BitLyft
Cybersecurity

SECURING SMB SIGNING GUIDE

COMPREHENSIVE GUIDE

Securing SMB Signing for On-Premises Windows Devices and Microsoft Cloud

Why SMB Signing Matters Today

SMB (Server Message Block) is a critical protocol for file sharing, printer access, and administrative tasks in Windows environments. Without signing, SMB traffic is vulnerable to interception and tampering, especially in hybrid setups where on-premises devices interact with Microsoft cloud services (e.g., Azure AD, Azure File Shares). As of 2025, Microsoft's Secure Future Initiative has made SMB signing mandatory by default in Windows 11 24H2 and Windows Server 2025, but legacy systems and misconfigurations still pose risks. This guide ensures a secure, consistent SMB signing policy across your environment.

Check first to verify Powershell command to Check:

```
# Function to check SMB Signing settings

function Check-SMBSigning {

$regPath =
"HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters"

$valueName = "RequireSecuritySignature"
}
try {
$smbSigningEnabled = Get-ItemProperty -Path $regPath -Name
$valueName -ErrorAction Stop
if ($smbSigningEnabled.RequireSecuritySignature -eq 1) {
Write-Output "SMB Signing is enabled."
exit 0
} else {
Write-Output "SMB Signing is NOT enabled."
exit 1
}
} catch {
Write-Output "Failed to check SMB Signing status."
exit 1
}
```

This script defines a function Check-SMBSigning (though it's not explicitly called in the code), then uses Get-ItemProperty to check the registry key HKLM:\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters for the RequireSecuritySignature value. If the value is 1, it indicates SMB signing is enabled on the server side; otherwise, it's not. The try/catch block handles errors, such as missing registry keys, and the exit codes (0 for success, 1 for failure) make it suitable for scripting or automation.

STEP 1: Assess Your Enviornment

Before enforcing SMB signing, understand your setup to avoid breaking compatibility with legacy devices or third-party systems.

1. Inventory Devices and Versions:
    - List all Windows devices (servers, workstations) and their OS versions (e.g., Windows Server 2016, Windows 10 21H2).
    - Identify non-Windows SMB clients (e.g., Linux Samba, printers, NAS devices).
    - Check for SMB protocol versions in use (SMB 1.0, 2.x, 3.x) via PowerShell:
    - powershell

```
Get-SmbConnection | Select-Object Dialect, ServerName, ShareName
```

Check Current SMB Signing Status:
On clients:
powershell

```
Get-SmbClientConfiguration | Select-Object RequireSecuritySignature
```

On servers:
powershell

```
Get-SmbServerConfiguration | Select-Object RequireSecuritySignature
```

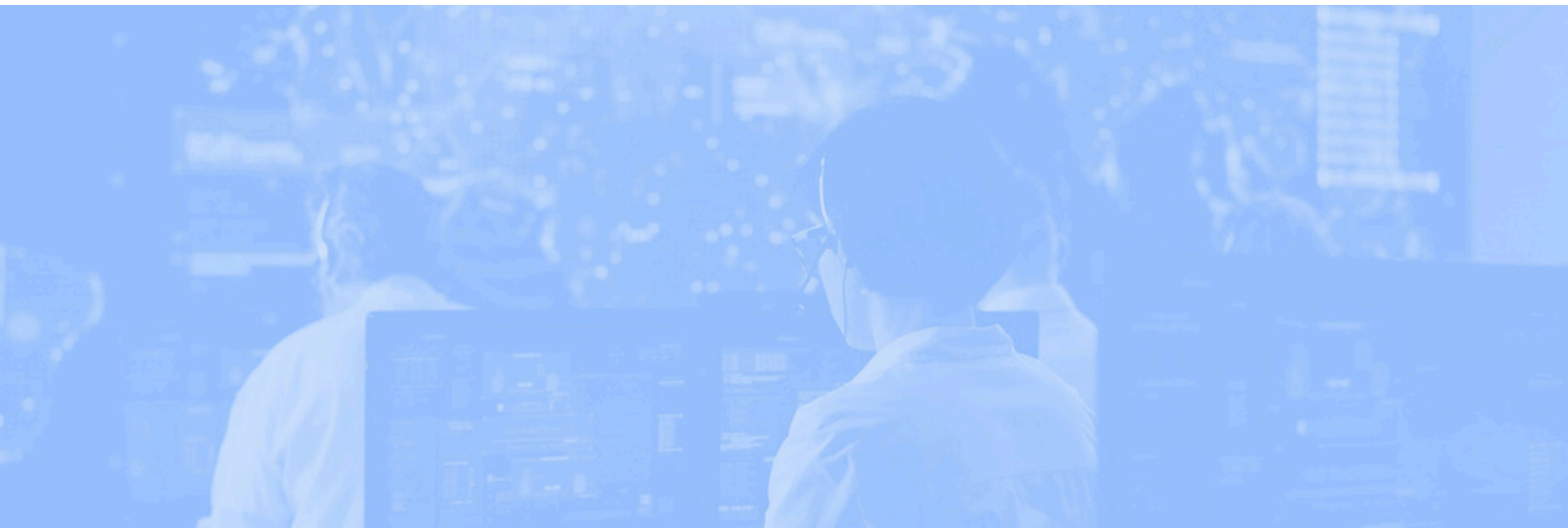
A result of $false indicates signing is not required.

