CST8202 – Windows Desktop Support

Lab 5 – Users, Groups, Shares and Security: Comprehensive Guide

---

Table of Contents

Introduction {#introduction}

Lab Purpose

This lab teaches you how to manage Windows security through user accounts, groups, file system permissions (NTFS), and network shares (SMB). You'll learn both GUI and PowerShell methods for managing access control.

Key Concepts

User Accounts: Individual identities that allow people to log in and access resources

Groups: Collections of users that simplify permission management. Instead of setting permissions for each user individually, you assign permissions to groups.

NTFS Permissions: Control access to files and folders on the local file system. These permissions apply whether accessing locally or over the network.

Share Permissions (SMB): Control access to folders when accessed over the network. These work in combination with NTFS permissions.

ACL (Access Control List): The complete set of permissions assigned to a file or folder

Principle of Least Privilege: Users should have only the minimum permissions needed to perform their job

Permission Types

NTFS Permissions:

- **Full Control:** Complete control including changing permissions
- **Modify:** Read, write, delete files and folders
- **Read & Execute:** View and run files
- **List Folder Contents:** View folder contents
- **Read:** View files and properties
- **Write:** Create new files and write data

Share Permissions:

- **Full Control:** Complete access over the network
- **Change:** Modify, add, and delete files
- **Read:** View files only

Important: When accessing files over the network, the MOST RESTRICTIVE permission between Share and NTFS applies.

Lab Setup {#setup}

Prerequisites

- Windows 11 Virtual Machine
- Administrator access
- VMware Workstation
- Take a snapshot before beginning!

Creating a Lab Snapshot

```
1. VM → Snapshot → Take Snapshot
2. Name: "Before Lab 5"
3. This allows recovery if something goes wrong
```

Important Notes

- Replace "abcd1234" with YOUR actual college username throughout this lab
- All passwords in this lab use: `P@ssW0rd`
- Keep PowerShell running as Administrator throughout the lab

---

Section 1: GUI Operations {#section1}

In this section, you'll create users, groups, and shares using the Windows graphical interface.

Step 30: Enable Local Administrator Account

Objective: Activate the built-in Administrator account

Background: Windows has a built-in Administrator account that's disabled by default for security. Sometimes it's needed for recovery or troubleshooting.

PowerShell Method (Recommended):

```powershell
Enable-LocalUser -Name "Administrator"
```

Alternative PowerShell:

```powershell
Get-LocalUser -Name "Administrator" | Enable-LocalUser
```

GUI Method:

1. Computer Management → Local Users and Groups → Users
2. Right-click **Administrator**
3. Click **Properties**
4. Uncheck "Account is disabled"
5. Click OK

Command Prompt Method:

```cmd
net user administrator /active:yes
```

Verify:

```powershell
Get-LocalUser -Name "Administrator" | Select-Object Name, Enabled
```

Expected Output:

```
Name          Enabled
----          -------
Administrator True
```

Security Note:

- In production, keep Administrator disabled unless needed
- Use regular admin accounts with UAC instead
- If enabled, set a strong password
- Disable after use

Step 31: Add User1a to Administrators Group

Objective: Promote User1a to IT department manager by adding to Administrators group

Background: The "Administrators" group is a pre-built Windows group with complete system control.

PowerShell Method:

```powershell
Add-LocalGroupMember -Group "Administrators" -Member "User1a"
```

GUI Method:

1. Computer Management → Local Users and Groups → Groups
2. Double-click **Administrators**
3. Click **Add**
4. Type: `User1a`
5. Click **Check Names**
6. Click **OK**
7. Click **OK**

Verify:

```powershell
Get-LocalGroupMember -Group "Administrators"
```

Expected Output: Should show User1a in the list along with your account and Administrator.

Important Notes:

- User1a now has full administrative privileges
- Still a member of Management group (users can be in multiple groups)
- Should sign out/in for changes to take full effect
- In production, document all administrative account changes

Step 32: Remove User2b (Terminated Employee)

Objective: Delete User2b's account from the system

Background: When employees leave, their accounts should be removed to prevent unauthorized access.

PowerShell Method:

```powershell
Remove-LocalUser -Name "User2b"
```

GUI Method:

1. Computer Management → Local Users and Groups → Users
2. Right-click **User2b**
3. Click **Delete**
4. Click **Yes** to confirm

Verify:

```powershell
Get-LocalUser -Name "User2b"
```

Expected Output: Should return an error: "Get-LocalUser : User User2b was not found"

Best Practices for Terminated Employees:

1. **Disable First:** Disable account immediately (prevents login)
2. **Review Access:** Check what files/folders they accessed
3. **Transfer Data:** Move their important files to manager
4. **Wait Period:** Keep disabled account for 30-90 days
5. **Delete:** After waiting period, permanently delete

Alternative - Disable Instead of Delete:

```powershell
Disable-LocalUser -Name "User2b"
```

This is often better because:

- Can be re-enabled if needed
- Preserves audit trail
- Maintains file ownership information
- Can recover if deletion was a mistake

Step 33: Disable User2c (Contract Ended)

Objective: Temporarily disable User2c who will return later

Background: For temporary absences (contract breaks, leave of absence, etc.), disable rather than delete accounts.

PowerShell Method:

```powershell
Disable-LocalUser -Name "User2c"
```

GUI Method:

1. Computer Management → Local Users and Groups → Users
2. Right-click **User2c**
3. Click **Properties**
4. Check "Account is disabled"
5. Click **OK**

Verify:

```powershell
Get-LocalUser -Name "User2c" | Select-Object Name, Enabled
```

Expected Output:

```
Name   Enabled
----   -------
User2c False
```

Test:

1. Try to log in as User2c
2. Should get: "Your account has been disabled. Please see your system administrator."

Advantages of Disabling:

- Account remains in system
- All group memberships preserved
- File permissions and ownership intact
- Quick to re-enable when user returns
- Audit trail maintained

To Re-enable Later:

```powershell
Enable-LocalUser -Name "User2c"
```

Step 34: Deny User1d Access to Management Directory

Objective: Prevent User1d from accessing Management folder for policy violation

Background: Even though User1d is in the Management group, we need to explicitly deny access.

