SOP
Using Access Permissions and Rights to Secure a Folder

v1.0.1

Author:
Eldon Gabriel

March 20, 2025

Security Systems Specialist

Table of Contents

Revision History

Version	Date	Author	Description of Changes
1.0.0	08/28/2025	Eldon G.	Initial draft.
1.0.1	03/20/2025	Eldon G.	Aligned folder security protocols with NIST 800-53 (AC-6); implemented PowerShell execution restrictions via GPO; added Executive Summary and verified via multi-role testing.

0.0 Executive Summary

0.1 Project Overview

Secure Access Control & Folder Hardening

This report outlines the process for establishing detailed access controls and enhancing security within a Windows system. The project stresses the persistent use of the **Principle of Least Privilege (POLP)** to safeguard sensitive data and limit the use of administrative tools that are often exploited in lateral movement attacks.

Project Objective

The primary objective was to ensure safe data storage. This was achieved by setting up New Technology File System (NTFS) permissions and local security groups. This ensures that only those with access to private information are allowed. Additionally, the project involved hardening the host's security posture by utilizing the **Local Group Policy** to restrict **PowerShell** access, thereby reducing the attack surface available to unauthorized users or automated malware.

Technical Specifications

The implementation utilizes the following core security mechanisms.

- **Identity Management:** Created and managed local user accounts and security groups to facilitate role-based access control.
- **Filesystem Security:** Configured explicit **NTFS permissions** on sensitive directories and disabled inheritance to prevent "permission creep" from parent folders.
- **Execution Control: Local Group Policy Objects (GPOs)** are applied to prevent specific user groups from launching PowerShell, thereby mitigating the risk of script-based exploitation.
- **Environment Integrity:** Developed and validated a systematic "Clean Up" protocol to ensure no residual technical debt or unauthorized accounts remained post-implementation.

Validation and Quality Assurance

To confirm the effectiveness of the security boundaries, the following testing protocols were executed:

- **Multi-Role Testing:** Verified access across three distinct personas (Administrator, Authorized User, and Restricted User) to ensure that permissions functioned as intended.
- **Negative Testing:** Unauthorized users were explicitly denied folder access, and PowerShell execution was successfully blocked by the system policy.
- **Audit Verification:** Utilized the `icacls` command to generate a definitive report of the final Access Control List (ACL) for administrative review.

1.0 Securing Folders With Access Permissions

1.1 Project Description

This guide demonstrates how to create and manage local users, groups, and access permissions in Windows. The objective is to secure a confidential folder so that only authorized groups can access it. Additionally, **PowerShell** will be restricted for specific groups using Local Group Policy.

1.2 Show Existing Users and Groups

To display the current users and groups, run the following commands:

```
PowerShell

net user

net localgroup
```

✅ **Visual:** A list of users and groups is displayed (Admin, Finance, HR, and other defaults).

1.3 Create Users and Groups

Create new groups and assign users to them.

```
PowerShell

:: Create groups

net localgroup [Group Name] /add

:: Create a [User Name],[New Password] and add to a [Group Name]

net user user1 [Input New Password] /add

net localgroup [Group Name] [User Name] /add
```

Note: Repeat for multiple users and groups.

To verify:

```
PowerShell

net user

net localgroup
```

1.4 Create Confidential Folder & Apply Permissions

PowerShell

```
:: Create folder
mkdir C:\[Folder Name]

:: Remove inherited permissions
icacls "C:\[Folder Name]" /inheritance:r

:: Remove generic Users
icacls "C:\[Folder Name]" /remove "Users"
icacls "C:\[Folder Name]" /remove "Authenticated Users"

:: Grant access
icacls "C:\[Folder Name]" /grant [Group Name]:(OI)(CI)(F)
icacls "C:\[Folder Name]" /grant Administrators:(OI)(CI)(F)

:: Verify
icacls "C:\[Folder Name]"
```

1.5 Restrict PowerShell Access via Group Policy

1. Open Local Group Policy Editor (`gpedit.msc`)
    - Navigate to:

      User Configuration → Windows Settings → Security Settings → Software Restriction Policies

    - If no policy exists: Right-click **Software Restriction Policies → New Software Restriction Policies**

2. Create path rules to block PowerShell for specific groups

```
PowerShell
%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe

%SystemRoot%\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
```

    - Set Security Level = **Disallowed**
    - Apply to: `[Group Name]`

Note: *Restricting PowerShell via GPO serves as a deterrent for standard users; however, for high-security environments, we recommend combining this with* ***AppLocker*** *or* ***Windows Defender Application Control (WDAC)*** *for robust execution prevention.*

2.0 Testing and Verification

2.1 Test Permissions

As **[User Name]** (Member of **[Group Name]**)

```bash
runas /user:[ComputerName]\[User Name] cmd
cd C:\[Folder Name] → should succeed/fail if granted/blocked
powershell          → should succeed/fail if granted/blocked
```

Note: Repeat for multiple users.

As an Administrator:

```bash
cd C:\[Folder Name] → should succeed
powershell          → should succeed
```

2.2 Clean Up (Optional for VM reset)

```bash
takeown /F C:\[Folder Name] /R /D Y
rmdir /S /Q C:\[Folder Name]
net user [User Name] /delete
net localgroup [Group Name] /delete
```

✅ **Visual:** Folders, users, and groups have been removed.

To verify:

```bash
net user
net localgroup
icacls "C:\[Folder Name]"
```

3.0 Conclusion

3.1 Key Takeaways

- Access permissions can restrict sensitive data to specific groups.
- Group Policy rules prevent the misuse of administrative tools.
- Testing with multiple accounts confirms the effectiveness of security controls.

3.2 Security Implications and Recommendations

- Regularly review user and group memberships to prevent privilege creep.
- The **principle of least privilege (POLP)** is applied when assigning folder access.
- Enforce auditing of sensitive folders using Windows Security Logs to track access attempts.
- Consider combining access controls with centralized management solutions, such as Active Directory Group Policy, for scalability.
- Align permissions and controls with recognized frameworks, such as **NIST 800-53 (AC-6 Least Privilege)** and **ISO 27001 Annex A.9 (Access Control),** for compliance and best practice consistency.

Although this lab utilized local accounts, these principles translate to an Active Directory environment using Domain Groups and Group Policy Objects (GPOs) for centralized scalability.