VOKE

PENTEST REMEDIATION PLAYBOOK

Step-by-Step Fixes for Common Penetration Testing Findings

30	5	19
Findings	Categories	High / Critical

Version 1.0 | April 2026

Penetration Testing. No Fillers. Just Findings.

vokecyber.com | info@vokecyber.com | +1-765-422-5464

HOW TO USE THIS PLAYBOOK

This playbook provides step-by-step remediation guidance for the 30 most common findings from internal penetration tests. It is designed for IT administrators, system engineers, and security teams responsible for implementing fixes.

EACH FINDING INCLUDES

RISK RATING
Critical, High, Medium, or Low -- indicates the severity if left unaddressed. Start with Critical and High items.

IMPLEMENTATION EFFORT
Low, Medium, or High -- estimates the time and complexity to remediate. Low-effort, high-risk items give you the most impact for the least work.

WHO NEEDS TO ACT
Each finding specifies the role or access level required (e.g., Domain Admin, GPO Admin, Network Admin).

REMEDIATION STEPS
Exact commands, GPO paths, registry keys, and configurations. Copy-paste ready for PowerShell, Group Policy, and network devices.

GOTCHAS
Real-world warnings about what may break when you implement the fix. Always test in a pilot environment first.

VERIFICATION
Commands to confirm the remediation was applied successfully. Run these after every change.

RECOMMENDED APPROACH

1. Start with the Quick Reference table to prioritize by risk and effort
2. Address Critical and High risk findings with Low effort first
3. Schedule Medium and High effort items during change windows
4. Verify each fix using the provided verification commands
5. Re-test or request a validation scan to confirm remediation

TABLE OF CONTENTS

Network Protocols (6)

Active Directory (12)

Credential Management (4)

Host Security (6)

SSL/TLS and Web (2)

QUICK REFERENCE: REMEDIATION PRIORITY

Start with Low-effort, High/Critical-risk findings for maximum impact. The table below is sorted by risk (descending) then effort (ascending) to help you prioritize your remediation schedule.

#	ID	Finding	Risk	Effort	Requires
1	AD-005	Print Spooler on Domain Controllers	Critical	Low	Domain Admin
2	AD-011	DCSync-Capable Non-Admin Accounts	Critical	Low	Domain Admin
3	AD-003	Unconstrained Delegation	Critical	Medium	Domain Admin
4	HOST-005	Outdated or End-of-Life Operating Systems	Critical	High	IT Management, System Admin
5	NET-001	LLMNR/NBT-NS Poisoning	High	Low	Domain Admin or GPO Admin
6	NET-002	SMB Signing Not Required	High	Low	Domain Admin or GPO Admin
7	AD-002	AS-REP Roastable Accounts	High	Low	Domain Admin
8	AD-007	Weak Domain Password Policy	High	Low	Domain Admin
9	AD-008	GPP Passwords (MS14-025)	High	Low	Domain Admin
10	AD-009	LDAP Signing Not Required	High	Low	Domain Admin
11	CRED-001	Default or Weak Credentials	High	Low	System Admin, Application Owner
12	CRED-003	WDigest Authentication Enabled	High	Low	Domain Admin, GPO Admin
13	NET-003	IPv6 DNS Takeover	High	Medium	Network Admin, Domain Admin
14	AD-001	Kerberoastable Service Accounts	High	Medium	Domain Admin, Service Account Owner
15	AD-004	NTLM Relay Attack Paths	High	Medium	Domain Admin, GPO Admin
16	AD-006	LAPS Not Deployed	High	Medium	Domain Admin, Schema Admin (for LAPS ins..
17	CRED-002	Cleartext Passwords in Network Shares and Scripts	High	Medium	System Admin, Application Owner
18	CRED-004	LSASS Not Protected (Credential Guard / RunAsPPL)	High	Medium	Domain Admin, GPO Admin
19	HOST-001	Local Admin Password Reuse	High	Medium	Domain Admin, GPO Admin
20	NET-004	WPAD Abuse	Medium	Low	DNS Admin, GPO Admin
21	NET-005	SNMP Default Community Strings	Medium	Low	Network Admin
22	AD-010	Excessive Machine Account Quota	Medium	Low	Domain Admin
23	AD-012	Null Session Enumeration	Medium	Low	Domain Admin, GPO Admin
24	HOST-002	RDP Without Network Level Authentication (NLA)	Medium	Low	System Admin, GPO Admin
25	HOST-003	PowerShell v2 Enabled	Medium	Low	System Admin, GPO Admin
26	HOST-004	Windows Firewall Disabled	Medium	Low	GPO Admin
27	NET-006	Unencrypted Protocol Usage (FTP, Telnet, HTTP)	Medium	Medium	System Admin, Application Owner
28	WEB-001	SSL/TLS Weak Configurations	Medium	Medium	System Admin, Application Owner
29	HOST-006	Unrestricted Outbound Internet Access	Medium	High	Network Admin, Firewall Admin
30	WEB-002	Default Web Application Pages and Headers	Low	Low	System Admin, Application Owner

NETWORK PROTOCOLS

NET-001 LLMNR/NBT-NS Poisoning

HIGH | Effort: Low | Requires: Domain Admin or GPO Admin

DESCRIPTION

Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) are fallback name resolution protocols used when DNS fails. Attackers on the same network segment can respond to these broadcast/multicast queries, redirecting authentication attempts to capture NTLMv2 password hashes or relay credentials to other services.

IMPACT

An attacker on the internal network can intercept authentication attempts to capture NTLMv2 hashes for offline password cracking or relay credentials to gain unauthorized access to file shares, email, databases, and other services. This is one of the most commonly exploited findings in internal penetration tests.

REMEDIATION STEPS

Step 1: Disable LLMNR via Group Policy

Create or edit a GPO linked to all workstation and server OUs

```
Computer Configuration > Administrative Templates > Network > DNS Client > Turn off multicast nam...
```

Note: This policy is available on Windows Server 2012 R2 and later ADMX templates

Step 2: Disable NBT-NS via PowerShell

Deploy as a GPO startup script or via SCCM/Intune

```
# Disable NetBIOS over TCP/IP on all network adapters
$adapters = Get-WmiObject Win32_NetworkAdapterConfiguration
$adapters | Where-Object {$_.IPEnabled} | ForEach-Object {
    $_.SetTcpipNetbios(2)  # 0=Default, 1=Enable, 2=Disable
}
```

Alternative: Disable via DHCP vendor option (recommended for DHCP environments)

```
Set-DhcpServerv4OptionValue -ScopeId <scope> -VendorClass 'Microsoft Options' -OptionId 001 -Valu...
```

Note: Setting value 2 disables NetBIOS over TCP/IP

Step 3: Verify remediation

Confirm NBT-NS is disabled on a target host

```
nbtstat -n
```

Confirm LLMNR is disabled

```
Resolve-DnsName -LlmnrOnly testhost  # Should fail with error
```

Network-level verification with Responder (from pentest box)

```
responder -I eth0 -A  # Analyze mode only  --  should see no responses
```

Note: Run verification across multiple network segments

GOTCHAS

! Legacy printers and scanners may rely on NBT-NS for discovery -- inventory these devices before disabling

! WPAD auto-configuration may break if clients relied on LLMNR to resolve 'wpad' -- create a DNS A record for wpad pointing to your proxy
! Some VoIP phones use NetBIOS -- test in a pilot OU before domain-wide rollout
! DHCP option 001 only applies to DHCP clients -- static IP hosts need the GPO startup script

VERIFICATION

Run Responder in analyze mode to confirm no hosts respond to poisoning attempts

```
responder -I eth0 -A
```

Audit via PowerShell across domain computers

```
Get-WmiObject Win32_NetworkAdapterConfiguration -ComputerName $targetHost |
  Where-Object {$_.IPEnabled} |
  Select-Object Description, TcpipNetbiosOptions
```

REFERENCES

https://www.blackhillsinfosec.com/how-to-disable-llmnr-why-you-want-to/
https://attack.mitre.org/techniques/T1557/001/

NET-002 SMB Signing Not Required

HIGH | Effort: Low | Requires: Domain Admin or GPO Admin

DESCRIPTION

SMB (Server Message Block) signing ensures the integrity and authenticity of SMB communications. When SMB signing is not required, an attacker can perform man-in-the-middle attacks to intercept and relay SMB authentication to other hosts, effectively impersonating the victim to access network resources.