PowerShell Method:

```powershell
$acl = Get-Acl -Path "C:\Management"
$denyRule = New-Object System.Security.AccessControl.FileSystemAccessRule("User1d","FullControl","ContainerInl
$acl.AddAccessRule($denyRule)
Set-Acl -Path "C:\Management" -AclObject $acl
```

GUI Method:

1. Right-click **Management** folder → Properties
2. Security tab → Advanced
3. Click **Add**
4. Click **Select a principal**
5. Type: `User1d`
6. Click **Check Names** → OK
7. In "Type" dropdown, select: **Deny**
8. Check: **Full Control**
9. Click **OK**
10. Click **OK** → **OK**

Understanding the Result:

- User1d is still in Management group
- Management group has Allow permissions
- BUT User1d has explicit Deny
- **Deny wins** - User1d cannot access the folder
- Other Management members can still access

Verify:

```powershell
Get-Acl -Path "C:\Management" | Select-Object -ExpandProperty Access | Where-Object IdentityReference -like "*U
```

Test:

1. Log in as User1d
2. Try to open C:\Management
3. Should get "Access Denied"

When to Use Deny:

- Exception to group permission
- Explicit security requirement
- Temporary restriction
- Compliance requirement

Step 35: Map Z: Drive for User2d (%8)

Objective: Configure User2d's profile to automatically map Z: drive to Accounting Share at login

Background: Mapped drives make network shares appear as local drives, making them easier for users to access.

Method 1: PowerShell (Create Persistent Mapping)

First, log in as User2d, then run:

```powershell
New-PSDrive -Name "Z" -PSProvider FileSystem -Root "\\localhost\Accounting Share" -Persist
```

Method 2: Group Policy (More Professional)

1. Press `Win + R`
2. Type: `gpedit.msc` → OK
3. Navigate to: User Configuration → Preferences → Windows Settings → Drive Maps
4. Right-click → New → Mapped Drive
5. Location: `\\localhost\Accounting Share`
6. Drive Letter: Z:
7. Reconnect: Checked
8. Show this drive: All drives
9. OK

Method 3: Login Script

Create a batch file:

```batch
net use Z: "\\localhost\Accounting Share" /persistent:yes
```

Method 4: Registry (Advanced)

For User2d only:

```powershell
# Must run as User2d or modify HKEY_USERS
$path = "HKCU:\Network\Z"
New-Item -Path $path -Force
Set-ItemProperty -Path $path -Name "RemotePath" -Value "\\localhost\Accounting Share"
Set-ItemProperty -Path $path -Name "UserName" -Value ""
```

Recommended Method for This Lab: Command Line (as User2d)

1. Log in as User2d
2. Open Command Prompt
3. Run:

```cmd
net use Z: "\\localhost\Accounting Share" /persistent:yes
```

Verify:

1. Open File Explorer
2. Should see Z: drive under "This PC"
3. Click Z: drive
4. Should show Accounting Share contents

Take Screenshot (%8):

1. Open File Explorer
2. Click on "This PC" in the left pane
3. Ensure Z: drive is visible
4. Make sure the address bar or title shows the mapping
5. Press `Win + Shift + S` to capture
6. Save as: `Lab5_Screenshot8_MappedDrive.png`

Lab Report Entry:

```
%8: [Insert screenshot showing Z: drive mapped to Accounting Share]
```

Understanding Persistent Mappings:

- `/persistent:yes` makes the mapping survive reboots
- User credentials are stored securely
- Mapping reconnects automatically at login
- Can be removed with: `net use Z: /delete`

Section 4: Prepare Final Deliverables {#section4}

This section creates a comprehensive report of all your work using PowerShell.

Step 36: Ensure Logged in as College User

Procedure:

1. If logged in as another user, sign out
2. Log in with your college username (e.g., abcd1234)
3. Launch PowerShell as Administrator

Why:

- Need admin rights to run all commands
- Output should show your perspective as administrator
- Ensures consistent results

Step 37: Navigate to Documents Folder

Command:

```powershell
cd $env:HOMEPATH\Documents
```

Or more explicitly:

```powershell
Set-Location "$env:HOMEPATH\Documents"
```

Verify Your Location:

```powershell
Get-Location
```

Should Show:

```
Path
----
C:\Users\YourUsername\Documents
```

Step 38: Understand $env:HOMEPATH (#9)

Command to Explore:

```powershell
Get-Item env:\
```

This lists all environment variables.

Question: What does the $env:HOMEPATH do?

Explore Specifically:

```powershell
$env:HOMEPATH
$env:USERPROFILE
$env:USERNAME
```

Lab Report Entry:

#9: What does $env:HOMEPATH do?

$env:HOMEPATH is an environment variable that contains the path to the current user's home directory relative to the user profile root.

Specifically:
- It typically returns: \Users\Username
- When combined with the drive letter, it points to C:\Users\Username
- $env:USERPROFILE gives the complete path (C:\Users\Username)
- $env:HOMEPATH gives just the relative path (\Users\Username)

Environment variables are system-wide or user-specific settings that store useful information like:
- User directories
- System paths
- Installed program locations
- Temporary folder locations

They make scripts portable - instead of hardcoding "C:\Users\abcd1234", we use $env:HOMEPATH which works regardless of the actual username.

Examples:
- $env:USERNAME - Current logged-in user
- $env:COMPUTERNAME - Computer name
- $env:TEMP - Temporary files folder
- $env:PROGRAMFILES - Program Files directory
- $env:SYSTEMROOT - Windows installation folder (C:\Windows)

Step 39: Examine Output of Key Commands

Objective: Run each command and review the output before appending to file

Commands to Run (one at a time):

a. List All Local Users:

```powershell
Get-LocalUser
```

What to Look For:

- All User1 and User2 accounts
- User2b should be missing (deleted)
- User2c should show Enabled = False (disabled)
- Administrator should show Enabled = True

---

b. List All Local Groups:

```powershell
Get-LocalGroup
```

What to Look For:

- Management group
- Accounting group
- Built-in groups (Administrators, Users, etc.)

---

c. List Administrators Group Members:

```powershell
Get-LocalGroupMember -Group Administrators
```

What to Look For:

- Your college account
- User1a (promoted to admin)
- Built-in Administrator account

---

d. List Management Group Members:

```powershell
Get-LocalGroupMember -Group Management
```

What to Look For:

- User1a, User1b, User1c, User1d
- All should be listed

---

e. List Accounting Group Members:

```powershell
Get-LocalGroupMember -Group Accounting
```

What to Look For:

- User2a, User2c, User2d
- User2b should be missing (deleted)

---

f. List All Network Shares:

```powershell
Get-SmbShare
```

What to Look For:

- Management Share
- Accounting Share
- Default shares (C, $ADMIN$, IPC$)

---

g. Show Accounting Share Permissions:

```powershell
Get-SmbShareAccess -Name "accounting share"
```

What to Look For:

- Accounting: Full Control
- Management: Read
- Possibly Everyone: Read (default)

---

h. Show Management Share Permissions:

```powershell
Get-SmbShareAccess -Name "management share"
```

What to Look For:

- Management: Change
- Your account: Full Control

---

i. Show Management Directory ACL:

```powershell
Get-Acl -Path "C:\Management" | Format-Table -Wrap
```

Note: The lab instructions show incomplete path. You need to add the full path!