Identify Cloud Integration:

- Note Microsoft cloud services in use (e.g., Azure Files, OneDrive mapped via SMB).
- Azure Files supports SMB 3.0+ with encryption, not just signing, so compatibility is key.

Risk Assessment:

- Legacy devices using SMB 1.0 or unsigned SMB 2.x are prime targets for NTLM relay attacks.
- Unsecured outbound connections to cloud services increase exposure.



STEP 2: Plan Your SMB Signing Strategy

Design a phashed approach to minimize disruption while maximizing security.

Set Goals:

- Require SMB signing on all outbound (client) and inbound (server) connections.
- Disable SMB 1.0 entirely (deprecated and insecure).
- Prefer SMB 3.1.1 for encryption where possible, but ensure signing as a baseline.
-

Compatibility Considerations:

- Windows XP and Server 2003 only support SMB 1.0 (no signing by default). Upgrade or isolate these.
- Windows 7/Server 2008 R2 and later support SMB 2.0+ with signing.
- Non-Windows devices may need firmware updates or Samba configuration (e.g., server signing = mandatory).

Performance Trade-offs:

- Signing adds overhead (CPU and latency), especially on older hardware or high-throughput networks.
- Test impact in a lab environment before deployment.

Cloud Alignment:

- Azure Files requires SMB 3.0+ and supports encryption. Signing alone isn't sufficient for cloud shares; plan for encryption where applicable.

Step 3: Disable SMB 1.0

Check SMB 1.0 Usage:

Run:

powershell

```
Get-WindowsFeature *FS-SMB1* # Servers
Get-WindowsOptionalFeature -Online | Where-Object {$_.FeatureName -like "*SMB1*"} # Clients
```

Disable SMB 1.0:

On servers:

powershell

```
Remove-WindowsFeature FS-SMB1
```

On clients:

powershell

```
Disable-WindowsOptionalFeature -Online -FeatureName SMB1Protocol
```

Reboot devices after disabling.

Verify Removal:

Confirm SMB 1.0 is gone:

powershell

```
Get-SmbServerConfiguration | Select-Object EnableSMB1Protocol
```

STEP 4: Configure SMB Signing via Group Policy (On-Premises)

Use Group Policy to enforce SMB signing across your AD domain.

1. Open Group Policy Management:
   - Launch gpmc.msc from a domain controller or admin workstation.
2. Create or Edit a GPO:
   - Create a new GPO (e.g., "SMB Signing Policy") or edit an existing one.
   - Link it to the appropriate OU (e.g., all servers or workstations).
3. Set Client-Side Signing:
   - Navigate to: Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options.
   - Configure:
     - Microsoft network client: Digitally sign communications (always): Enabled
       - Forces clients to require signing, rejecting unsigned servers.
     - Microsoft network client: Digitally sign communications (if server agrees): Enabled (optional fallback for compatibility).
4. Set Server-Side Signing:
   - Same path as above.
   - Configure:
     - Microsoft network server: Digitally sign communications (always): Enabled
       - Forces servers to require signing from clients.
     - Microsoft network server: Digitally sign communications (if client agrees): Enabled (optional for legacy clients).
5. Apply and Test:
   - Run gpupdate /force on test machines.
   - Verify connectivity to file shares and printers.