IMPACT

Attackers can relay captured NTLM authentication to unsigned SMB hosts to execute commands, access file shares, or move laterally across the network. Combined with LLMNR/NBT-NS poisoning, this provides a reliable path to gaining initial foothold and escalating privileges.

REMEDIATION STEPS

Step 1: Enable SMB Signing via Group Policy

For all domain members -- require signing

```
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options:
  Microsoft network client: Digitally sign communications (always) -> Enabled
  Microsoft network server: Digitally sign communications (always) -> Enabled
```

Note: Apply to all workstation and server OUs. Domain Controllers already require signing by default.

Step 2: Verify via PowerShell

Check local SMB signing configuration

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

Remote check with CrackMapExec (from pentest/audit box)

```
nxc smb 10.0.0.0/24 --gen-relay-list unsigned.txt
```

Note: The unsigned.txt file should be empty after remediation

GOTCHAS

- ! SMB signing adds ~10-15% CPU overhead on file servers -- monitor performance after enabling
- ! Very old devices (Windows XP, legacy NAS) may not support SMB signing -- these must be upgraded or isolated
- ! Some Linux Samba configurations need explicit signing settings in smb.conf: 'server signing = mandatory'
- ! Third-party SMB implementations (printers, scanners, NAS) may lose connectivity -- test first

VERIFICATION

Scan the network and confirm no unsigned hosts remain

```
nxc smb 10.0.0.0/24 --gen-relay-list should-be-empty.txt
```

Verify via registry

```
reg query HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters /v RequireSecuritySignature
# Should return 0x1
```

REFERENCES

https://attack.mitre.org/techniques/T1557/001/
https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/overview-server-message-block-signing

NET-003 IPv6 DNS Takeover

HIGH | Effort: Medium | Requires: Network Admin, Domain Admin

DESCRIPTION

Most Windows environments have IPv6 enabled by default but do not use it for DNS. An attacker can set up a rogue DHCPv6 server to assign themselves as the DNS server for IPv6, allowing them to intercept DNS queries and redirect authentication to capture credentials via WPAD or other protocols.

IMPACT

An attacker can intercept and redirect all DNS queries from hosts that prefer IPv6, enabling credential theft, man-in-the-middle attacks, and NTLM relay attacks. This bypasses many network security controls that only monitor IPv4.

REMEDIATION STEPS

Step 1: Option A: Disable IPv6 via GPO (if not in use)

Disable IPv6 on all interfaces via registry (GPO preference)

```
# GPO Registry Preference:
HKLM\SYSTEM\CurrentControlSet\Services\Tcpip6\Parameters
Value: DisabledComponents
Type: REG_DWORD
Data: 0xFF (Disable all IPv6 components)
```

Note: Microsoft recommends 0x20 (prefer IPv4 over IPv6) rather than full disable. Use 0xFF only if IPv6 is not needed.

Step 2: Option B: Configure legitimate DHCPv6 and IPv6 DNS (if IPv6 is in use)

Deploy DHCPv6 with guard policies on network switches

```
# Cisco IOS example  --  enable DHCPv6 guard
ipv6 dhcp guard policy DHCP_GUARD
  device-role server
  trusted-port
interface GigabitEthernet0/1
  ipv6 dhcp guard attach-policy DHCP_GUARD
```

Note: DHCPv6 guard blocks rogue DHCPv6 servers on untrusted ports

Step 3: Enable RA Guard on switches

Block rogue Router Advertisements (Cisco IOS example)

```
ipv6 nd raguard policy RA_GUARD
  device-role host
interface range GigabitEthernet0/1 - 48
  ipv6 nd raguard attach-policy RA_GUARD
```

GOTCHAS

- ! Fully disabling IPv6 can break Windows features -- DirectAccess, HomeGroup, and some clustering services require it
- ! Microsoft officially does not recommend disabling IPv6 entirely -- prefer setting the IPv4 preference flag (0x20)
- ! Network switch configuration varies by vendor -- consult your specific vendor documentation for RA Guard and DHCPv6 guard

VERIFICATION

Run mitm6 in analyze mode to test for IPv6 takeover susceptibility

```
mitm6 --ignore-nofqdn -d domain.local  # From pentest/audit box
```

Verify registry setting on endpoints

```
reg query HKLM\SYSTEM\CurrentControlSet\Services\Tcpip6\Parameters /v DisabledComponents
```

REFERENCES

https://blog.fox-it.com/2018/01/11/mitm6-compromising-ipv4-networks-via-ipv6/
https://attack.mitre.org/techniques/T1557/003/

NET-004 WPAD Abuse

MEDIUM | Effort: Low | Requires: DNS Admin, GPO Admin

DESCRIPTION

Web Proxy Auto-Discovery (WPAD) allows clients to automatically find a proxy configuration file. If no DNS entry for 'wpad' exists, clients fall back to LLMNR/NBT-NS to resolve it, enabling attackers to serve a malicious proxy configuration that intercepts all HTTP traffic.

IMPACT

An attacker can intercept all HTTP traffic from affected clients, capturing credentials submitted to web applications, session tokens, and sensitive data. When combined with LLMNR poisoning, this attack is highly reliable.

REMEDIATION STEPS

Step 1: Create a DNS entry for WPAD

Add a DNS A record that either points to a real proxy or a black hole

```
# PowerShell  --  create a DNS A record pointing to a non-existent host
Add-DnsServerResourceRecordA -Name "wpad" -ZoneName "domain.local" -IPv4Address "127.0.0.1"
```

Note: If you use a proxy, point this to your real PAC file server. If not, point it to localhost.

Step 2: Disable WPAD auto-detection via GPO

Disable auto-detect proxy settings in Internet Explorer/Edge

```
User Configuration > Preferences > Windows Settings > Registry:
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections
# Or via GPO:
User Configuration > Admin Templates > Windows Components >
Internet Explorer > Disable changing Automatic Configuration settings -> Enabled
```

PowerShell -- disable for current user

```
Set-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet Settings" -Name ...
```

GOTCHAS

- ! Some organizations legitimately use WPAD for proxy configuration -- verify with the network team before disabling
- ! Chrome and Firefox have their own WPAD settings independent of IE/Edge GPO settings
- ! The WinHttpAutoProxySvc service must also be considered -- it runs independently of browser settings

VERIFICATION

Verify DNS resolution of wpad

```
nslookup wpad.domain.local
```

Verify auto-detect is disabled

```
Get-ItemProperty "HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet Settings" |
  Select-Object AutoDetect
```

REFERENCES

https://attack.mitre.org/techniques/T1557/001/

NET-005 SNMP Default Community Strings

MEDIUM | Effort: Low | Requires: Network Admin

DESCRIPTION

Simple Network Management Protocol (SNMP) devices using default community strings (public/private) allow unauthorized reading and potentially writing of device configuration data including network topology, ARP tables, routing information, and device credentials.

IMPACT

Attackers can enumerate complete network topology, discover internal IP ranges and VLANs, extract device configurations containing credentials, and in some cases reconfigure network devices to facilitate further attacks.