What to Look For:

- Your account: Full Control
- Management: Modify (or specific permissions)
- Accounting: Deny Full Control
- User1d: Deny Full Control

---

j. Show Accounting Directory ACL:

```powershell
Get-Acl -Path "C:\Accounting" | Format-Table -Wrap
```

What to Look For:

- Accounting: Modify or Full Control
- Management: Read & Execute
- Your account: Full Control

Step 40: Redirect All Output to lab5.txt

Objective: Run each command again but append output to a file

Important Notes:

- Use `>>` (append) not `>` (overwrite)
- Each command adds to the same file
- Creates a complete report

Commands (Run Each in Order):

```powershell
# Command a
Get-LocalUser >> lab5.txt

# Command b
Get-LocalGroup >> lab5.txt

# Command c
Get-LocalGroupMember -Group Administrators >> lab5.txt

# Command d
Get-LocalGroupMember -Group Management >> lab5.txt

# Command e
Get-LocalGroupMember -Group Accounting >> lab5.txt

# Command f
Get-SmbShare >> lab5.txt

# Command g
Get-SmbShareAccess -Name "accounting share" >> lab5.txt

# Command h
Get-SmbShareAccess -Name "management share" >> lab5.txt

# Command i
Get-Acl -Path "C:\Management" | Format-Table -Wrap >> lab5.txt

# Command j
Get-Acl -Path "C:\Accounting" | Format-Table -Wrap >> lab5.txt
```

Alternative - All at Once:

```powershell
Get-LocalUser >> lab5.txt
Get-LocalGroup >> lab5.txt
Get-LocalGroupMember -Group Administrators >> lab5.txt
Get-LocalGroupMember -Group Management >> lab5.txt
Get-LocalGroupMember -Group Accounting >> lab5.txt
Get-SmbShare >> lab5.txt
Get-SmbShareAccess -Name "accounting share" >> lab5.txt
Get-SmbShareAccess -Name "management share" >> lab5.txt
Get-Acl -Path "C:\Management" | Format-Table -Wrap >> lab5.txt
Get-Acl -Path "C:\Accounting" | Format-Table -Wrap >> lab5.txt
```

Verify File Creation:

```powershell
Test-Path lab5.txt
Get-Item lab5.txt | Select-Object Name, Length
```

Step 41: Verify lab5.txt Contents

Objective: Check that the file contains all expected output

Command:

```powershell
Get-Content lab5.txt
```

Or page through it:

```powershell
Get-Content lab5.txt | More
```

Or open in Notepad:

```powershell
notepad lab5.txt
```

What to Verify:

- ✅ File exists
- ✅ Contains output from all 10 commands
- ✅ Shows all users, groups, shares
- ✅ No error messages
- ✅ Formatting is readable

If Something is Missing:

- Re-run the specific command with `>>`
- Make sure you're in the Documents directory
- Check for typos in share names

Step 42: Append lab5.txt to Lab Report (#10)

Objective: Include the complete contents in your final submission

Procedure:

1. Open lab5.txt in Notepad
2. Select All (Ctrl+A)
3. Copy (Ctrl+C)
4. Open your lab report document
5. Paste at the appropriate location

Or use PowerShell to read it:

```powershell
Get-Content lab5.txt | Out-String
```

Lab Report Entry:

```
#10: Contents of lab5.txt

[Paste entire contents here]

The file should contain:
- All local user accounts
- All local groups
- Administrators group membership
- Management group membership
- Accounting group membership
- All SMB shares
- Accounting Share permissions
- Management Share permissions
- Management directory ACL
- Accounting directory ACL
```

Complete Deliverables Checklist {#submission}

Screenshots Required (%)

✅ **%1:** Screenshot of all local users (Step 12)

- Must show: User1a, User1b, User1c, User1d, and your account
- From Computer Management → Users

✅ **%2:** Screenshot of Management group members (Step 13)

- Must show: All four User1 accounts
- From Management group properties

✅ **%3:** Screenshot of Management Directory NTFS permissions (Step 14)

- Must show: Your account (Full Control), Management (Modify)
- From folder properties → Security tab

✅ **%8:** Screenshot of Z: drive mapping (Step 35)

- Must show: Z: drive connected to Accounting Share
- From File Explorer, logged in as User2d

Written Answers Required (#)

✅ **#4:** Share permissions on Management Share and reasoning (Step 15)

- Explain: Management (Change), Your account (Full Control)
- Why: Principle of least privilege, defense in depth

✅ **#5:** PowerShell command to copy ACL (Step 22)

```
Get-Acl -Path "C:\Management" | Set-Acl -Path "C:\Accounting"
```

✅ **#6:** PowerShell command to create Accounting Share (Step 24)

```
New-SmbShare -Name "Accounting Share" -Path "C:\Accounting" -FullAccess "Accounting"
```

✅ **#7:** Why accounting user has no access to Accounting Share (Step 25)

- Explain: Share permissions vs NTFS permissions
- Most restrictive wins
- Accounting has Share permissions but no NTFS permissions

✅ **#9:** What is $env:HOMEPATH (Step 38)

- Explain: Environment variable
- Contains user home directory path
- Makes scripts portable

✅ **#10:** Complete contents of lab5.txt (Step 42)

- Paste entire file output
- Should contain all 10 command outputs

Troubleshooting Common Issues

User Cannot Log In

Problem: User account won't accept password

Solutions:

- Verify password is exactly: `P@ssW0rd` (case-sensitive)
- Check "User must change password" is UNCHECKED
- Verify account is not disabled
- Ensure account is in "Users" group

PowerShell Check:

```powershell
Get-LocalUser -Name "Username" | Select-Object Name, Enabled, PasswordRequired
```

Access Denied on Network Share

Problem: User cannot access share they should have access to

Diagnosis Steps:

1. **Check Share Permissions:**

```powershell
Get-SmbShareAccess -Name "ShareName"
```

2. **Check NTFS Permissions:**

```powershell
Get-Acl -Path "C:\FolderPath" | Format-List
```

3. **Check Group Membership:**

```powershell
Get-LocalGroupMember -Group "GroupName"
```

4. **Check for Deny Permissions:**

```powershell
Get-Acl -Path "C:\FolderPath" | Select-Object -ExpandProperty Access | Where-Object AccessControlType -eq "Den
```