STEP 5: SECURE CLOUD INTEGRATION

Align on-premises SMB signing with Microsoft cloud services.

Azure Files Configuration:

- Ensure clients use SMB 3.0+ (Windows 8/Server 2012 or later).
- Enable encryption (preferred over signing alone):

powershell

```
Set-SmbClientConfiguration -RequireSecureNegotiate $true -
RequirePrivacy $true
```

Test connectivity:
powershell

```
New-SmbMapping -RemotePath "\\
<storageaccount>.file.core.windows.net\<share>" -UserName "<AzureUser>"
-Password "<Key>"
```

Monitor Signing with Azure:

Azure Files logs signing and encryption status. Check Azure Monitor for unsigned connection attempts.

STEP 6: Harden Authentication

1. Prefer Kerberos Over NTLM:
   - Ensure all devices are domain-joined and use Kerberos (default in AD).
   - Block NTLM outbound:

powershell

```
Set-SmbClientConfiguration -EnableSecuritySignature $true -
RequireSecuritySignature $true
Set-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\Services\LanmanWorkstation\Parameters"
-Name "RestrictNTLM" -Value 1
```

1. Enable Authentication Rate Limiting:
   - In Windows Server 2025/Windows 11 24H2, enable SMB rate limiter (default on):

powershell

```
Set-SmbServerConfiguration -AuthenticationRateLimit 2
```

STEP 7: Test and Monitor

Validate your configuration and watch for issues.

1. Test Connectivity:
   - Map a share from a client to a server:

powershell

```
New-SmbMapping -LocalPath Z: -RemotePath "\\server\share"
```

Check signing:

powershell

```
Get-SmbConnection | Select-Object Dialect, Signed
```

1. Monitor Event Logs:
   - Look for SMB signing failures in the Event Viewer under Microsoft-Windows-SMBServer/Operational.
2. Audit Third-Party Devices:
   - Test printers, NAS, etc., for signing support. Update firmware or replace non-compliant devices.

STEP 8: Handle Exceptions and Legacy Systems

1. Isolate Legacy Devices:
   - Move unsupported systems (e.g., Windows XP) to a separate VLAN with no internet access.
   - Use firewalls to block SMB traffic from these segments.
2. Temporary Workarounds:
   - If signing breaks critical apps, disable it temporarily on specific servers:
   - powershell

```
Set-SmbServerConfiguration -RequireSecuritySignature $false
```

Plan upgrades ASAP.

STEP 9: Document and Educate

Ensure your setup is maintainable and understood.

1. Document Settings:
    - Record GPOs, PowerShell scripts, and exceptions in a central wiki or IT repository.
1. Train Staff:
    - Educate admins on SMB signing's importance and troubleshooting steps (e.g., Event Log analysis).

Troubleshooting Tips

- Connection Failures: Check if the server or client rejects unsigned traffic. Temporarily set “if agrees” policies to diagnose.
- Performance Issues: Upgrade to SMB 3.1.1 (AES-GMAC signing reduces overhead) or optimize hardware.
- Cloud Errors: Verify SMB 3.0+ and encryption settings for Azure Files.

Securing SMB signing in a hybrid on-premises and Microsoft cloud environment requires careful planning but significantly reduces attack surfaces. By enforcing signing, disabling SMB 1.0, and aligning with cloud requirements, you protect data integrity and authenticity. Regular audits and updates will keep your setup resilient as threats evolve.

WE'RE HERE TO HELP

When you're looking to protect an entire organization on a limited budget you can have trouble prioritizing spending. The one certainty is that you need people. Whether you're an established company, or just getting started, BitLyft has a solution to help you protect your organization from cyber attacks. Our team of cybersecurity professionals can fully augment, or come along your existing team to help you illuminate and eliminate cyber threats. Schedule a meeting with our BitLyft Success Team to get started.

"In over a decade of helping businesses decide to build their own SOC or outsource to a SOCaaS provider, I have seen time and time again a much better ROI to subscribe to the right vendor than to build internally. Everyone's environment is different and the use cases can vary. However, there's never a bad reason to take a moment to review your current state and future needs with experts like those at BitLyft."

securonix

MIKE JOHNSON
CHANNEL SALES DIRECTOR
SECURONIX, INC.