REMEDIATION STEPS

Step 1: Change community strings to complex values

Use long, random community strings -- treat them as passwords

```
# Generate a strong community string:
python3 -c "import secrets; print(secrets.token_hex(16))"
```

Note: Never use 'public', 'private', or the organization name as community strings

Step 2: Migrate to SNMPv3 where possible

SNMPv3 supports authentication and encryption

```
# Cisco IOS example
snmp-server group SECUREGRP v3 priv
snmp-server user SNMPUSER SECUREGRP v3 auth sha AuthP@ss123 priv aes 128 PrivP@ss123
no snmp-server community public
no snmp-server community private
```

Step 3: Restrict SNMP access with ACLs

Limit SNMP access to management stations only

```
# Cisco IOS example
access-list 99 permit 10.0.1.50    # NMS server
access-list 99 deny any log
snmp-server community <complex-string> RO 99
```

GOTCHAS

- ! Changing community strings requires updating all NMS/monitoring tools (Nagios, PRTG, SolarWinds, etc.) simultaneously
- ! Some legacy devices only support SNMPv1/v2c -- these should be documented and have compensating controls
- ! SNMP community strings are sent in cleartext over the network -- even non-default strings can be sniffed without SNMPv3

VERIFICATION

Test that default community strings no longer work

```
snmpwalk -v2c -c public <target-ip> 2>&1 | head -5  # Should timeout or return error
```

Scan for remaining SNMP default strings

```
nmap -sU -p 161 --script snmp-brute --script-args snmp-brute.communitiesdb=community.txt 10.0.0.0/24
```

REFERENCES

https://attack.mitre.org/techniques/T1602/

NET-006 Unencrypted Protocol Usage (FTP, Telnet, HTTP)

MEDIUM | Effort: Medium | Requires: System Admin, Application Owner

DESCRIPTION

Cleartext protocols such as FTP (port 21), Telnet (port 23), and HTTP (port 80 for authenticated services) transmit credentials and data without encryption, allowing any attacker on the network to intercept sensitive information via packet capture.

IMPACT

Credentials and sensitive data transmitted over cleartext protocols can be trivially captured by any attacker with network access, using tools as simple as Wireshark or tcpdump. This provides easy credential harvesting for lateral movement.

REMEDIATION STEPS

Step 1: Replace FTP with SFTP or SCP

Install and configure OpenSSH Server (Windows)

```
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
Start-Service sshd
Set-Service -Name sshd -StartupType Automatic
```

Disable FTP service

```
Stop-Service -Name FTPSVC
Set-Service -Name FTPSVC -StartupType Disabled
```

Step 2: Replace Telnet with SSH

Disable Telnet Client feature

```
Disable-WindowsOptionalFeature -Online -FeatureName TelnetClient
```

For network devices, enable SSH and disable Telnet

```
# Cisco IOS example
line vty 0 15
  transport input ssh
  transport output ssh
no line vty 0 15 transport input telnet
```

Step 3: Enforce HTTPS for web applications

IIS -- require SSL

```
# PowerShell  --  require SSL on IIS site
Set-WebConfigurationProperty -Filter "system.webServer/security/access" `
  -Name "sslFlags" -Value "Ssl" -PSPath "IIS:\Sites\Default Web Site"
```

Configure HTTP to HTTPS redirect

```
# IIS URL Rewrite rule (web.config)
# Or deploy an HSTS header via GPO/application config
```

GOTCHAS

- ! Automated scripts and scheduled tasks may use FTP -- audit scheduled tasks and scripts before disabling
- ! Legacy applications may require cleartext protocols -- document these as accepted risks with compensating controls
- ! Network devices may need firmware updates to support SSH -- plan maintenance windows

VERIFICATION

Scan for remaining cleartext services

```
nmap -sV -p 21,23,80 --open 10.0.0.0/24 -oG cleartext-services.txt
```

REFERENCES

https://attack.mitre.org/techniques/T1040/

ACTIVE DIRECTORY

AD-001 Kerberoastable Service Accounts

HIGH | Effort: Medium | Requires: Domain Admin, Service Account Owner

DESCRIPTION

Service accounts with Service Principal Names (SPNs) registered in Active Directory have a weakness in how Kerberos authentication works: any domain user -- even a low-privilege one -- can request an encrypted copy of the service account's password hash from the domain controller. This is a normal Kerberos operation and generates no security alerts. The attacker then takes that encrypted data offline and uses password-cracking tools to recover the plaintext password. Because service accounts often have elevated privileges and passwords that never change, this is one of the highest-value attacks in Active Directory environments.

IMPACT

Attackers with any domain user account can extract service account password hashes without triggering alerts. If the service account has elevated privileges (Domain Admin, database admin, etc.), this provides direct privilege escalation. No special tools or admin access required -- only standard domain user credentials.

REMEDIATION STEPS

Step 1: Identify Kerberoastable accounts

PowerShell -- find all accounts with SPNs

```
Get-ADUser -Filter {ServicePrincipalName -ne "$null"} -Properties ServicePrincipalName,PasswordLa...
  Select-Object SamAccountName, ServicePrincipalName, PasswordLastSet, Enabled |
  Format-Table -AutoSize
```

Note: Focus on accounts with passwords older than 1 year and accounts with admin group memberships

Step 2: Convert to Group Managed Service Accounts (gMSA)

Create KDS root key (one-time, domain-wide)

```
Add-KdsRootKey -EffectiveImmediately
```

Create a gMSA

```
New-ADServiceAccount -Name "svc_sqlprod" `
  -DNSHostName "svc_sqlprod.domain.local" `
  -PrincipalsAllowedToRetrieveManagedPassword "SQLServers$" `
  -KerberosEncryptionType AES256
```

Install the gMSA on the target server

```
Install-ADServiceAccount -Identity "svc_sqlprod"
Test-ADServiceAccount -Identity "svc_sqlprod"  # Should return True
```

Note: gMSAs automatically rotate 240-character random passwords every 30 days

Step 3: If gMSA is not possible: enforce strong passwords and AES encryption

Set a 30+ character password on the service account

```
$password = ConvertTo-SecureString (
  -join ((48..122) | Get-Random -Count 32 | ForEach-Object {[char]$_})
) -AsPlainText -Force
Set-ADAccountPassword -Identity "svc_sql" -NewPassword $password
```

Enforce AES-only Kerberos encryption (blocks RC4 ticket requests)

```
Set-ADUser -Identity "svc_sql" -KerberosEncryptionType AES256
# Also set via AD Users & Computers > Account tab >
# "This account supports Kerberos AES 256 bit encryption"
```

Note: AES tickets are significantly harder to crack than RC4 tickets

GOTCHAS

- ! Changing service account passwords requires updating the password on all services using that account
- ! gMSAs require Windows Server 2012+ domain functional level and a 2012+ DC
- ! Some applications (older SQL Server, IIS app pools) may not support gMSA -- check vendor documentation
- ! Disabling RC4 encryption may break older Kerberos clients -- test in a pilot group first

VERIFICATION

Re-run Kerberoast to confirm accounts are no longer vulnerable

```
# From audit/pentest perspective
GetUserSPNs.py domain.local/audituser:password -dc-ip 10.0.0.1 -request
# gMSA tickets will be uncrackable; AES-only tickets are extremely difficult
```

REFERENCES

https://attack.mitre.org/techniques/T1558/003/
https://learn.microsoft.com/en-us/windows-server/security/group-managed-service-accounts/group-managed-service-accounts-overview

AD-002 AS-REP Roastable Accounts

HIGH | Effort: Low | Requires: Domain Admin

DESCRIPTION

Active Directory accounts can have a setting called "Do not require Kerberos preauthentication" checked in their properties. When this is enabled, anyone -- even without a valid domain account -- can request encrypted password data for that account from the domain controller. The attacker can then crack this data offline to recover the password. This setting is rarely needed and is usually left on by mistake during account creation or migration.

IMPACT

An attacker can obtain crackable password hashes for these accounts without any authentication. If the account has privileged group memberships or access to sensitive resources, this provides easy initial access or privilege escalation.