Common Causes:

- User not in correct group
- Share permissions missing
- NTFS permissions too restrictive
- Explicit Deny permission
- Network discovery disabled

Share Not Visible on Network

Problem: Cannot see share when browsing network

Solutions:

1. **Check Share Exists:**

```powershell
Get-SmbShare -Name "ShareName"
```

2. **Enable Network Discovery:**

```powershell
# Enable in Windows Firewall
Set-NetFirewallRule -DisplayGroup "Network Discovery" -Enabled True
```

3. **Access Directly with UNC Path:**

```
\\ComputerName\ShareName
or
\\localhost\ShareName
```

4. **Check Sharing Settings:**

- Settings → Network & Internet → Advanced sharing settings
- Turn on network discovery
- Turn on file and printer sharing

PowerShell Access Denied

Problem: PowerShell commands fail with access denied

Solutions:

- Ensure PowerShell is running as Administrator
- Right-click PowerShell → Run as Administrator
- Check UAC settings

Cannot Remove Permissions

Problem: Unable to modify or remove permissions on folder

Solutions:

1. **Take Ownership:**

```powershell
takeown /F "C:\FolderPath" /R /D Y
```

2. **Reset Permissions:**

```powershell
icacls "C:\FolderPath" /reset /T
```

3. **Via GUI:**

- Advanced Security Settings → Owner tab
- Change owner to Administrators
- Apply → Try removing permissions again

Key Concepts Review

NTFS vs Share Permissions

Aspect	NTFS Permissions	Share Permissions
Where Applied	Local and network	Network only
Granularity	Very detailed	Basic (Read, Change, Full)
Scope	Files and folders	Folders only
When Accessed Locally	Applied	Not applied
When Accessed Over Network	Applied	Applied
Most Common Use	Primary security	Additional network control

Effective Permissions Formula:

- Local Access: NTFS only
- Network Access: Most restrictive of (NTFS AND Share)

Permission Inheritance

How It Works:

- Child objects inherit permissions from parent
- Can be disabled per object
- Changes to parent flow to children
- Explicit permissions override inherited

Best Practices:

- Set permissions at highest level possible
- Disable inheritance only when necessary
- Document non-inherited permissions
- Review inheritance chain for troubleshooting

Group Strategy

Benefits of Groups:

- Manage permissions for many users at once
- Add/remove users without changing folder permissions
- Easier to audit
- Follows principle of role-based access

Best Practices:

- Create groups based on job function
- Use descriptive group names
- Document group purposes
- Regular review of membership
- Remove users promptly when roles change

Security Principles

Principle of Least Privilege:

- Users get minimum permissions needed
- Reduces security risks
- Limits damage from compromised accounts

Defense in Depth:

- Multiple security layers
- Both Share and NTFS permissions
- User authentication
- Group membership
- Network security

Deny vs Not Allowing:

- Prefer "not granting" over explicit Deny
- Deny is harder to troubleshoot
- Deny overrides all Allow permissions
- Use Deny only for specific exceptions

Advanced Topics

Using AGDLP Strategy

A-G-DL-P (Account-Global-Domain Local-Permission):

For Domain environments:

1. **A**ccounts → added to
2. **G**lobal groups → added to
3. **D**omain **L**ocal groups → assigned
4. **P**ermissions

For Workgroups (this lab):

- Accounts → Local Groups → Permissions

ACL Structure

Components:

- **Owner:** Who owns the object
- **DACL (Discretionary ACL):** Permissions list
- **SACL (System ACL):** Audit settings
- **ACE (Access Control Entry):** Individual permission entry

ACE Contains:

- Security principal (user/group)
- Permission type (Allow/Deny)
- Permissions (Read, Write, etc.)
- Inheritance flags

PowerShell Permission Management

Get Permissions:

```powershell
Get-Acl -Path "C:\Folder" | Format-List
```

Detailed View:

```powershell
(Get-Acl -Path "C:\Folder").Access | Format-Table IdentityReference, FileSystemRights, AccessControlType
```

Copy Permissions:

```powershell
Get-Acl -Path "C:\Source" | Set-Acl -Path "C:\Destination"
```

Add Permission:

```powershell
$acl = Get-Acl -Path "C:\Folder"
$permission = "UserName", "FullControl", "ContainerInherit,ObjectInherit", "None", "Allow"
$accessRule = New-Object System.Security.AccessControl.FileSystemAccessRule $permission
$acl.SetAccessRule($accessRule)
Set-Acl -Path "C:\Folder" -AclObject $acl
```

Best Practices Summary

User Management

- ✅ Use strong passwords in production
- ✅ Enable "User must change password at next logon"
- ✅ Set password expiration policies
- ✅ Disable (don't delete) when users leave
- ✅ Remove from sensitive groups immediately
- ✅ Document all admin accounts

Group Management

- ✅ Use groups instead of individual permissions
- ✅ Name groups descriptively
- ✅ Add descriptions to groups
- ✅ Review membership regularly
- ✅ Limit administrators group membership
- ✅ Create groups based on job roles

Permission Management

- ✅ Follow principle of least privilege
- ✅ Use NTFS permissions as primary security
- ✅ Set Share permissions to match NTFS
- ✅ Document permission structures
- ✅ Avoid Deny unless necessary
- ✅ Test permissions after changes
- ✅ Review permissions quarterly

Share Management

- ✅ Use descriptive share names
- ✅ Document share purposes
- ✅ Hide administrative shares (end with $)
- ✅ Disable default shares if not needed
- ✅ Monitor share access logs
- ✅ Use encryption for sensitive data

Additional Resources

PowerShell Commands Reference

User Management:

```powershell
New-LocalUser
Get-LocalUser
Set-LocalUser
Remove-LocalUser
Enable-LocalUser
Disable-LocalUser
Rename-LocalUser
```

Group Management:

```powershell
New-LocalGroup
Get-LocalGroup
Set-LocalGroup
Remove-LocalGroup
Add-LocalGroupMember
Get-LocalGroupMember
Remove-LocalGroupMember
```

Share Management:

```powershell
New-SmbShare
Get-SmbShare
Set-SmbShare
Remove-SmbShare
Grant-SmbShareAccess
Get-SmbShareAccess
Revoke-SmbShareAccess
Block-SmbShareAccess
Unblock-SmbShareAccess
```

Permission Management:

```powershell
Get-Acl
Set-Acl
Get-Permission # Third-party
Set-Permission # Third-party
```

GUI Tools Reference

Computer Management (compmgmt.msc):

