূল বৈশিষ্ট্য:

1. Single Central Authority: এখানে একটি কেন্দ্রীয় সার্ভার বা সিস্টেম সমস্ত কাজ পরিচালনা করে এবং অন্যান্য উপকরণগুলি সেই কেন্দ্রীয় সার্ভারের সাথে সংযুক্ত থাকে।

2. Single Point of Control: সমস্ত সিদ্ধান্ত এবং নিয়ন্ত্রণ একটি নির্দিষ্ট কেন্দ্রীয় জায়গা থেকে নেওয়া হয়।

3. Limited Resources: সাধারণত একটি নির্দিষ্ট জায়গাতেই সমস্ত রিসোর্স থাকে, এবং অন্যান্য ডিভাইসগুলো এসব রিসোর্স ব্যবহার করে।

4. Reliability Issue: যদি কেন্দ্রীয় সিস্টেমে কোনো সমস্যা হয়, পুরো সিস্টেমটি থমকে যেতে পারে, কারণ সকল নোড বা ডিভাইস সেই একক সিস্টেমের ওপর নির্ভরশীল থাকে।

Centralized System এর উদাহরণ:

- **Mainframe Computers:** যেখানে একটি বড় কম্পিউটার বা সার্ভার সমস্ত কাজ করে এবং অন্যান্য কম্পিউটার সেগুলোর মাধ্যমে কাজ করে।
- **Traditional Database Servers:** যেখানে একটি প্রধান সার্ভার সব ডাটা ম্যানেজ করে, এবং সমস্ত ক্লায়েন্ট সেই এক সার্ভারের সাথে যোগাযোগ করে ডাটা এক্সেস করে।

Distributed System এবং Centralized System এর মধ্যে পার্থক্য:

বৈশিষ্ট্য	Distributed System	Centralized System
কন্ট্রোল	একাধিক ডিভাইস বা নোড দ্বারা কন্ট্রোল করা হয়	একটি একক কেন্দ্রীয় সার্ভার দ্বারা কন্ট্রোল হয়
প্রযুক্তি	নেটওয়ার্ক এবং মাল্টিপল ডিভাইসের মাধ্যমে কাজ করা হয়	একটি মূল কম্পিউটার বা সার্ভারে কাজ করা হয়
বিকল হওয়ার ঝুঁকি	একাধিক নোড থাকলে অন্য নোডগুলো কাজ চালিয়ে যেতে পারে	একটি সার্ভারে সমস্যা হলে পুরো সিস্টেম বন্ধ হয়ে যেতে পারে
রিসোর্স শেয়ারিং	বিভিন্ন নোডের মধ্যে রিসোর্স শেয়ার করা যায়	একক সার্ভারের মধ্যে রিসোর্স শেয়ারিং সীমিত থাকে

সোজা ভাষায়:

- **Distributed System:** একাধিক কম্পিউটার বা ডিভাইস একে অপরের সাথে কাজ করে।
- **Centralized System:** একটি সেন্ট্রাল সার্ভার সবকিছু নিয়ন্ত্রণ করে এবং অন্যান্য ডিভাইস শুধুমাত্র ওই সার্ভারের সাথে কাজ করে।

Active Directory (AD) domain এ যে **multiple domains** থা কে, তা একটি **distributed system** এর মতো কাজ করে।

কেন এটি **distributed system?**

1. Multiple Domain Controllers: Active Directory domain এ **multiple domain controllers** থা কে। প্রতিটি **domain controller** তার নিজস্ব কপি **(replica)** থা কে **Active Directory** ডে টা বে সে র, এবং তারা একে অপরের সাথে সিঙ্ক্রোনাইজড **(replicated)** থাকে। অর্থাৎ, একাধিক সার্ভার একসাথে কাজ করে এবং তাদের মধ্যে তথ্য শেয়ার করে, যা **distributed system** এর বৈশিষ্ট্য।

2. Decentralized Control: একাধিক **domain** থাকলেও প্রতিটি **domain** এর নিজস্ব **administrative control** থা কে। এক্টি **domain** এর প্রশাসক শুধুমাত্র ওই **domain** এর জন্য দায়িত্বশীল, অন্য **domain** এর জন্য নয়। এভাবে ব্যবস্থাপনা একটি কেন্দ্রীভূত জায়গায় সীমাবদ্ধ না থেকে, অনেক জায়গায় বিস্তৃত হয়।

3. Replication: Active Directory এর মধ্যে ডেটা **replication** হয়, অর্থাৎ, এক**domain controller** থেকে অন্য **domain controller** এ ডেটা কপি হয়। এই **replication** প্রক্রিয়া সিস্টেমকে আরও বিশ্বাসযোগ্য এবং **fault-tolerant** করে তোলে, কারণ ডেটা একাধিক জায়গায় থাকে।

4. Scalability: Active Directory কে **multiple domain** ব্যবহার করে স্কেল করা যায়। প্রতিটি **domain controller** বা **domain** আলাদা রকম ডেটা এবং ব্যবস্থাপনা করে, কিন্তু সবগুলো সিস্টেম একে অপরের সাথে সংযুক্ত থা কে। নতু ন **domain controller** বা **domain** সহজেই যোগ করা যায়।

উদাহরণ:
ধরা যাক একটি কোম্পানিতে একাধিক অফিস রয়েছে এবং প্রতিটি অফিসের জন্য আলাদা **Active Directory domain** তৈরি করা হয়েছে। প্রতিটি অফিসের **domain controller** তাদের নিজস্ব **domain** এর ডেটাবেস মেইনটেইন করে, তবে তারা একে অপরের সাথে সিঙ্ক্রোনাইজড থাকে। এই সিস্টেমটি একটি **distributed system** এর মতো কাজ করে।

Centralized System এর তুলনায়:
- Centralized System এ সাধারণত একটি সার্ভার বা ডিভাইস সমস্ত কন্ট্রোল এবং ডেটা পরিচালনা করে, যেখানে **Active Directory** এ **multiple, independent servers** থা কে, যা একসাথে কাজ করে।

সংক্ষেপে:
Active Directory domain এবং এর **multiple domains** এক্টি **distributed system** এর মতো কাজ করে, কারণ এখানে একাধিক **server (domain controller)** তথ্য সিঙ্ক্রোনাইজড এবং ভাগ করে কাজ করে।

যদি মাত্র **1**টি **Domain Controller** থা কে, তবে সে টি এক্টি **Centralized System** এর মতো কাজ করবে।

কেন এটি **Centralized System** হবে ?