REMEDIATION STEPS

Step 1: Identify accounts with pre-auth disabled

PowerShell -- find AS-REP roastable accounts

```
Get-ADUser -Filter {DoesNotRequirePreAuth -eq $true} -Properties DoesNotRequirePreAuth,MemberOf |
  Select-Object SamAccountName, DoesNotRequirePreAuth, @{N='Groups';E={($_.MemberOf | ForEach-Obj...
  Format-Table -AutoSize
```

Step 2: Enable pre-authentication

Enable Kerberos pre-authentication for identified accounts

```
Set-ADAccountControl -Identity "username" -DoesNotRequirePreAuth $false
```

Bulk fix -- enable pre-auth on all affected accounts

```
Get-ADUser -Filter {DoesNotRequirePreAuth -eq $true} |
  Set-ADAccountControl -DoesNotRequirePreAuth $false
```

Note: There are very few legitimate reasons to disable pre-auth -- usually only for legacy Unix Kerberos clients

Step 3: Force password reset on affected accounts

These passwords may already be compromised -- force rotation

```
Set-ADUser -Identity "username" -ChangePasswordAtLogon $true
```

GOTCHAS

! Some legacy applications (old Linux/Unix Kerberos implementations) may genuinely require pre-auth disabled
! Check if any automation sets this flag -- it may get re-enabled by scripts or provisioning tools
! The account password should be changed after enabling pre-auth, as the hash may already have been captured

VERIFICATION

Confirm no accounts remain with pre-auth disabled

```
Get-ADUser -Filter {DoesNotRequirePreAuth -eq $true} |
  Measure-Object | Select-Object Count  # Should be 0
```

REFERENCES

https://attack.mitre.org/techniques/T1558/004/

AD-003 Unconstrained Delegation

CRITICAL | Effort: Medium | Requires: Domain Admin

DESCRIPTION

Servers configured with unconstrained delegation can impersonate any user that authenticates to them. When a user connects to a service on an unconstrained delegation host, their full TGT is sent and cached on that server. An attacker who compromises such a host can extract these TGTs and use them to impersonate any user -- including Domain Admins.

IMPACT

Compromising a single unconstrained delegation host can lead to full domain compromise. The attacker can coerce authentication from the Domain Controller itself (via Print Spooler or other methods) and use the captured TGT to perform a DCSync attack, extracting all domain password hashes.

REMEDIATION STEPS

Step 1: Identify unconstrained delegation hosts

PowerShell -- find all unconstrained delegation computers (excluding DCs)

```
Get-ADComputer -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation,DNSHostN...
  Where-Object {$_.DistinguishedName -notlike "*Domain Controllers*"} |
  Select-Object Name, DNSHostName, TrustedForDelegation
```

Step 2: Migrate to constrained or resource-based constrained delegation

Remove unconstrained delegation

```
Set-ADComputer -Identity "SERVER01" -TrustedForDelegation $false
```

Configure constrained delegation instead (if delegation is needed)

```
Set-ADComputer -Identity "SERVER01" -Add @{
  'msDS-AllowedToDelegateTo' = @(
    'MSSQLSvc/sqlserver.domain.local:1433',
    'HTTP/webapp.domain.local'
  )
}
# Also enable "Use any authentication protocol" (protocol transition) if needed:
Set-ADAccountControl -Identity "SERVER01$" -TrustedToAuthForDelegation $true
```

Note: Constrained delegation limits which services the server can impersonate users to

Step 3: Alternative: Use Resource-Based Constrained Delegation (RBCD)

RBCD is configured on the target, not the source -- easier to manage

```
Set-ADComputer -Identity "SQLSERVER" -PrincipalsAllowedToDelegateToAccount (
  Get-ADComputer "WEBSERVER"
)
```

GOTCHAS

- ! Domain Controllers always have unconstrained delegation -- this cannot be changed and is by design
- ! Removing unconstrained delegation may break applications that rely on double-hop authentication (e.g., IIS to SQL)
- ! Test delegation changes in a staging environment -- broken delegation causes authentication failures
- ! Document each delegation relationship: source host, target service, business justification

VERIFICATION

Confirm no non-DC computers have unconstrained delegation

```
Get-ADComputer -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation |
  Where-Object {$_.DistinguishedName -notlike "*Domain Controllers*"} |
  Measure-Object | Select-Object Count  # Should be 0
```

REFERENCES

https://attack.mitre.org/techniques/T1558/
https://posts.specterops.io/hunting-in-active-directory-unconstrained-delegation-forests-trusts-71f2b33688e1

AD-004 NTLM Relay Attack Paths

HIGH | Effort: Medium | Requires: Domain Admin, GPO Admin

DESCRIPTION

When a user authenticates to a service using NTLM (the older Windows authentication protocol), an attacker positioned on the network can intercept that authentication attempt and forward ("relay") it to a different service. The target service believes the attacker is the legitimate user and grants access -- all without the attacker knowing the user's password. This is possible when SMB signing is disabled, LDAP signing/channel binding is not enforced, or other services accept relayed NTLM authentication.

IMPACT

Attackers can relay captured credentials to create new domain accounts via LDAP, modify ACLs to grant themselves elevated privileges, access file shares, or execute commands on remote systems -- all without knowing the victim's password.

REMEDIATION STEPS

Step 1: Enforce SMB signing (see NET-002)

Require SMB signing on all domain members

```
# GPO: Computer Configuration > Windows Settings > Security Settings > Local Policies > Security ...
# Microsoft network client: Digitally sign communications (always) -> Enabled
# Microsoft network server: Digitally sign communications (always) -> Enabled
```

Step 2: Enforce LDAP signing

Require LDAP signing on Domain Controllers

```
# GPO linked to Domain Controllers OU:
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options:
  Domain controller: LDAP server signing requirements -> Require signing
```

Require LDAP signing on clients

```
# GPO linked to all OUs:
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options:
  Network security: LDAP client signing requirements -> Require signing
```

Step 3: Enable LDAP channel binding

Set channel binding on Domain Controllers

```
# Registry (deploy via GPO):
reg add "HKLM\System\CurrentControlSet\Services\NTDS\Parameters" /v LdapEnforceChannelBinding /t ...
# 0 = Never, 1 = When supported, 2 = Always
```

Note: Setting 2 (Always) may break older clients -- test with setting 1 first

Step 4: Enable Extended Protection for Authentication (EPA) on web services

IIS -- enable EPA

```
# IIS Manager > Site > Authentication > Windows Authentication > Advanced Settings
# Extended Protection: Required
# Or via PowerShell:
Set-WebConfigurationProperty -Filter "system.webServer/security/authentication/windowsAuthenticat...
  -Name "extendedProtection.tokenChecking" -Value "Require" -PSPath "IIS:\Sites\Default Web Site"
```

GOTCHAS

- ! LDAP channel binding level 2 (Always) may break legacy LDAP clients -- monitor event IDs 3039 and 3040 on DCs
- ! Before enforcing LDAP signing, audit current unsigned LDAP connections: Event ID 2889 on DCs
- ! Some Linux LDAP clients may need configuration changes to support LDAP signing
- ! EPA on IIS may break older browsers or non-domain-joined clients

VERIFICATION

Check LDAP signing events on DCs (unsigned bind attempts)

```
Get-WinEvent -FilterHashtable @{LogName='Directory Service'; Id=2889} -MaxEvents 10 |
  Format-Table TimeCreated, Message -Wrap
```

Test NTLM relay with ntlmrelayx (from audit box)

```
ntlmrelayx.py -t ldap://DC01 -smb2support  # Should fail if properly configured
```

REFERENCES

https://attack.mitre.org/techniques/T1557/001/

https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/domain-controller-ldap-server-signing-requirements

AD-005 Print Spooler on Domain Controllers

CRITICAL | Effort: Low | Requires: Domain Admin

DESCRIPTION

The Print Spooler service running on Domain Controllers enables the PrinterBug/SpoolSample attack, where an attacker can coerce the DC to authenticate to an attacker-controlled host. Combined with unconstrained delegation or NTLM relay, this leads to full domain compromise.