- Manage users and groups
- View system information
- Manage services
- View event logs

Local Security Policy (secpol.msc):

- Password policies
- Account lockout policies
- User rights assignment
- Security options

Group Policy Editor (gpedit.msc):

- Not available in Windows Home
- Configure computer and user policies
- Map network drives
- Deploy software

Advanced Sharing Settings:

- Network discovery
- File and printer sharing
- Public folder sharing
- Password protected sharing

Conclusion

This lab covered essential Windows security and networking concepts:

Skills Learned:

- ✅ Creating and managing user accounts (GUI and PowerShell)
- ✅ Creating and managing security groups
- ✅ Understanding and configuring NTFS permissions
- ✅ Creating and securing network shares
- ✅ Configuring Share (SMB) permissions
- ✅ Understanding effective permissions
- ✅ Troubleshooting access issues
- ✅ Using Deny permissions appropriately
- ✅ Mapping network drives
- ✅ Managing user lifecycle (enable/disable/delete)
- ✅ Copying and modifying ACLs

Real-World Applications:

- Department folder structure with appropriate security
- User onboarding and offboarding procedures
- Shared resource management
- Access control and audit requirements
- IT security policies implementation

Next Steps:

- Practice in Active Directory environments
- Learn Group Policy management
- Study Windows Server file services
- Explore advanced NTFS features (encryption, compression)
- Learn audit policy configuration

Remember: **Security is layered**. Always use multiple security mechanisms (authentication, permissions, groups, auditing) to protect resources.

Document Version: 1.0
Last Updated: September 29, 2025
Course: CST8202 – Windows Desktop Support
Lab: Lab 5 – Users, Groups, Shares and Security1: Create Four Users

Objective: Create user accounts User1a, User1b, User1c, and User1d

Method 1: Computer Management (Recommended)

1. Right-click Start → **Computer Management**
2. Navigate to: **System Tools** → **Local Users and Groups** → **Users**
3. Right-click in the users pane → **New User**

For Each User (User1a, User1b, User1c, User1d):

4. Fill in the form:
   - **User name:** User1a (change for each)
   - **Full name:** User1a (or leave blank)
   - **Description:** (leave blank or add "Management Team")
   - **Password:** `P@ssW0rd`
   - **Confirm password:** `P@ssW0rd`
5. **Uncheck:** "User must change password at next logon"
6. **Check:** "Password never expires" (for lab purposes only)
7. Click **Create**
8. Repeat for User1b, User1c, and User1d

Method 2: Advanced Users Settings

1. Press `Win + R`
2. Type: `netplwiz` → Click OK
3. Click **Add**
4. Click "Sign in without a Microsoft account"
5. Click "Local account"
6. Enter username and password
7. Repeat for all users

Why These Settings:

- In production, users SHOULD change passwords at first logon
- Passwords SHOULD expire regularly
- We're using simplified settings for lab practice only

Verification: You should now see User1a, User1b, User1c, and User1d in the user list.

Step 2: Create Management Group

Objective: Create a local group called "Management"

Procedure:

1. In **Computer Management**
2. Navigate to: **System Tools** → **Local Users and Groups** → **Groups**
3. Right-click in the groups pane → **New Group**
4. **Group name:** Management
5. **Description:** (Leave blank for now - we'll add this next)
6. Click **Create**
7. Click **Close**

Understanding Groups:

- Groups simplify permission management
- You can add/remove users from groups without changing folder permissions
- Users can belong to multiple groups
- Permissions from all groups are cumulative (combined)

Step 3: Add Description to Management Group

Objective: Add a meaningful description to the group

Procedure:

1. In **Computer Management** → **Groups**
2. Double-click the **Management** group
3. In the **Description** field, type: `Management team members`
4. Click **OK**

Why Add Descriptions:

- Helps administrators understand group purposes
- Important in large organizations with many groups
- Good documentation practice

Step 4: Add All User1 Accounts to Management Group

Objective: Add User1a, User1b, User1c, and User1d as members

Procedure:

1. Double-click the **Management** group
2. Click **Add**
3. Click **Advanced**
4. Click **Find Now**
5. Hold **Ctrl** and click: User1a, User1b, User1c, User1d
6. Click **OK**
7. Click **OK** again
8. Click **OK** to close the group properties

Alternative Method (Adding One at a Time):

1. Click **Add**
2. Type: `User1a`
3. Click **Check Names** (verifies user exists)
4. Click **OK**
5. Repeat for each user

Verification: The Management group properties should now show all four User1 accounts as members.

Step 5: Create Management Directory

Objective: Create a folder called "Management" that we'll secure and share

Procedure:

1. Open **File Explorer**
2. Navigate to *C:* (or any drive you prefer)
3. Right-click in empty space → **New** → **Folder**
4. Name it: `Management`
5. Press **Enter**

Alternative Location: You could create this on:

- C:\Shares\Management
- P:\ (the drive you created in Lab 4)
- Any other drive

Best Practice: In production environments:

- Store shared folders on a separate drive/partition
- Keep them organized in a dedicated Shares folder
- Use consistent naming conventions

Step 6: Set NTFS Permissions on Management Directory

Objective: Configure security so Management group members can Modify files, while other users cannot access it. Your college account should have Full Control.

Understanding the Goal:

- **Management Group:** Modify access
- **Your Account (abcd1234):** Full Control
- **Everyone Else:** No Access

Procedure:

1. Right-click the **Management** folder → **Properties**
2. Click the **Security** tab
3. Click **Advanced**
4. Click **Disable inheritance**
5. Choose **"Remove all inherited permissions from this object"**
   - This removes default permissions and gives us a clean slate
6. Click **Add** to add permissions

Add Your College Account (Full Control):

1. Click **Select a principal**
2. Type your username (e.g., `abcd1234`)
3. Click **Check Names**
4. Click **OK**
5. Check **Full Control**
6. Click **OK**

Add Management Group (Modify):

1. Click **Add**
2. Click **Select a principal**
3. Type: `Management`
4. Click **Check Names**
5. Click **OK**
6. Check the following permissions:
   - ✅ Modify
   - ✅ Read & Execute
   - ✅ List folder contents
   - ✅ Read
   - ✅ Write
7. Click **OK**
8. Click **OK** to close Advanced Security Settings
9. Click **OK** to close folder Properties

Understanding Permissions:

Permission	What It Allows
Full Control	Everything including changing permissions and taking ownership
Modify	Create, read, write, delete files and folders
Read & Execute	Open and run files, view folder contents
List Folder Contents	See what's in a folder (folders only)
Read	Open and view files, view properties
Write	Create new files and folders, write data