1. Single Point of Control: এক্মা ত্র **Domain Controller** সমস্ত **Active Directory** এর তথ্য এবং ডেটাবেস পরিচালনা করে। সমস্ত ডেটা এবং কন্ট্রোল একটি একক সার্ভারে থাকে, অর্থাৎ কোনও **distributed** প্রকৃতি নেই।

2. No Replication: এক্মা ত্র **Domain Controller** থা কলে ডে টা র **replication** বা সিঙ্ক্রোনাইজেশন হয় না, কারণ শুধু একটি সার্ভারেই সমস্ত ডেটা রাখা হচ্ছে। অন্য কোন সার্ভার বা ডিভাইসের সাথে ডেটা শেয়ার করা হচ্ছে না।

3. Single Point of Failure: যদি একমাত্র **Domain Controller** ত্রুটিপূর্ণ হয়ে যায় বা কাজ করতে না পারে, তা হলে পুরো **Active Directory** সিস্টেমের কাজ বন্ধ হয়ে যেতে পারে, কারণ আর কোন **backup domain controller** নেই। এটি **centralized** সিস্টেমের **typical feature**।

তুলনা :
- Distributed System: এখানে একাধিক সার্ভার বা ডিভাইস থাকে, এবং ডেটা একাধিক জায়গায় থাকে (যেমন, **multiple domain controllers with replication in Active Directory).**
- Centralized System: এখানে সমস্ত কন্ট্রোল এবং ডেটা একটি একক সার্ভার বা ডিভাইসে থাকে (যেমন, **only one domain controller in Active Directory).**

সংক্ষেপে:
যদি মাত্র **1**টি **Domain Controller** থাকে, তবে এটি **Centralized System** হিসেবেই কাজ করবে।

যদি একটি **Root Domain Controller** এবং একটি **Additional Domain Controller (ADC)** থাকে, তবে এটি **Distributed System** হবে।

কেন এটি **Distributed System?**

1. Multiple Domain Controllers: এখানে দুটি **Domain Controllers (Root Domain Controller** এবং **Additional Domain Controller)** রয়েছে, যেগুলি একে অপরের সাথে **replicate** (সিঙ্ক্রোনাইজ) করে। অর্থাৎ, ডেটা এবং কন্ট্রোল একাধিক জায়গায় (ডিভাইসে) ভাগ হয়ে থাকে।

2. Data Replication: Root Domain Controller এবং **Additional Domain Controller** এর মধ্যে **replication** হয়, যার ফলে ডেটা একাধিক জায়গায় থাকে এবং দুটি ডোমেইন কন্ট্রোলার পরস্পর একে অপরের সাথে ডেটা সিঙ্ক্রোনাইজ করে থাকে।

3. Fault Tolerance: যদি একটি **Domain Controller** ব্যর্থ হয়, অন্যটি কার্যক্রম চালিয়ে যেতে পারে কারণ ডেটা দুটি জায়গায় রয়েছে। এটি **distributed system** এর একটি বৈশিষ্ট্য।

4. Decentralized Control: এখানে একক সার্ভার বা ডিভাইসের উপরে সমস্ত কন্ট্রোল নয়, বরং একাধিক **Domain Controllers** এই কন্ট্রোল ভাগ করে নেয়।

Centralized System এর তুলনায়:
- Centralized System: যেখানে একক সার্ভার সমস্ত ডেটা এবং কন্ট্রোল পরিচালনা করে। যদি একটিই **Domain Controller** থাকে, তা হলে সেটি একটি **centralized** সিস্টেম হবে।

- Distributed System: এখানে একাধিক **Domain Controllers** থাকে এবং ডেটা ও কন্ট্রোল একাধিক জায়গায় বিতরণ করা হয়, এবং সেগুলোর মধ্যে সিঙ্ক্রোনাইজেশন বা **replication** ঘটে।

সংক্ষেপে:
যদি একটি **Root Domain Controller** এবং একটি **Additional Domain Controller** থাকে, তবে এটি **Distributed System** হবে , কারণ এখানে ডেটা এবং কন্ট্রোল একাধিক সার্ভারে ভাগ হয়ে থাকে এবং সেগুলোর মধ্যে **replication** হয়।

==========

SOA রেকর্ড

SOA রেকর্ড কি?