IMPACT

An attacker can force a Domain Controller to authenticate to a host they control, capturing the DC machine account's TGT or relaying its NTLM authentication. This enables DCSync attacks and complete domain compromise. PrintNightmare (CVE-2021-34527) also requires the Print Spooler service.

REMEDIATION STEPS

Step 1: Disable Print Spooler on all Domain Controllers

PowerShell -- stop and disable the service

```
# Run on each DC, or deploy via GPO:
Stop-Service -Name Spooler -Force
Set-Service -Name Spooler -StartupType Disabled
```

GPO -- disable the service across all DCs

```
# GPO linked to Domain Controllers OU:
Computer Configuration > Policies > Windows Settings > Security Settings >
System Services > Print Spooler -> Startup Mode: Disabled
```

Step 2: Disable on any server that does not need printing

Audit and disable on non-print servers

```
# Find servers with Spooler running:
Get-ADComputer -Filter {OperatingSystem -like "*Server*"} |
  ForEach-Object {
    $svc = Get-Service -Name Spooler -ComputerName $_.Name -ErrorAction SilentlyContinue
    if ($svc.Status -eq 'Running') {
      [PSCustomObject]@{Server=$_.Name; Status=$svc.Status; StartType=$svc.StartType}
    }
  } | Format-Table
```

GOTCHAS

- ! Do NOT disable Print Spooler on print servers -- only on DCs and non-print servers
- ! Some GPO features (deploying printers via GPO) may require the Spooler service -- test first
- ! If Point and Print is needed, restrict it via GPO: 'Point and Print Restrictions' -> Enabled with specific print servers

VERIFICATION

Confirm Spooler is disabled on all DCs

```
Get-ADDomainController -Filter * | ForEach-Object {
  $svc = Get-Service -Name Spooler -ComputerName $_.HostName -ErrorAction SilentlyContinue
  [PSCustomObject]@{DC=$_.HostName; SpoolerStatus=$svc.Status; StartType=$svc.StartType}
} | Format-Table
```

REFERENCES

https://attack.mitre.org/techniques/T1547/012/
https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34527

AD-006 LAPS Not Deployed

HIGH | Effort: Medium | Requires: Domain Admin, Schema Admin (for LAPS installation)

DESCRIPTION

Without Local Administrator Password Solution (LAPS), local administrator passwords are either identical across all workstations (set during imaging) or managed manually. This means compromising one local admin password grants access to every machine with the same password.

IMPACT

A single compromised workstation reveals the local admin password, which works on every other workstation in the environment. This enables rapid lateral movement across the entire domain without needing domain credentials.

REMEDIATION STEPS

Step 1: Install Windows LAPS (built into Windows 10 21H2+ / Server 2019+)

Update AD schema for Windows LAPS

```
Update-LapsADSchema
```

Set permissions for computer objects to write their password

```
Set-LapsADComputerSelfPermission -Identity "OU=Workstations,DC=domain,DC=local"
```

Configure read permissions for IT admins

```
Set-LapsADReadPasswordPermission -Identity "OU=Workstations,DC=domain,DC=local" `
  -AllowedPrincipals "domain\DesktopAdmins"
```

Step 2: Configure LAPS via Group Policy

GPO settings for Windows LAPS

```
Computer Configuration > Policies > Administrative Templates >
System > LAPS:
  Configure password backup directory -> Enabled -> Active Directory
  Password Settings -> Enabled
    Complexity: Large letters + small letters + numbers + specials
    Length: 20
    Age (days): 30
```

Step 3: Legacy LAPS (for older OS versions)

Download and install Microsoft LAPS from Microsoft

```
# Install the LAPS management tools on a management server:
msiexec /i LAPS.x64.msi ADDLOCAL=ALL /quiet
# Extend AD schema:
Import-Module AdmPwd.PS
Update-AdmPwdADSchema
# Set permissions:
Set-AdmPwdComputerSelfPermission -OrgUnit "OU=Workstations,DC=domain,DC=local"
```

GOTCHAS

- ! Windows LAPS and Legacy LAPS can coexist but should not manage the same machine
- ! LAPS passwords are stored in AD -- ensure only authorized groups can read them (check with Get-LapsADPassword)
- ! SCCM/Intune task sequences that use the local admin need updating to fetch the LAPS password
- ! Document the LAPS admin password retrieval process for helpdesk and IT support

VERIFICATION

Verify LAPS is managing passwords on target OUs

```
Get-LapsADPassword -Identity "WORKSTATION01" -AsPlainText
# Should return a unique, complex password and expiration date
```

Find computers without LAPS passwords

```
Get-ADComputer -Filter * -SearchBase "OU=Workstations,DC=domain,DC=local" -Properties ms-Mcs-AdmP...
  Where-Object {!$_.'ms-Mcs-AdmPwd'} |
  Select-Object Name
```

REFERENCES

https://learn.microsoft.com/en-us/windows-server/identity/laps/laps-overview
https://attack.mitre.org/techniques/T1078/003/

AD-007 Weak Domain Password Policy

HIGH | Effort: Low | Requires: Domain Admin

DESCRIPTION

A weak domain password policy (short minimum length, no complexity, long max age or no expiry) makes password-based attacks significantly easier. Attackers can crack captured hashes faster and password spraying attacks are more likely to succeed.

IMPACT

Weak passwords are crackable within minutes to hours using modern GPU hardware. Combined with Kerberoasting, AS-REP Roasting, or NTLM hash capture, weak password policies dramatically reduce the effort required to compromise accounts.

REMEDIATION STEPS

Step 1: Audit current password policy

View the current domain password policy

```
Get-ADDefaultDomainPasswordPolicy | Format-List
```

Check for Fine-Grained Password Policies

```
Get-ADFineGrainedPasswordPolicy -Filter * | Format-Table Name,MinPasswordLength,Precedence
```

Step 2: Implement a strong password policy

Set domain password policy via GPO (Default Domain Policy)

```
Computer Configuration > Policies > Windows Settings > Security Settings >
Account Policies > Password Policy:
  Minimum password length: 14 characters (CIS Benchmark recommendation)
  Password must meet complexity requirements: Enabled
  Minimum password age: 1 day
  Maximum password age: 365 days (NIST suggests no expiry with monitoring)
  Enforce password history: 24 passwords
Account Policies > Account Lockout Policy:
  Account lockout threshold: 10 invalid attempts
  Account lockout duration: 30 minutes
  Reset account lockout counter after: 30 minutes
```

Step 3: Implement Fine-Grained Password Policy for privileged accounts

Create a stricter policy for admins

```
New-ADFineGrainedPasswordPolicy -Name "Admin-StrongPassword" `
  -Precedence 10 `
  -MinPasswordLength 20 `
  -ComplexityEnabled $true `
  -MaxPasswordAge "90.00:00:00" `
  -LockoutThreshold 5 `
  -LockoutDuration "00:30:00" `
  -LockoutObservationWindow "00:30:00"
Add-ADFineGrainedPasswordPolicySubject -Identity "Admin-StrongPassword" `
  -Subjects "Domain Admins","Enterprise Admins","Schema Admins"
```

Step 4: Consider deploying a banned password list

Azure AD Password Protection (hybrid) blocks common passwords

```
# Install Azure AD Password Protection Proxy and DC Agent
# Download from: Microsoft Download Center
# Requires Azure AD Premium P1 or P2 license
```

Note: This blocks passwords like 'Summer2024!', 'Company123!', 'Password1' that meet complexity but are easily guessed

GOTCHAS

- ! Changing password policy does NOT force existing passwords to change -- users keep old passwords until they expire
- ! Password complexity alone is insufficient -- 'Summer2024!' meets complexity but is trivially guessed
- ! Very strict lockout policies (3 attempts) can cause denial of service -- 10 attempts is a good balance
- ! NIST 800-63B recommends against forced periodic password changes when proper monitoring is in place

VERIFICATION

Verify the policy is applied

```
Get-ADDefaultDomainPasswordPolicy
```

Test password strength with internal audit (requires DA)

```
# Use DSInternals to audit password quality
Get-ADReplAccount -All -Server DC01 |
  Test-PasswordQuality -WeakPasswordsFile rockyou-top1000.txt
```

REFERENCES

https://pages.nist.gov/800-63-4/sp800-63b.html
https://attack.mitre.org/techniques/T1110/

AD-008 GPP Passwords (MS14-025)

HIGH | Effort: Low | Requires: Domain Admin

DESCRIPTION

Group Policy Preferences (GPP) allowed administrators to set local administrator passwords, map drives with credentials, and configure scheduled tasks with embedded passwords. These passwords are stored in SYSVOL in AES-256 encrypted XML files, but Microsoft published the decryption key in MSDN documentation, making them trivially recoverable by any domain user.