Why We Disabled Inheritance:

- By default, folders inherit permissions from parent folders
- This can include "Users" group having Read access
- Disabling inheritance gives us complete control
- We only want specific users/groups to access this folder

Step 7: Share the Management Directory

Objective: Make the folder available over the network with the name "Management Share"

Procedure:

1. Right-click the **Management** folder → **Properties**
2. Click the **Sharing** tab
3. Click **Advanced Sharing**
4. Check **"Share this folder"**
5. In **Share name:** type `Management Share`
6. Click **Permissions**

(We'll set share permissions in the next step)

Understanding Shares:

- Shares make folders accessible over the network
- Accessed via UNC paths: `\\ComputerName\ShareName`
- Share names can be different from folder names
- Share names with $ at the end are hidden (e.g., C, $Admin$)

Step 8: Set Share Permissions

Objective: Configure network access permissions for the share

In the Permissions dialog (still open from Step 7):

Remove Everyone:

1. Select **Everyone**
2. Click **Remove**

Add Management Group:

1. Click **Add**
2. Type: `Management`
3. Click **Check Names**
4. Click **OK**
5. With Management selected, check **Change** under Allow
6. Click **OK**

Add Your College Account:

1. Click **Add**
2. Type your username (e.g., `abcd1234`)
3. Click **Check Names**
4. Click **OK**
5. With your account selected, check **Full Control** under Allow
6. Click **OK**
7. Click **OK** to close Advanced Sharing
8. Click **Close** to close folder Properties

Understanding Share Permissions:

Share Permission	What It Allows Over Network
Full Control	Complete network access
Change	Read, write, modify files over network
Read	View files only over network

Critical Concept - Effective Permissions: When accessing over the network, Windows applies BOTH:

1. Share permissions
2. NTFS permissions

The most restrictive permission wins.

Example:

- Share permissions: Change
- NTFS permissions: Read
- **Effective permission: Read**

Step 9: Verify Your Work

Objective: Ensure all settings are correct before proceeding

Check User Creation:

1. Computer Management → Users
2. Verify: User1a, User1b, User1c, User1d exist

Check Group Membership:

1. Computer Management → Groups
2. Double-click **Management**
3. Verify all User1 accounts are members

Check NTFS Permissions:

1. Right-click Management folder → Properties → Security
2. Verify:
   - Your account: Full Control
   - Management: Modify
   - No other users/groups listed

Check Share:

1. Right-click Management folder → Properties → Sharing
2. Verify share name is "Management Share"
3. Click Advanced Sharing → Permissions
4. Verify:
   - Management: Change
   - Your account: Full Control
   - Everyone: REMOVED

Step 10: Test Login as Management User

Objective: Verify that User1a can log in successfully

Procedure:

1. Click **Start**
2. Click your user icon
3. Click **Sign out** (or **Switch user**)
4. Select **Other user**
5. Username: `User1a`
6. Password: `P@ssW0rd`
7. Press **Enter**

What to Verify:

- Login succeeds without errors
- Desktop loads properly
- User can access basic functions

If Login Fails:

- Verify password is exactly: `P@ssW0rd` (capital P, @ symbol, capital W, zero)
- Verify "User must change password" is unchecked
- Check that account isn't disabled

Step 11: Switch Back to Your Account

Procedure:

1. Click **Start**
2. Click the User1a user icon
3. Click **Sign out** (or **Switch user**)
4. Log in with your college account (abcd1234)

Why We're Switching Back:

- Need administrator privileges for remaining tasks
- User1a has limited permissions
- Your college account has administrative access

Step 12: Screenshot User List (%1)

Objective: Capture proof of user creation for lab report

Procedure:

1. Open **Computer Management**
2. Navigate to: **Local Users and Groups** → **Users**
3. Make sure all users are visible in the list
4. Press **Windows** + **Shift** + **S** (Snipping Tool)
    - Or use **Win** + **PrtScn** for full screenshot
5. Capture the window showing: User1a, User1b, User1c, User1d
6. Save as: `Lab5_Screenshot1_Users.png`

What Should Be Visible:

- Computer Management title bar
- Local Users and Groups → Users in navigation
- User list showing User1a, User1b, User1c, User1d
- Your college username

Lab Report Entry:

```
%1: [Insert screenshot of user list here]
```

Step 13: Screenshot Management Group Members (%2)

Objective: Document group membership

Procedure:

1. In **Computer Management** → **Groups**
2. Double-click **Management** group
3. Ensure the Members list is visible
4. Take screenshot showing:
   - "Management Properties" window title
   - Description field
   - All four User1 accounts listed as members
5. Save as: `Lab5_Screenshot2_ManagementGroup.png`

Lab Report Entry:

```
%2: [Insert screenshot of Management group members here]
```

Step 14: Screenshot Management Directory Permissions (%3)

Objective: Document NTFS security settings

Procedure:

1. Right-click **Management** folder → **Properties**
2. Click **Security** tab
3. Make sure both groups/users are visible:
   - Your college account (Full Control)
   - Management group (Modify)
4. Take screenshot
5. Save as: `Lab5_Screenshot3_NTFSPermissions.png`

What Should Be Visible:

- Folder path in title bar
- Security tab selected
- Group/user names listed
- Permission checkboxes showing Modify for Management

Lab Report Entry:

```
%3: [Insert screenshot of NTFS permissions here]
```

Step 15: Document Share Permissions (#4)

Objective: Explain the share permissions and reasoning

To Check Share Permissions:

1. Right-click Management folder → Properties
2. Sharing tab → Advanced Sharing → Permissions

Question: What are the Share permissions on the share called Management Share? Why did you set the permissions this way?

Lab Report Entry:

#4:
Share Permissions on "Management Share":
- Management group: Change
- [Your username]: Full Control

Reasoning:
I set these permissions to follow the principle of least privilege. The Management group needs Change permissions to create, modify, and delete files over the network for their daily work. My administrator account needs Full Control for administrative tasks, troubleshooting, and managing the share. I removed the default "Everyone" group to prevent unauthorized network access.

The combination of Share and NTFS permissions provides defense in depth - even if someone bypasses one security layer, the other still protects the data.

Section 2: PowerShell Operations {#section2}

In this section, you'll accomplish similar tasks using PowerShell commands instead of the GUI. This is faster and can be scripted for automation.

Important: Launch PowerShell as Administrator for all commands in this section!