SOA (Start of Authority) রেকর্ড হল ডোমেইন নেম সিস্টেম **(DNS)** এর একটি বিশেষ ধরনের রেকর্ড যা কোনো নির্দিষ্ট ডোমেইনের জন্য বিভিন্ন ধরনের প্রশাসনিক তথ্য ধারণ করে। এটি একটি জোনের **(zone)** শুরুতে থাকে এবং সেই জোন সম্পর্কে মূল তথ্য দেয়।

একটি **SOA** রেকর্ডে সাধারণত নিম্নলিখিত তথ্য থাকে:

- **Primary Name Server (NS):** কোন নেম সার্ভার এই জোনের জন্য প্রাথমিকভাবে দায়ী।
- **Email:** যদি কোন সমস্যা হয় তাহলে কাকে যোগাযোগ করতে হবে, সেই ব্যক্তির ইমেইল অ্যাড্রেস।
- **Serial Number:** এই নাম্বারটি পরিবর্তন হলে বোঝা যায় যে জোনটি আপডেট হয়েছে।
- **Refresh Interval:** অন্য নেম সার্ভারগুলোকে কতক্ষণ পর পর এই জোনের তথ্য আপডেট করতে হবে।
- **Retry Interval:** যদি কোন নেম সার্ভার তথ্য আপডেট করতে ব্যর্থ হয়, তাহলে কতক্ষণ পর পর আবার চেষ্টা করবে ।
- **Expire Time:** অন্য নেম সার্ভারগুলোকে কতক্ষণ পর পর এই জোনের তথ্যকে অবৈধ হিসেবে চিহ্নিত করতে হবে ।
- **Minimum TTL:** ক্যাশেতে এই জোনের তথ্য কতক্ষণ পর্যন্ত রাখা যাবে।

SOA রেকর্ডের কাজ:

- জোন পরিচয়: কোন নেম সার্ভার এই জোনের জন্য দায়ী তা নির্ধারণ করে।
- জোন আপডেট: অন্য নেম সার্ভারগুলোকে কখন জোনের তথ্য আপডেট করতে হবে তা নির্দেশ করে।
- ত্রুটি সমাধান: যদি কোন সমস্যা হয়, তাহলে কাকে যোগাযোগ করতে হবে তা জানায়।
- **DNS** ক্যাশিং: ক্যাশেতে এই জোনের তথ্য কতক্ষণ পর্যন্ত রাখা যাবে তা নির্দেশ করে।

সহজ কথায়:

SOA রেকর্ড হল একটি ডোমেইনের পরিচয়পত্রের মতো। এটি ডোমেইন সম্পর্কে মূল তথ্য দেয় এবং অন্য নেম সার্ভারগুলোকে এই ডোমেইনের তথ্য কীভাবে পরিচালনা করতে হবে তা নির্দেশ করে।

উদাহরণ:

```
; SOA record for example.com
example.com.        86400  IN      SOA    ns1.example.com. admin.example.com. (
                    2023010101          ; Serial
                    3600                ; Refresh
                    600                 ; Retry
                    604800              ; Expire
                    86400 )             ; Negative Cache TTL
```

SOA রেকর্ড হল একটি ডোমেইনের পরিচয়পত্র। যেমন, একটি বাড়ির নামফলক বা একটি কোম্পানির আইডি কার্ড। এই পরিচয়পত্রে ডোমেইন সম্পর্কে সবচেয়ে গুরুত্বপূর্ণ তথ্য থাকে।

এই তথ্যগুলো কেন গুরুত্বপূর্ণ?

- ডোমেইনটি কার? **SOA** রেকর্ডে ডোমেইনটি কার মালিকানাধীন, সেই ব্যক্তি বা সংস্থার ইমেইল আছে।
- ডোমেইনের তথ্য কোথায় পাওয়া যাবে? এই রেকর্ডে বলা থাকে যে, ডোমেইন সম্পর্কিত সব তথ্য কোন সার্ভারে রয়েছে।
- কখন তথ্য আপডেট হবে? এই রেকর্ডে বলা থাকে যে, ডোমেইনের তথ্য কতক্ষণ পর পর আপডেট হবে।

একটি উদাহরণ দিলে ব্যাপারটা আরো স্পষ্ট হবে:

ধরো, আপনার একটি ওয়েবসাইট আছে এবং এর ডোমেইন নাম হল "আপনারওয়েবসাইট.কম"। এই ডোমেইনের জন্য একটি **SOA** রেকর্ড থাকবে, যেখানে লেখা থাকবে:

- মালিক: আপনি
- ইমেইল: আপনার ইমেইল
- তথ্যের জায়গা: আপনার ওয়েব হোস্টিং কোম্পানির সার্ভার
- তথ্য আপডেটের সময়: প্রতিদিন

এই তথ্যগুলোর সাহায্যে, ইন্টারনেটের অন্যান্য কম্পিউটার বুঝতে পারবে যে, "আপনারওয়েবসাইট.কম" ডোমেইনটি কার, কোথায় এবং কখন আপডেট হবে।

সহজ কথায়:

SOA রেকর্ড হল একটি ডোমেইন সম্পর্কে সবচেয়ে গুরুত্বপূর্ণ তথ্যের সংগ্রহ। এটি ইন্টারনেটকে বোঝাতে সাহায্য করে যে, একটি ডোমেইন কী এবং কোথায় পাওয়া যাবে।

==========

DACL: কোনো অবজেক্টের (যেমন - ফাইল, ফোল্ডার) মালিক বা অ্যাডমিনিস্ট্রেটর ব্যবহারকারী বা গ্রুপের অ্যাক্সেস অধিকার নিয়ন্ত্রণ করার জন্য ব্যবহৃত হয়। অর্থাৎ, কে কোনো অবজেক্টের উপর কী ধরনের অ্যাকশন (যেমন - পড়া, লেখা, পরিবর্তন) করতে পারবে, তা **DACL** দ্বারা নির্ধারিত হয়।

SACL: কোনো অবজেক্টের অ্যাক্সেস এর অডিট করার জন্য ব্যবহৃত হয়। অর্থাৎ, কে কখন কোন অবজেক্টের উপর কী ধরনের অ্যাকশন করেছে, তার লগ তৈরি করার জন্য **SACL** ব্যবহার করা হয়। **SACL** মূলত সিস্টেম অ্যাডমিনিস্ট্রেটরদের জন্য, যারা সিস্টেমের নিরাপত্তা পর্যবেক্ষণ করতে চান।