IMPACT

Any authenticated domain user can read SYSVOL and decrypt GPP passwords, potentially gaining local admin or service account credentials. Despite MS14-025 being patched in 2014, many environments still have legacy GPP files containing passwords.

REMEDIATION STEPS

Step 1: Find and remove GPP files with embedded passwords

Search SYSVOL for GPP XML files containing cpassword

```
Get-ChildItem "\\domain.local\SYSVOL\domain.local\Policies" -Recurse -Include Groups.xml,Services...
  Select-String -Pattern "cpassword" |
  Select-Object Path, Line
```

PowerShell -- decrypt found GPP passwords for impact assessment

```
# Use Get-GPPPassword from PowerSploit or:
findstr /S /I "cpassword" "\\domain.local\SYSVOL\domain.local\Policies\*.xml"
```

Step 2: Delete the offending GPO preferences

Edit the GPO and remove the password-containing preference

```
# Open Group Policy Management Console (GPMC)
# Navigate to the identified GPO
# Remove the preference item that contains the password
# Or delete the entire GPO if it only existed for password deployment
```

Note: Document what each GPP was doing before removing it -- may need to replace with LAPS or other solutions

Step 3: Change any passwords that were exposed

Rotate credentials for any account whose password was in GPP

```
# Identify the accounts from the GPP XML and change their passwords
Set-ADAccountPassword -Identity "affected_account" -Reset `
  -NewPassword (ConvertTo-SecureString "NewStr0ngP@ssw0rd!" -AsPlainText -Force)
```

GOTCHAS

- ! GPP files may be in old/deleted GPOs still present in SYSVOL -- check for orphaned policy folders
- ! The MS14-025 patch only prevents CREATING new GPP passwords -- existing ones are NOT removed
- ! Replace GPP password deployment with LAPS for local admin passwords

VERIFICATION

Confirm no GPP passwords remain in SYSVOL

```
Get-ChildItem "\\domain.local\SYSVOL" -Recurse -Include *.xml |
  Select-String "cpassword" | Measure-Object | Select-Object Count  # Should be 0
```

REFERENCES

https://attack.mitre.org/techniques/T1552/006/
https://learn.microsoft.com/en-us/security-updates/securitybulletins/2014/ms14-025

AD-009 LDAP Signing Not Required

HIGH | Effort: Low | Requires: Domain Admin

DESCRIPTION

When LDAP signing is not required on Domain Controllers, attackers can perform NTLM relay attacks against LDAP to modify Active Directory objects, add users to privileged groups, or configure resource-based constrained delegation for privilege escalation.

IMPACT

Without LDAP signing, relayed NTLM authentication can be used to make arbitrary changes to Active Directory, including adding accounts to Domain Admins, modifying ACLs, and creating new privileged accounts.

REMEDIATION STEPS

Step 1: Audit current unsigned LDAP connections first

Enable diagnostic logging and check for unsigned LDAP binds (event 2889 on DCs)

```
# Enable diagnostic logging first:
reg add "HKLM\System\CurrentControlSet\Services\NTDS\Diagnostics" /v "16 LDAP Interface Events" /...
# Then monitor:
Get-WinEvent -FilterHashtable @{LogName='Directory Service'; Id=2889} -MaxEvents 20 |
  Format-Table TimeCreated, Message -Wrap
```

Note: Monitor for 1-2 weeks before enforcing to identify clients that will break. Enforcing without auditing first can silently break LDAP clients

Step 2: Enable LDAP signing requirement

Configure via GPO on Domain Controllers

```
# GPO linked to Domain Controllers OU:
Computer Configuration > Windows Settings > Security Settings >
Local Policies > Security Options:
  Domain controller: LDAP server signing requirements -> Require signing
```

Configure LDAP client signing on all domain members

```
# GPO linked to all OUs:
Computer Configuration > Windows Settings > Security Settings >
Local Policies > Security Options:
  Network security: LDAP client signing requirements -> Require signing
```

GOTCHAS

- ! Linux LDAP clients (PAM, SSSD, nslcd) may need configuration changes to support LDAP signing
- ! Some legacy applications (older Java LDAP clients, network appliances) may not support signed LDAP
- ! Monitor event 2889 on DCs for at least 2 weeks before enforcing -- this identifies clients that will break
- ! LDAPS (LDAP over SSL on port 636) is NOT the same as LDAP signing -- both should be configured

VERIFICATION

Confirm LDAP signing is enforced

```
Get-ItemProperty "HKLM:\System\CurrentControlSet\Services\NTDS\Parameters" |
  Select-Object LDAPServerIntegrity  # Should be 2
```

REFERENCES

https://learn.microsoft.com/en-us/troubleshoot/windows-server/active-directory/enable-ldap-signing-in-windows-server

AD-010 Excessive Machine Account Quota

MEDIUM | Effort: Low | Requires: Domain Admin

DESCRIPTION

By default, any authenticated domain user can create up to 10 machine (computer) accounts in the domain. Attackers abuse this to create machine accounts they control, which can then be used for resource-based constrained delegation attacks or other privilege escalation paths.

IMPACT

An attacker with any domain user account can create a machine account and use it to perform RBCD attacks, leading to privilege escalation on target servers. This provides a reliable path from low-privilege domain user to local admin on server resources.

REMEDIATION STEPS

Step 1: Set Machine Account Quota to 0

PowerShell -- set the attribute to 0

```
Set-ADDomain -Identity "domain.local" -Replace @{"ms-DS-MachineAccountQuota"=0}
```

Verify the change

```
Get-ADObject -Identity (Get-ADDomain).DistinguishedName -Properties ms-DS-MachineAccountQuota |
  Select-Object ms-DS-MachineAccountQuota  # Should be 0
```

Step 2: Delegate machine account creation to authorized groups

Grant specific groups the right to join computers to the domain

```
# In AD Users and Computers > Delegate Control on the target OU:
# 1. Select the group (e.g., "HelpDesk-ComputerJoin")
# 2. Delegate "Create Computer Objects" and "Delete Computer Objects"
# This replaces the broad machine account quota with specific delegation
```

GOTCHAS

- ! Users who pre-stage computer accounts in AD before domain join are not affected by quota changes
- ! Some automated deployment tools (MDT, SCCM) may create machine accounts -- ensure they use a delegated service account
- ! This does not remove existing machine accounts created by non-admins -- audit and clean up orphaned accounts

VERIFICATION

Confirm quota is set to 0

```
Get-ADObject -Identity (Get-ADDomain).DistinguishedName -Properties ms-DS-MachineAccountQuota |
  Select-Object ms-DS-MachineAccountQuota
```

REFERENCES

https://attack.mitre.org/techniques/T1136/002/

AD-011 DCSync-Capable Non-Admin Accounts

CRITICAL | Effort: Low | Requires: Domain Admin

DESCRIPTION

Accounts with Replicating Directory Changes and Replicating Directory Changes All permissions on the domain root can perform a DCSync attack, extracting all password hashes from Active Directory without ever touching a Domain Controller. By default, only Domain Admins, Enterprise Admins, and the DC machine accounts should have these rights.