Step 16: Create Four Users with PowerShell

Objective: Create User2a, User2b, User2c, and User2d using cmdlets

Understanding the Command:

```powershell
New-LocalUser -Name "Username" -Password (ConvertTo-SecureString "Password" -AsPlainText -Force) -Password
```

Breaking It Down:

- `New-LocalUser` - Creates local user account
- `-Name "Username"` - Specifies the username
- `-Password (ConvertTo-SecureString...)` - Creates encrypted password
  - `ConvertTo-SecureString` - Converts plain text to secure string
  - `-AsPlainText` - Indicates we're providing plain text
  - `-Force` - Bypasses security warning
- `-PasswordNeverExpires` - Password doesn't expire
- `-UserMayNotChangePassword` - User cannot change their password

Complete Commands (Run Each Separately):

```powershell
New-LocalUser -Name "User2a" -Password (ConvertTo-SecureString "P@ssW0rd" -AsPlainText -Force) -PasswordN
```

```powershell
New-LocalUser -Name "User2b" -Password (ConvertTo-SecureString "P@ssW0rd" -AsPlainText -Force) -PasswordN
```

```powershell
New-LocalUser -Name "User2c" -Password (ConvertTo-SecureString "P@ssW0rd" -AsPlainText -Force) -PasswordN
```

```powershell
New-LocalUser -Name "User2d" -Password (ConvertTo-SecureString "P@ssW0rd" -AsPlainText -Force) -PasswordN
```

Advanced Alternative (Creates All Four at Once):

```powershell
$password = ConvertTo-SecureString "P@ssW0rd" -AsPlainText -Force
"User2a","User2b","User2c","User2d" | ForEach-Object {
    New-LocalUser -Name $_ -Password $password -PasswordNeverExpires -UserMayNotChangePassword
}
```

Expected Output:

```
Name   Enabled Description
----   ------- -----------
User2a True
User2b True
User2c True
User2d True
```

Verify Creation:

```powershell
Get-LocalUser | Where-Object Name -like "User2*"
```

Step 17: Create Accounting Group

Objective: Create a local group using PowerShell

Command:

```powershell
New-LocalGroup -Name "Accounting"
```

Breaking Down the Command:

- `New-LocalGroup` - Creates a new local group
- `-Name "Accounting"` - Name of the group

Expected Output:

```
Name       Description
----       -----------
Accounting
```

Verify:

```powershell
Get-LocalGroup -Name "Accounting"
```

Step 18: Add Description to Accounting Group

Objective: Add a meaningful description using PowerShell

Command:

```powershell
Set-LocalGroup -Name "Accounting" -Description "Accounting department members"
```

Breaking Down the Command:

- `Set-LocalGroup` - Modifies an existing group
- `-Name "Accounting"` - Specifies which group
- `-Description "..."` - Sets the description text

Verify:

```powershell
Get-LocalGroup -Name "Accounting" | Format-List Name, Description
```

Expected Output:

```
Name        : Accounting
Description : Accounting department members
```

Step 19: Add User2 Accounts to Accounting and Users Groups

Objective: Add all User2 accounts to two groups: Accounting and Users

Commands (Run Each):

Add to Accounting Group:

```powershell
Add-LocalGroupMember -Group "Accounting" -Member "User2a","User2b","User2c","User2d"
```

Add to Users Group:

```powershell
Add-LocalGroupMember -Group "Users" -Member "User2a","User2b","User2c","User2d"
```

Breaking Down the Commands:

- `Add-LocalGroupMember` - Adds users to a group
- `-Group "Accounting"` - Target group
- `-Member "User2a","User2b"...` - Comma-separated list of users

Why Add to Users Group:

- The built-in "Users" group provides basic permissions
- Allows users to log in and access basic functions
- Without this, User2 accounts might have login issues

Verify Accounting Group:

```powershell
Get-LocalGroupMember -Group "Accounting"
```

Expected Output:

```
ObjectClass Name                PrincipalSource
----------- ----                ---------------
User        COMPUTERNAME\User2a     Local
User        COMPUTERNAME\User2b     Local
User        COMPUTERNAME\User2c     Local
User        COMPUTERNAME\User2d     Local
```

Verify Users Group:

```powershell
Get-LocalGroupMember -Group "Users" | Where-Object Name -like "*User2*"
```

Step 20: Create Accounting Directory

Objective: Create a folder for the Accounting team

Command:

```powershell
New-Item -Path "C:\Accounting" -ItemType Directory
```

Alternative Locations:

```powershell
# On the P: drive from Lab 4
New-Item -Path "P:\Accounting" -ItemType Directory

# In a Shares folder
New-Item -Path "C:\Shares\Accounting" -ItemType Directory
```

Breaking Down the Command:

- `New-Item` - Creates new items (files, folders, etc.)
- `-Path "C:\Accounting"` - Where to create it
- `-ItemType Directory` - Specifies we're creating a folder

Expected Output:

```
    Directory: C:\

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d----        9/29/2025   3:00 PM                Accounting
```

Verify:

```powershell
Test-Path "C:\Accounting"
```

Should return: `True`

Step 21 & 22: Copy ACL from Management to Accounting Directory

Objective: Copy the security permissions from Management folder to Accounting folder

This is your answer for #5 in the lab report!

Command:

```powershell
Get-Acl -Path "C:\Management" | Set-Acl -Path "C:\Accounting"
```

Breaking Down the Command:

- `Get-Acl -Path "C:\Management"` - Gets the Access Control List from Management folder
- `|` - Pipeline operator (sends output to next command)
- `Set-Acl -Path "C:\Accounting"` - Applies that ACL to Accounting folder

Understanding ACL:

- ACL = Access Control List
- Contains all permissions for a file/folder
- Includes: users, groups, permissions, inheritance settings
- Copying ACL is faster than manually recreating permissions

Why This is Useful:

- Ensures consistent permissions across similar folders
- Much faster than manually setting permissions
- Reduces errors from manual configuration
- Essential for scripting and automation

Verify the Copy:

```powershell
Get-Acl -Path "C:\Accounting" | Format-List
```

Expected Output: Should show the same permissions as Management folder:

- Your college account: Full Control
- Management group: Modify

Important Note: After copying the ACL, the Accounting folder will still have Management group permissions. We'll need to modify this later (Step 28).

Lab Report Entry:

```
#5: Get-Acl -Path "C:\Management" | Set-Acl -Path "C:\Accounting"
```

Step 23 & 24: Share the Accounting Directory

Objective: Create an SMB share with Full Control for Accounting group

This is your answer for #6 in the lab report!