মনে করুন, একটি অফিসের কথা।

- **DACL (Discretionary Access Control List):**
    - **DACL** হলো অফিসের "দরজা নিয়ন্ত্রণ ব্যবস্থা"। কোন ঘরে কে ঢুকতে পারবে, সেটা **DACL** ঠিক করে।
    - উদাহরণ: অফিসের প্রধান (মালিক বা অ্যাডমিনিস্ট্রেটর) ঠিক করলেন যে, "কর্মচারী **A**" শুধুমাত্র রিসেপশন রুমে যেতে পারবে, "কর্মচারী **B**" ক্যান্টিনে যেতে পারবে, এবং "কর্মচারী **C**" সার্ভার রুমে যেতে পারবে। অন্য কেউ অন্য রুমে যেতে পারবে না। এই নিয়মগুলোই হলো **DACL**।
    - অর্থাৎ, **DACL** নির্ধারণ করে, কার কী "অধিকার" আছে, কোথায় যাওয়ার এবং কী করার।
- **SACL (System Access Control List):**
    - **SACL** হলো অফিসের "সিসিটিভি ক্যামেরা"। কোন ঘরে কে কখন গেল, সেটা **SACL** রেকর্ড করে রাখে।
    - উদাহরণ: সিসিটিভি ক্যামেরা (**SACL**) সবসময় রেকর্ড করছে, কে কখন রিসেপশন রুমে গেল, কে কখন ক্যান্টিনে গেল, ইত্যাদি। এটা অফিসের প্রধান দেখতে পারেন, যাতে তিনি জানতে পারেন, কখন কে কোথায় ছিল।
    - অর্থাৎ, **SACL** অ্যাক্সেস "অডিট" করে, কে কী করেছে, তার প্রমাণ রাখে। এটা সাধারণত নিরাপত্তার জন্য ব্যবহার করা হয়।

তা হলে, মূল পার্থক্যটা হলো:

- **DACL:** "কার কী করার অনুমতি আছে" - এটা নিয়ন্ত্রণ করে। (যেমন - দরজা খুলে ভেতরে যাওয়া)
- **SACL:** "কে কী করেছে, তা রেকর্ড রাখা" - এটা নিরীক্ষণ করে। (যেমন - সিসিটিভি ফুটেজ দেখা)

আশা করি, এই উদাহরণ দিয়ে **DACL** এবং **SACL**-এর পার্থক্যটা আরও সহজে বুঝতে পেরেছেন।

উইন্ডোজ সার্ভারে "**Service Account**" হলো এক ধরনের বিশেষ ইউজার অ্যাকাউন্ট, যা কোনো নির্দিষ্ট অ্যাপ্লিকেশন বা সার্ভিসের অধীনে চলে। সাধারণ ইউজার অ্যাকাউন্টের মতো, সার্ভিস অ্যাকাউন্টেরও নিজস্ব পরিচয় থাকে, কিন্তু এর প্রধান কাজ হলো ব্যাকগ্রাউন্ডে অ্যাপ্লিকেশন বা সার্ভিস চালানো, ব্যবহারকারীর সরাসরি ইন্টার্যাকশন ছাড়াই।

ধরা যাক, আপনার **Windows Server-**এ কিছু সার্ভিস চলছে, যেমন **IIS (Web Server), SQL Server,** বা **File Sharing** সার্ভিস। এই সার্ভিসগুলো ব্যবহারকারী বা অ্যাডমিনিস্ট্রেটরের মতো সোজাসুজি লগইন বা কাজ করে না। পরিবর্তে, সার্ভিসগুলোকে বিশেষ একটি একাউন্ট দিয়ে পরিচালনা করা হয়। সেই একাউন্টকেই **Service Account** বলা হয়।

সহজ উদাহরণ**:**

ধরা যাক, আপনার একটা **Website** চালু আছে **IIS Web Server** দিয়ে, এবং এই সার্ভিসের জন্য আপনাকে বিশেষ প্রমাণীকরণ **(authentication)** করতে হয়।

- Service Account হলো সেই অ্যাকাউন্ট, যার মাধ্যমে **IIS** সার্ভিসটি **Windows Server-**এ লগইন করে এবং কাজ করে ।

- এই **Service Account** এর জন্য থাকে নির্দিষ্ট অধিকার **(permissions)** এবং এটি **IIS** সার্ভিসের জন্য নির্দিষ্ট নিরাপত্তা নিশ্চিত করে।

কিছু **Account** উদাহরণ**:**

1. Local System Account:

- **IIS** বা **SQL Server** সার্ভিস একটি **Local System Account** ব্যবহার করতে পারে।

- এটি একটি বিল্ট-ইন অ্যাকাউন্ট, যা সাধারণত সার্ভারে সমস্ত কাজ পরিচালনা করে।

2. Domain Service Account:

- যদি সার্ভিসটি **Network Resource** বা অন্যান্য সার্ভিস এর সাথে যোগাযোগ করতে চায়, তবে একটি **Domain Service Account** ব্যবহার করা হতে পারে। এটি ডোমেইন এর অংশ এবং **Network-**এ অ্যাক্সেস পায়।

3. Managed Service Accounts (MSAs):

- এই ধরনের অ্যাকাউন্টগুলি নতুন সুরক্ষা ফিচার সহ আসে, যেখানে **Windows Server self-managed** বা **automatic password rotation** প্রদান করে।

সহজ উদাহরণ দিয়ে বুঝানো যাক:

মনে করু ন, আপনার অফিসে একটি প্রিন্টিং মেশিন আছে, যা নেটওয়ার্কের সাথে যুক্ত। এই প্রিন্টিং মেশিনটি সবসময় চালু থাকে এবং যখন কেউ কম্পিউটার থেকে প্রিন্ট করার কমান্ড দেয়, তখন এটি স্বয়ংক্রিয়ভাবে প্রিন্ট করে দেয়। এই কাজটি করার জন্য প্রিন্টিং মেশিনের একটি "পরিচয়" দরকার, কারণ নেটওয়ার্কে প্রতিটি ডিভাইসের একটি পরিচয় থাকা আবশ্যক।

এখানে, প্রিন্টিং মেশিনের "পরিচয়" হিসেবে কাজ করে **"Service Account"**। এই অ্যাকাউন্টটি উইন্ডোজ সার্ভারে তৈরি করা হয় এবং প্রিন্টিং সার্ভিসের সাথে যুক্ত করা হয়। যখন কেউ প্রিন্ট করার কমান্ড দেয়, তখন প্রিন্টিং সার্ভিস এই সার্ভিস অ্যাকাউন্টের অধীনে প্রিন্টিংয়ের কাজটি করে। ব্যবহারকারীকে নিজে থেকে প্রিন্টিং মেশিনে গিয়ে কিছু করতে হয় না।

আরও কিছু সহজ উদাহরণ:

- একটি ওয়েব সার্ভার, যা ওয়েবসাইট হোস্ট করে, সেটিও একটি সার্ভিস অ্যাকাউন্টের অধীনে চলতে পারে। যখন কেউ ওয়েবসাইট ভিজিট করে, তখন ওয়েব সার্ভার সেই সার্ভিস অ্যাকাউন্টের মাধ্যমে ফাইলের অ্যাক্সেস করে এবং ওয়েবসাইট প্রদর্শন করে।
- একটি ডেটাবেস সার্ভার, যা তথ্য সংরক্ষণ করে, সেটিও একটি সার্ভিস অ্যাকাউন্টের অধীনে চলতে পারে। যখন কোনো অ্যাপ্লিকেশন ডেটাবেস থেকে তথ্য চায়, তখন ডেটাবেস সার্ভার সেই সার্ভিস অ্যাকাউন্টের মাধ্যমে ডেটাবেস থেকে তথ্য নিয়ে অ্যাপ্লিকেশনকে দেয়।
- একটি অনলাইন গেম: আপনি যখন গেম খেলেন, তখন গেমটি একটি সার্ভিস অ্যাকাউন্টের মাধ্যমে সার্ভারের সাথে যুক্ত হয় এবং আপনার স্কোর আপডেট করে।
- একটি এটিএম মেশিন: আপনি যখন এটিএম থেকে টাকা তোলেন, তখন এটিএম মেশিনটি একটি সার্ভিস অ্যাকাউন্টের মাধ্যমে আপনার ব্যাংক অ্যাকাউন্টের সাথে যুক্ত হয় এবং টাকা তোলার কাজটি করে।
- একটি স্মার্ট টিভি: আপনার স্মার্ট টিভি যখন নেটফ্লিক্স বা ইউটিউব চালায়, তখন এটি একটি সার্ভিস অ্যাকাউন্টের মাধ্যমে ইন্টারনেটের সাথে যুক্ত হয় এবং ভিডিও স্ট্রিম করে।

সার্ভিস অ্যাকাউন্টের সুবিধা:

- নিরাপত্তা: সার্ভিস অ্যাকাউন্ট সাধারণত নির্দিষ্ট কিছু কাজের জন্য অনুমতিপ্রাপ্ত হয়, যা সিস্টেমের নিরাপত্তা বাড়াতে সাহায্য করে। সাধারণ ইউজার অ্যাকাউন্টের মতো এর বেশি অধিকার থাকে না। সাধারণত, সার্ভিস অ্যাকাউন্টের **password** এবং **permissions** অটোমেটিকভাবে পরিচালিত হয়, যা নিরাপত্তা নিশ্চিত করে।

- স্বয়ংক্রিয়তা: সার্ভিস অ্যাকাউন্ট ব্যাকগ্রাউন্ডে কাজ করে, তাই ব্যবহারকারীর সরাসরি ইন্টার্যাকশনের প্রয়োজন হয় না। এতে কাজ দ্রুত এবং সহজে হয়।

- <u>নির্ভরযোগ্যতা:</u> সার্ভিস অ্যাকাউন্ট সবসময় চালু থাকে, যতক্ষণ না সার্ভার বন্ধ করা হয়। এতে অ্যাপ্লিকেশন বা সার্ভিসের নিরবচ্ছিন্নতা বজায় থাকে।

- <u>কোনো ব্যবহারকারী বা অ্যাডমিনিস্ট্রেটরের প্রয়োজন নেই:</u> সার্ভিসগুলো সার্ভিস একাউন্টের মাধ্যমে চালিত হয়, সুতরাং মানুষের লগইন করার প্রয়োজন নেই।

উপরে আমরা "সাধারণত, সার্ভিস অ্যাকাউন্টের **password** এবং **permissions** অটোমেটিকভাবে পরিচালিত হয়, যা নিরাপত্তা নিশ্চিত করে।" অটোমেটিকভাবে পরিচালিত হলে কি সেটা আসলে শক্তিশালী হয়**?** জি অবস্যই। ম্যানুয়াল পাসওয়ার্ড সেট করা থেকে শক্তিশালী হবে।

1. পাসওয়ার্ড ম্যানেজমেন্ট **(Password Management):**

- ম্যানুয়াল পাসওয়ার্ড পরিবর্তন অনেক ঝামেলা এবং ভুলের সুযোগ দেয়। সার্ভিস অ্যাকাউন্টের পাসওয়ার্ড যদি মানবিক ভুলের কারণে খুব সহজে অনুমানযোগ্য হয় বা বারবার একই পাসওয়ার্ড ব্যবহৃত হয়, তাহলে নিরাপত্তা ঝুঁকি বাড়ে।