IMPACT

Any account with these permissions can extract every password hash in the domain, including the krbtgt hash (enabling Golden Ticket attacks). This is equivalent to full domain compromise.

REMEDIATION STEPS

Step 1: Audit replication permissions

Find all principals with DCSync rights

```
Import-Module ActiveDirectory
$domainDN = (Get-ADDomain).DistinguishedName
$acl = Get-ACL "AD:\$domainDN"
$acl.Access | Where-Object {
  $_.ObjectType -eq "1131f6ad-9c07-11d1-f79f-00c04fc2dcd2" -or  # DS-Replication-Get-Changes-All
  $_.ObjectType -eq "1131f6aa-9c07-11d1-f79f-00c04fc2dcd2" -or  # DS-Replication-Get-Changes
  $_.ObjectType -eq "89e95b76-444d-4c62-991a-0facbeda640c"      # DS-Replication-Get-Changes-In-F...
} | Select-Object IdentityReference, ActiveDirectoryRights, ObjectType |
  Format-Table -AutoSize
```

Note: Expected principals: Domain Admins, Enterprise Admins, SYSTEM, Domain Controllers, and the domain itself

Step 2: Remove unauthorized replication rights

Remove DCSync permissions from unauthorized accounts

```
# Using AD Users and Computers or dsacls:
dsacls "DC=domain,DC=local" /R "DOMAIN\unauthorized_account"
# Or via PowerShell:
$acl = Get-ACL "AD:\DC=domain,DC=local"
$acl.Access | Where-Object {$_.IdentityReference -eq "DOMAIN\unauthorized_account"} |
  ForEach-Object { $acl.RemoveAccessRule($_) }
Set-ACL "AD:\DC=domain,DC=local" $acl
```

Step 3: Change the krbtgt password if compromise is suspected

Reset krbtgt password (do this TWICE, 12 hours apart)

```
# First reset (use the Microsoft krbtgt reset script from GitHub, or manual reset):
Set-ADAccountPassword -Identity krbtgt -Reset `
  -NewPassword (ConvertTo-SecureString (New-Guid).Guid -AsPlainText -Force)
# Wait 12+ hours for replication, then reset again
```

Note: Resetting krbtgt invalidates all existing Kerberos tickets -- plan for service disruption

GOTCHAS

- ! Azure AD Connect service accounts legitimately need replication rights -- verify these are expected
- ! Some backup solutions (e.g., Veeam) require replication rights -- document these as accepted
- ! Resetting krbtgt will invalidate ALL Kerberos tickets domain-wide -- plan maintenance windows
- ! After removing unauthorized rights, force a password change on the affected account

VERIFICATION

Re-audit replication permissions and confirm only expected principals remain

```
(Get-ACL "AD:\$(( Get-ADDomain).DistinguishedName)").Access |
  Where-Object {$_.ObjectType -in @("1131f6ad-9c07-11d1-f79f-00c04fc2dcd2","1131f6aa-9c07-11d1-f7...
  Select-Object IdentityReference
```

REFERENCES

https://attack.mitre.org/techniques/T1003/006/

AD-012 Null Session Enumeration

MEDIUM | Effort: Low | Requires: Domain Admin, GPO Admin

DESCRIPTION

Null sessions allow unauthenticated access to enumerate domain information including user lists, group memberships, password policies, and share names. This provides attackers with critical reconnaissance data without requiring any credentials.

IMPACT

An attacker without any credentials can enumerate valid usernames for password spraying, identify privileged accounts, discover the password policy to optimize attacks, and find accessible network shares.

REMEDIATION STEPS

Step 1: Restrict anonymous access via GPO

Configure security settings

```
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options:
  Network access: Do not allow anonymous enumeration of SAM accounts -> Enabled
  Network access: Do not allow anonymous enumeration of SAM accounts and shares -> Enabled
  Network access: Restrict anonymous access to Named Pipes and Shares -> Enabled
  Network access: Let Everyone permissions apply to anonymous users -> Disabled
```

Restrict null session pipes and shares

```
# Clear the list of named pipes and shares accessible anonymously:
Computer Configuration > Windows Settings > Security Settings >
Local Policies > Security Options:
  Network access: Named Pipes that can be accessed anonymously -> (clear the list)
  Network access: Shares that can be accessed anonymously -> (clear the list)
```

Step 2: Disable the RestrictAnonymous bypass

Set RestrictAnonymous to 2 on sensitive servers

```
reg add "HKLM\System\CurrentControlSet\Control\LSA" /v RestrictAnonymous /t REG_DWORD /d 2 /f
# 0 = None, 1 = Do not allow enumeration of SAM accounts, 2 = No access without explicit anonymou...
```

Note: Level 2 is more restrictive but may break some legacy applications

GOTCHAS

- ! RestrictAnonymous = 2 may break legacy applications that rely on anonymous enumeration
- ! Some monitoring tools use anonymous LDAP binds -- these will need service accounts
- ! Domain Controllers should have additional hardening beyond member servers

VERIFICATION

Test null session access from an external/unauthenticated perspective

```
# From a pentest/audit box:
enum4linux -a -U target_ip  # Should return no useful information
rpcclient -U "" -N target_ip -c "enumdomusers"  # Should be denied
```

REFERENCES

https://attack.mitre.org/techniques/T1087/002/

CREDENTIAL MANAGEMENT

CRED-001 Default or Weak Credentials

HIGH | Effort: Low | Requires: System Admin, Application Owner

DESCRIPTION

Systems, applications, and services using default, blank, or easily guessable credentials provide trivial unauthorized access. Common examples include admin/admin, sa/sa (SQL), admin/password on web management interfaces, and vendor default credentials on appliances.

IMPACT

Default credentials provide immediate authenticated access to systems and applications, bypassing all authentication controls. This is often the easiest path to initial access or privilege escalation in internal penetration tests.

REMEDIATION STEPS

Step 1: Inventory all systems and change defaults

Audit for common default credentials across services

```
# Network devices (Cisco, HP, etc.):
# Change admin/admin, cisco/cisco, admin/password
# SQL Server: Disable SA account or set complex password
# Web management interfaces: Change defaults on iDRAC, iLO, IPMI, UPS, etc.
# Application defaults: Tomcat (tomcat/tomcat), Jenkins (no auth), JBoss, etc.
```

Use CrackMapExec for bulk default credential testing

```
# Test for blank/default passwords (from audit box):
nxc smb targets.txt -u Administrator -p "" --local-auth
nxc smb targets.txt -u Administrator -p "Password1" --local-auth
```

Step 2: Implement a credential management process

Deploy a privileged access management (PAM) solution or password vault

```
# Popular options:
# - CyberArk, BeyondTrust, Delinea (enterprise)
# - KeePass, Bitwarden (team/personal)
# - HashiCorp Vault (secrets management)
```

Note: All service and admin accounts should be in a password vault with rotation policies

GOTCHAS

- ! Changing credentials on network infrastructure may cause outages -- schedule maintenance windows
- ! Embedded systems (IPMI, iLO, iDRAC) are often forgotten -- audit out-of-band management interfaces
- ! Vendor appliances may have hard-coded back-door accounts -- check vendor security advisories

VERIFICATION

Re-test with common default credential lists

```
# From audit perspective:
nxc smb targets.txt -u defaults_users.txt -p defaults_pass.txt --no-bruteforce
```

REFERENCES

https://attack.mitre.org/techniques/T1078/001/

CRED-002 Cleartext Passwords in Network Shares and Scripts

HIGH | Effort: Medium | Requires: System Admin, Application Owner

DESCRIPTION

Credentials stored in plaintext within batch files, PowerShell scripts, configuration files, and network shares are accessible to anyone with read access. Common locations include SYSVOL scripts, IT department shares, and deployment/automation scripts.