Command:

```powershell
New-SmbShare -Name "Accounting Share" -Path "C:\Accounting" -FullAccess "Accounting"
```

Breaking Down the Command:

- `New-SmbShare` - Creates a new network share
- `-Name "Accounting Share"` - The share name (how it appears on network)
- `-Path "C:\Accounting"` - The local folder to share
- `-FullAccess "Accounting"` - Gives Accounting group Full Control share permissions

Alternative with Multiple Permissions:

```powershell
New-SmbShare -Name "Accounting Share" -Path "C:\Accounting" -FullAccess "Accounting","YourUsername" -Read
```

Expected Output:

```
Name            ScopeName Path         Description
----            --------- ----         -----------
Accounting Share  *       C:\Accounting
```

Verify the Share:

```powershell
Get-SmbShare -Name "Accounting Share"
```

Check Share Permissions:

```powershell
Get-SmbShareAccess -Name "Accounting Share"
```

Expected Output:

```
Name            ScopeName AccountName      AccessControlType AccessRight
----            --------- -----------      ----------------- -----------
Accounting Share  *       Everyone         Allow             Read
Accounting Share  *       Accounting       Allow             Full
```

Note: "Everyone" with Read is added by default. We may need to remove this for security.

Lab Report Entry:

```
#6: New-SmbShare -Name "Accounting Share" -Path "C:\Accounting" -FullAccess "Accounting"
```

Step 25: Test Access as Accounting User (#7)

Objective: Discover why an accounting user cannot access their own share

Procedure:

1. **Log out and log in as User2a:**
   - Start → User Icon → Sign out
   - Log in with:
     - Username: `User2a`
     - Password: `P@ssW0rd`
2. **Try to access the share:**
   - Press `Win + R`
   - Type: `\\localhost\Accounting Share`
   - Press Enter

OR

- Open File Explorer
- In the address bar type: `\\localhost\Accounting Share`
- Press Enter

3. **Result:** You should get an **Access Denied** error

Question: Why do you not have access to this share?

Lab Report Entry:

#7: The accounting user cannot access the Accounting Share because of a permission mismatch.

Here's what happened:
1. We copied the ACL from the Management directory to Accounting directory
2. This means the NTFS permissions show "Management" group has Modify access
3. The Accounting group has NO NTFS permissions on the folder
4. Even though Accounting has Full Control SHARE permissions, Windows applies the most restrictive of Share AND NTFS permissions
5. Since Accounting has no NTFS permissions, the effective permission is "No Access"

To fix this, we need to either:
- Add Accounting group to the NTFS permissions, OR
- Change the NTFS permissions from Management to Accounting, OR
- Add Accounting users to the Management group (not ideal)

This demonstrates why both Share and NTFS permissions must be configured correctly for network access to work.

Understanding the Problem:

Permission Layer	**Accounting Group Permission**
Share Permissions	Full Control ✅
NTFS Permissions	None ❌
Effective Permission	**No Access**

Remember: Most Restrictive Wins!

Section 3: Troubleshooting and Modifications {#section3}

Now you'll fix the issues and make additional security modifications using either GUI or PowerShell (your choice).

Step 26: Troubleshoot Accounting Share Access

Objective: Fix the permissions so Accounting users can access their share

Problem Identified: Accounting group needs NTFS permissions on the Accounting folder

Solution Using PowerShell:

```powershell
$acl = Get-Acl -Path "C:\Accounting"
$permission = "Accounting","Modify","ContainerInherit,ObjectInherit","None","Allow"
$accessRule = New-Object System.Security.AccessControl.FileSystemAccessRule $permission
$acl.SetAccessRule($accessRule)
Set-Acl -Path "C:\Accounting" -AclObject $acl
```

Solution Using GUI:

1. Right-click **Accounting** folder → Properties
2. Security tab → Edit
3. Add → Type "Accounting" → Check Names → OK
4. Select Accounting
5. Check: Modify, Read & Execute, List folder contents, Read, Write
6. OK → OK

Verify:

1. Log back in as User2a
2. Try accessing: `\\localhost\Accounting Share`
3. Should now work!

Step 27: Give Management Users Read Access on Accounting Share

Objective: Allow Management team to view Accounting files over the network

PowerShell Method:

```powershell
Grant-SmbShareAccess -Name "Accounting Share" -AccountName "Management" -AccessRight Read -Force
```

GUI Method:

1. Right-click Accounting folder → Properties
2. Sharing tab → Advanced Sharing
3. Permissions → Add
4. Type "Management" → Check Names → OK
5. With Management selected, check "Read" under Allow
6. OK → OK → Close

Verify:

```powershell
Get-SmbShareAccess -Name "Accounting Share"
```

Should show:

- Accounting: Full Control
- Management: Read

Step 28: Remove Write Permissions from Management on Accounting Directory

Objective: Management can read but not modify files in Accounting folder

PowerShell Method (Recommended):

```powershell
$acl = Get-Acl -Path "C:\Accounting"
$accessRule = New-Object System.Security.AccessControl.FileSystemAccessRule("Management","Modify","Contain
$acl.RemoveAccessRule($accessRule)
$readRule = New-Object System.Security.AccessControl.FileSystemAccessRule("Management","ReadAndExecute","
$acl.AddAccessRule($readRule)
Set-Acl -Path "C:\Accounting" -AclObject $acl
```

GUI Method:

1. Right-click Accounting folder → Properties → Security
2. Click Advanced
3. Select the Management entry
4. Click Edit
5. Uncheck: Modify, Write
6. Ensure checked: Read & Execute, List folder contents, Read
7. OK → OK → OK

Verify:

```powershell
Get-Acl -Path "C:\Accounting" | Select-Object -ExpandProperty Access | Where-Object IdentityReference -like "*Ma
```

Step 29: Deny Accounting Users Access to Management Directory

Objective: Prevent Accounting team from viewing Management folder

Important: Deny permissions override all Allow permissions!

PowerShell Method:

```powershell
$acl = Get-Acl -Path "C:\Management"
$denyRule = New-Object System.Security.AccessControl.FileSystemAccessRule("Accounting","FullControl","Contain
$acl.AddAccessRule($denyRule)
Set-Acl -Path "C:\Management" -AclObject $acl
```

GUI Method:

1. Right-click Management folder → Properties
2. Security tab → Advanced
3. Add
4. Select a principal → Type "Accounting" → Check Names → OK
5. Type: Deny
6. Permissions: Full Control
7. OK → OK → OK

Understanding Deny:

- Deny ALWAYS wins over Allow
- Use sparingly - can cause confusion
- Often better to simply not grant permission than to explicitly deny
- Deny is inherited by subfolders

Verify:

1. Log in as User2a
2. Try to open C:\Management
3. Should get "Access Denied"

Step