- স্বয়ংক্রিয়ভাবে পাসওয়ার্ড পরিবর্তন করার ক্ষেত্রে, সিস্টেম নিজে নিয়মিত ভাবে শক্তিশালী পাসওয়ার্ড তৈরি এবং পরিবর্তন করতে পারে, যা **brute-force attack** বা **password guessing** প্রতিরোধে সহায়ক।

2. পাসওয়ার্ড সংরক্ষণ এবং এক্সেস কন্ট্রোল **(Password Storage and Access Control):**

- যখন পাসওয়ার্ড ***manual***ভাবে পরিচালনা করা হয়, তখন সেগুলি সঠিকভাবে সংরক্ষণ করা এবং নিরাপদে রাখা কঠিন হতে পারে, বিশেষ করে যদি পাসওয়ার্ড স্টোরেজ প্রক্রিয়া সুরক্ষিত না থাকে।

- স্বয়ংক্রিয় ব্যবস্থায় পাসওয়ার্ডগুলি নিরাপদ স্টোরেজে রাখা হয় এবং অ্যাক্সেস কন্ট্রোল অত্যন্ত কড়া থাকে। এছাড়াও, **service accounts** সাধারণত **least privilege** ভিত্তিতে কাজ করে, অর্থাৎ তাদের কাছে শুধুমাত্র তাদের কাজের জন্য প্রয়োজনীয় **permissions** দেওয়া হয়।

3. নিরাপত্তা টোকেন ও ক্রিপ্টোগ্রাফি **(Security Tokens and Cryptography):**

- Managed services যেমন **Windows Server** নিজেই সার্ভিস অ্যাকাউন্টের জন্য শক্তিশালী নিরাপত্তা টোকেন এবং **cryptographic methods** ব্যবহার করে।

- এই টোকেনগুলির মাধ্যমে সার্ভিসটি **authentication** এবং **authorization** প্রক্রিয়া সম্পন্ন করে, যা খুবই সুরক্ষিত।

4. ইন্টিগ্রেটেড সিস্টেম **(Integrated System):**

- যখন পাসওয়ার্ড এবং **permissions** স্বয়ংক্রিয়ভাবে পরিচালিত হয়, তখন এটি **Active Directory** বা অন্যান্য নিরাপত্তা সিস্টেমের মাধ্যমে ইন্টিগ্রেটেড থাকে। এর ফলে, নিরাপত্তা ব্যবস্থার সাথে সার্ভিস অ্যাকাউন্টের নিরাপত্তা একত্রিত হয়ে কাজ করে, এবং একাধিক নিরাপত্তা স্তর নিশ্চিত হয়।

5. এফিশিয়েন্সি এবং নির্ভুলতা **(Efficiency and Accuracy):**

- ম্যানুয়াল পাসওয়ার্ড বা **permissions** পরিবর্তন করা ভুল হতে পারে বা ভুল অ্যাকাউন্টে এপ্রুভাল দেওয়া হতে পারে, যা নিরাপত্তা ঝুঁকি তৈরি করে।

- স্বয়ংক্রিয় প্রক্রিয়ায় এরকম ভুলের সম্ভাবনা কমে যায়, এবং সিস্টেম একে অপরের সাথে সিঙ্ক্রোনাইজড থাকে।

6. Authentication and Audit:

- সার্ভিস অ্যাকাউন্টের পাসওয়ার্ড ম্যানেজমেন্ট এবং **permissions** পরিবর্তন হওয়া সবই সাধারণত **audit logs** দ্বারা ট্র্যাক করা হয়।

- এর ফলে যেকোনো সমস্যা বা নিরাপত্তা বিপর্যয় ঘটলে সেগুলি দ্রুত সনাক্ত এবং সমাধান করা যায়।

==========

অবশ্যই, আমি **NTLM** সম্পর্কে বিস্তারিত বলছি**:**

NTLM (NT LAN Manager) হল মাইক্রোসফটের তৈরি একটি পুরনো প্রমাণীকরণ পদ্ধতি। এটি মূলত উইন্ডোজ অপারেটিং সিস্টেমে ব্যবহৃত হত। **NTLM** ব্যবহার করে, একজন ব্যবহারকারী নেটওয়ার্কে অবস্থিত কোনো রিসোর্সে **(**যেমন **-** ফাইল সার্ভার, প্রিন্টার**)** অ্যাক্সেস করতে পারতেন।

NTLM কিভাবে কাজ করে**:**

১**.** যখন একজন ব্যবহারকারী কোনো রিসোর্সে অ্যাক্সেস করার চেষ্টা করেন, তখন তার কম্পিউটার একটি অনুরোধ পাঠায় সার্ভারের কাছে।

২**.** সার্ভার তখন ব্যবহারকারীর পরিচয় **(username)** জানতে চায়।

৩**.** ব্যবহারকারী তার **username** এবং **password** প্রদান করেন।

৪**.** কম্পিউটার **password-**এর একটি হ্যাশ তৈরি করে এবং সেটি সার্ভারে পাঠায়।

৫**.** সার্ভার তার কাছে থাকা ব্যবহারকারীর **password-**এর হ্যাশের সাথে কম্পিউটারের পাঠানো হ্যাশ মিলিয়ে দেখে। যদি মিলে যায়, তাহলে ব্যবহারকারীকে রিসোর্সে অ্যাক্সেস করার অনুমতি দেওয়া হয়।

এখানে হ্যাশ **(hash)** হল একটি বিশেষ কোড, যা **password** থেকে তৈরি করা হয়। হ্যাশ তৈরি করার প্রধান উদ্দেশ্য হল **password**-কে সুরক্ষিত রাখা। এমনকি যদি কেউ হ্যাশ জেনেও যায়, তবুও সে আসল **password** জানতে পারবে না।

NTLM-এর কিছু দুর্বলতা**:**

- পুরনো পদ্ধতি**: NTLM** একটি পুরনো প্রমাণীকরণ পদ্ধতি, যা বর্তমানে দুর্বল বলে মনে করা হয়।
- হ্যাকিংয়ের ঝুঁকি**: NTLM-**এ কিছু দুর্বলতা থাকার কারণে হ্যাকারদের পক্ষে **password** হ্যাশ চুরি করা সম্ভব।
- ফিশিং আক্রমণ**:** হ্যাকাররা বিভিন্ন উপায়ে ব্যবহারকারীদের কাছ থেকে তাদের **NTLM** হ্যাশ হাতিয়ে নিতে পারে, যা দিয়ে তারা নেটওয়ার্কে অননুমোদিত অ্যাক্সেস পেতে পারে।

এজন্য বর্তমানে **NTLM-**এর পরিবর্তে আরও আধুনিক এবং নিরাপদ প্রমাণীকরণ পদ্ধতি, যেমন **-** কেরবেরোস ব্যবহার করার পরামর্শ দেওয়া হয়। কেরবেরোস অনেক বেশি শক্তিশালী এবং এটি হ্যাকিংয়ের ঝুঁকি অনেক কমায়।

তবে **,** পুরনো সিস্টেমগুলোতে এখনও **NTLM** ব্যবহার করা হয়।

NTLM হ্যাশ কোথায় থাকে, সেটি একটি জটিল প্রশ্ন, কারণ এর উত্তর নির্ভর করে আপনি কোন ধরনের উইন্ডোজ সিস্টেম ব্যবহার করছেন তার উপর।

এখানে কিছু সম্ভাব্য স্থান উল্লেখ করা হলো**:**

১**.** লোকাল কম্পিউটারের **SAM** ফাইল**:**

যদি আপনি কোনো ডোমেইন নেটওয়ার্কে যুক্ত না থাকেন, তাহলে আপনার কম্পিউটারের ব্যবহারকারীর পাসওয়ার্ডের হ্যাশ লোকাল সিকিউরিটি অ্যাকাউন্ট ম্যানেজার **(SAM)** ফাইলে সংরক্ষিত থাকে। এই ফাইলটি সাধারণত `C:\Windows\System32\config` ফোল্ডারে থাকে। তবে, এই ফাইলটি অপারেটিং সিস্টেম চালু থাকা অবস্থায় অ্যাক্সেস করা সম্ভব নয়।

২**.** ডোমেইন কন্ট্রোলারের **NTDS.dit** ফাইল**:**

যদি আপনি কোনো ডোমেইন নেটওয়ার্কে যুক্ত থাকেন, তাহলে আপনার ব্যবহারকারীর পাসওয়ার্ডের হ্যাশ ডোমেইন কন্ট্রো লা রে র **NTDS.dit** ফাইলে সংরক্ষিত থাকে। এই ফাইলটি সাধারণত `C:\Windows\NTDS` ফোল্ডারে থাকে। তবে **,** এই ফাইলটিও অপারেটিং সিস্টেম চালু থাকা অবস্থায় অ্যাক্সেস করা সম্ভব নয়।

৩**.** রেজিস্ট্রি**:**

কিছু ক্ষেত্রে, **NTLM** হ্যাশ রেজিস্ট্রিতেও সংরক্ষিত থাকতে পারে। তবে, এটি সাধারণত খুব সুরক্ষিত জায়গায় থাকে এবং সহজে অ্যাক্সেস করা যায় না।

৪**. LSA** সি ক্রে টস**:**

উইন্ডোজ কিছু সংবেদনশীল তথ্য, যেমন - ক্যাশড ডোমেইন ক্রেডেনশিয়াল, লোকাল সিকিউরিটি অথরিটি **(LSA)** সিক্রেটসের মধ্যে সংরক্ষণ করে। এই সিক্রেটসগুলো রেজিস্ট্রির `HKEY_LOCAL_MACHINE\SECURITY\Policy\Secrets` -এর অধীনে থাকে।

গুরুত্বপূর্ণ বিষয়:

- **NTLM** হ্যাশ একটি অত্যন্ত সংবেদনশীল তথ্য, এবং এটি সুরক্ষিত রাখা খুবই জরুরি।
- এই ফাইলগুলো অ্যাক্সেস করার জন্য সাধারণত বিশেষ অনুমতির প্রয়োজন হয়।
- যদি আপনি এই ফাইলগুলোর সাথে পরিচিত না হন, তাহলে এগুলো পরিবর্তন করার চেষ্টা করা উচিত নয়।

স্বয়ংক্রিয় মেমরি ডাম্প (Automatic Memory Dump) হলো

একটি প্রক্রিয়া, যেখানে কোনো কম্পিউটার সিস্টেম অপ্রত্যাশিতভাবে ক্র্যাশ করলে বা বন্ধ হয়ে গেলে, তা র মে মরি র **(RAM)** সমস্ত ডেটা স্বয়ংক্রিয়ভাবে একটি ফাইলে সংরক্ষণ করা হয়। এই ফাইলটিকে মেমরি ডাম্প ফাইল বলা হয়।

সহজ ভাষায়, যখন আপনার কম্পিউটার হঠাৎ করে হ্যাং হয়ে যায় বা নীল স্ক্রিন **(Blue Screen of Death - BSOD)** দেখায় এবং রিস্টার্ট হয়, তখন এর মেমরিতে থাকা সমস্ত তথ্য একটি ফাইলের মধ্যে লিখে রাখা হয়। এই ফাইলটিই হলো মেমরি ডাম্প ফাইল।

এই মেমরি ডাম্প ফাইলটি সাধারণত সিস্টেম অ্যাডমিনিস্ট্রেটর বা সফটওয়্যার ডেভেলপারদের জন্য খুবই গুরুত্বপূর্ণ। তারা এই ফাইলটি বিশ্লেষণ করে কম্পিউটারের ক্র্যাশ হওয়ার কারণ জানতে পারেন এবং সমস্যা সমাধান করতে পারেন। মেমরি ডাম্প ফাইলে কম্পিউটারের মেমরির সমস্ত ডেটা থাকে, যার মধ্যে চলমান প্রোগ্রাম, ড্রাইভার, এবং সিস্টেমের অন্যান্য গুরুত্বপূর্ণ তথ্য অন্তর্ভুক্ত থাকে। এই তথ্যগুলো বিশ্লেষণ করে ক্র্যাশের সময় কম্পিউটারের অবস্থা সম্পর্কে বিস্তারিত জানা যায়।

উইন্ডোজ অপারেটিং সিস্টেমে, স্বয়ংক্রিয় মেমরি ডাম্প সাধারণত ডিফল্টভাবে সক্রিয় থাকে। তবে, ব্যবহারকারীরা চাইলে এটি সেটিংস থেকে পরিবর্তন করতে পারেন। মেমরি ডাম্প ফাইলগুলো সাধারণত `%SystemRoot%\MEMORY.DMP` এই ফোল্ডারে সংরক্ষিত থাকে।

স্বয়ংক্রিয় মেমরি ডাম্প একটি গুরুত্বপূর্ণ বৈশিষ্ট্য, যা কম্পিউটার সিস্টেমের ত্রুটি নির্ণয় এবং সমস্যা সমাধানে সহায়ক।

স্বয়ংক্রিয় মেমরি ডাম্প ফাইল থেকে পিসি হঠাৎ বন্ধ হয়ে যাওয়ার আগের কাজগুলো সরাসরি পুনরুদ্ধার করা সম্ভব নয়। মেমরি ডাম্প ফাইলের প্রধান কাজ হল কম্পিউটার কেন ক্র্যাশ করেছে বা বন্ধ হয়ে গেছে, সেই কারণটি খুঁজে বের করা।

মেমরি ডাম্প ফাইলে কি থাকে:

- **RAM**-এর সমস্ত ডে টা : এর মধ্যে থাকে চলমান প্রোগ্রাম, ড্রাইভার, এবং সিস্টেমের অন্যান্য তথ্য।
- সিস্টেমের অবস্থা: ক্র্যাশের সময় কম্পিউটারের কী অবস্থায় ছিল, সেই সম্পর্কিত তথ্য।

যা করা যায়:

- কারণ নির্ণয়: মেমরি ডাম্প ফাইল বিশ্লেষণ করে কম্পিউটার ক্র্যাশ করার কারণ খুঁজে বের করা যায়। এটি সিস্টেম অ্যাডমিনিস্ট্রেটর বা সফটওয়্যার ডেভেলপারদের জন্য খুবই উপযোগী। তারা এই তথ্য ব্যবহার করে সমস্যার সমাধান করতে পারে।
- কিছুটা তথ্য পুনরুদ্ধার: কিছু ক্ষেত্রে, যদি আপনি কোনো অ্যাপ্লিকেশন ব্যবহার করার সময় ক্র্যাশ করেন, তাহলে সেই অ্যাপ্লিকেশনের কিছু ডেটা হয়তো মেমরি ডাম্প ফাইলে থাকতে পারে। তবে, এটি নিশ্চিত নয় এবং সম্পূর্ণ ডেটা পাওয়ার সম্ভাবনা কম।

যা করা যায় না:

- কাজ পুনরুদ্ধার: আপনি যদি কোনো কাজ করছিলেন এবং পিসি হঠাৎ বন্ধ হয়ে যায়, তাহলে সেই কাজটি মেমরি ডাম্প ফাইল থেকে সরাসরি পুনরুদ্ধার করা সম্ভব নয়।
- সম্পূর্ণ ডেটা পুনরুদ্ধার: মেমরি ডাম্প ফাইলে থাকা ডেটা সাধারণত প্রোগ্রাম বা ফাইলের একটি স্ন্যাপশট। এটি সম্পূর্ণ ফাইল নয়। তাই, সম্পূর্ণ ডেটা পুনরুদ্ধার করা সম্ভব নয়।

কাজ হারানোর হাত থেকে বাঁচানোর উপায়:

- অটোসেভ ব্যবহার করুন: অনেক প্রোগ্রামে অটোসেভ অপশন থাকে। এটি ব্যবহার করলে আপনার কাজ কিছুক্ষণ পর পর স্বয়ংক্রিয়ভাবে সেভ হয়ে যায়। ফলে, পিসি বন্ধ হয়ে গেলেও আপনি আপনার কাজের একটি ব্যাকআপ পেয়ে যাবেন।
- নিয়মিত সেভ করুন: আপনার কাজ নিয়মিত সেভ করার অভ্যাস করুন।
- **UPS** ব্যবহার করুন: যদি আপনার বাড়িতে প্রায়ই বিদ্যুৎ চলে যায়, তা হলে একটি **UPS (Uninterruptible Power Supply)** ব্যবহার করতে পারেন। এটি পিসিকে কিছু সময়ের জন্য চালু রাখতে সাহায্য করে, যাতে আপনি আপনার কাজ সেভ করতে পারেন।

সংক্ষেপে, মেমরি ডাম্প ফাইল ক্র্যাশের কারণ জানতে সাহায্য করে, কিন্তু এটি কাজ পুনরুদ্ধারের জন্য তৈরি করা হয়নি। কাজ হারানোর হাত থেকে বাঁচানোর জন্য উপরে দেওয়া পদ্ধতিগুলো অনুসরণ করতে পারেন।