IMPACT

Attackers with basic domain user access can search file shares for plaintext credentials, often finding service account passwords, database connection strings, admin credentials, and API keys that enable significant privilege escalation.

REMEDIATION STEPS

Step 1: Scan for plaintext credentials

Search SYSVOL and common shares for credential patterns

```
# PowerShell  --  search for common password patterns
$shares = @("\\domain.local\SYSVOL", "\\fileserver\IT", "\\fileserver\scripts")
$patterns = @("password", "pwd", "credential", "cpassword", "connectionstring")
foreach ($share in $shares) {
  Get-ChildItem -Path $share -Recurse -Include *.ps1,*.bat,*.cmd,*.vbs,*.txt,*.xml,*.ini,*.cfg,*....
    Select-String -Pattern $patterns -SimpleMatch |
    Select-Object Path, LineNumber, Line
}
```

Step 2: Replace with secure alternatives

Use Windows Credential Manager for scripts

```
# Store credential securely (will prompt for password):
cmdkey /add:ServerName /user:domain\svc_account
# Or via PowerShell (prompts interactively):
$cred = Get-Credential
cmdkey /add:ServerName /user:$($cred.UserName) /pass:$($cred.GetNetworkCredential().Password)
```

Use encrypted credential files for PowerShell automation

```
# Create encrypted credential file (machine+user-specific):
Get-Credential | Export-Clixml -Path "C:\Scripts\creds.xml"
# Use in script:
$cred = Import-Clixml -Path "C:\Scripts\creds.xml"
```

Use gMSA for service accounts (see AD-001)

```
# gMSAs eliminate the need to store service account passwords entirely
```

Step 3: Restrict share permissions

Remove broad read access from shares containing scripts

```
# Audit share permissions:
Get-SmbShareAccess -Name "IT" | Format-Table
# Remove 'Everyone' or 'Domain Users' from sensitive shares:
Revoke-SmbShareAccess -Name "IT" -AccountName "Everyone" -Force
```

GOTCHAS

! Developers may push back -- provide alternative credential storage solutions before removing plaintext

! SYSVOL scripts used for logon/startup may need refactoring -- plan the transition
! Don't just delete scripts -- some may be critical for operations. Replace the credential storage method.
! Check version control systems (Git repos on internal servers) for committed credentials

VERIFICATION

Re-scan shares for plaintext credentials

```
Get-ChildItem "\\domain.local\SYSVOL" -Recurse -Include *.ps1,*.bat,*.xml |
  Select-String -Pattern "password|pwd|credential" -SimpleMatch |
  Measure-Object | Select-Object Count  # Should be 0 or known exceptions only
```

REFERENCES

https://attack.mitre.org/techniques/T1552/001/

CRED-003 WDigest Authentication Enabled

HIGH | Effort: Low | Requires: Domain Admin, GPO Admin

DESCRIPTION

WDigest authentication stores plaintext passwords in LSASS memory when enabled. On Windows 8.1/Server 2012 R2 and later, WDigest is disabled by default, but older systems or misconfigured GPOs may have it enabled, allowing attackers to extract cleartext passwords from memory.

IMPACT

An attacker with local admin access can dump LSASS memory and obtain cleartext domain passwords for all users who have logged into that system. This eliminates the need to crack password hashes entirely.

REMEDIATION STEPS

Step 1: Disable WDigest authentication

Set the UseLogonCredential registry value to 0 via GPO

```
# GPO Registry Preference:
HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest
Value: UseLogonCredential
Type: REG_DWORD
Data: 0
```

PowerShell -- immediate fix on individual hosts

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" `
  -Name "UseLogonCredential" -Value 0 -Type DWord
```

Note: This setting takes effect at next logon -- existing sessions in memory are not affected until the user logs off

Step 2: Install KB2871997 on older systems

Windows 7/Server 2008 R2 require this update for the registry key to work

```
wusa.exe KB2871997-x64.msu /quiet /norestart
```

Note: After installing the update, set the registry key as described above

GOTCHAS

! Existing logged-in sessions still have cleartext passwords in memory until the user logs off
! Windows 7/2008 R2 require KB2871997 before the UseLogonCredential registry key is recognized
! Some legacy applications may rely on WDigest -- extremely rare in modern environments

VERIFICATION

Verify the registry key is set

```
Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" |
  Select-Object UseLogonCredential  # Should be 0
```

REFERENCES

https://attack.mitre.org/techniques/T1003/001/

CRED-004 LSASS Not Protected (Credential Guard / RunAsPPL)

HIGH | Effort: Medium | Requires: Domain Admin, GPO Admin

DESCRIPTION

Without Credential Guard or LSASS running as a Protected Process Light (PPL), attackers with local admin access can dump LSASS memory using tools like Mimikatz to extract NTLM hashes, Kerberos tickets, and potentially cleartext passwords.

IMPACT

An attacker who gains local admin on any domain-joined workstation can extract cached credentials for all users who have logged into that machine, enabling rapid lateral movement across the domain.

REMEDIATION STEPS

Step 1: Enable LSA Protection (RunAsPPL)

Enable via GPO registry preference

```
# GPO Registry Preference:
HKLM\SYSTEM\CurrentControlSet\Control\LSA
Value: RunAsPPL
Type: REG_DWORD
Data: 1
```

PowerShell -- immediate fix

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\LSA" `
  -Name "RunAsPPL" -Value 1
# Requires reboot to take effect
```

Note: RunAsPPL prevents unsigned DLLs from loading into LSASS -- blocks most credential dumping tools

Step 2: Enable Credential Guard (Windows 10 Enterprise / Server 2016+)

Enable via GPO

```
Computer Configuration > Administrative Templates > System > Device Guard:
  Turn On Virtualization Based Security -> Enabled
    Credential Guard Configuration: Enabled with UEFI lock
```

Verify Credential Guard is running

```
Get-CimInstance -ClassName Win32_DeviceGuard -Namespace root\Microsoft\Windows\DeviceGuard |
  Select-Object SecurityServicesRunning
# Value should include 1 (Credential Guard)
```

Note: Credential Guard requires UEFI Secure Boot, virtualization extensions, and TPM 2.0

GOTCHAS

! Credential Guard is incompatible with some older applications that use NTLMv1 or older Kerberos DES encryption

- ! RunAsPPL may break third-party security products that inject DLLs into LSASS (some AV/EDR)
- ! Credential Guard requires hardware support (UEFI, VBS-capable CPU) -- not all systems qualify
- ! Enable RunAsPPL first (wider compatibility) and add Credential Guard where hardware supports it
- ! Test both in a pilot group before domain-wide deployment

VERIFICATION

Verify LSA protection is enabled

```
Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Control\LSA" | Select-Object RunAsPPL  # Should ...
```

Verify Credential Guard status

```
Get-ComputerInfo | Select-Object DeviceGuard*
```

REFERENCES

https://learn.microsoft.com/en-us/windows/security/identity-protection/credential-guard/
https://attack.mitre.org/techniques/T1003/001/

HOST SECURITY

HOST-001 Local Admin Password Reuse

HIGH | Effort: Medium | Requires: Domain Admin, GPO Admin

DESCRIPTION

When the same local administrator password is used across multiple workstations and servers (commonly set during OS imaging), compromising one system reveals the credentials for all systems sharing that password. This is the primary enabler of lateral movement in most environments.

IMPACT

An attacker who extracts the local admin hash from one workstation can use pass-the-hash to authenticate to every other system with the same password, rapidly expanding their access across the entire network.

REMEDIATION STEPS

Step 1: Deploy LAPS (see AD-006)

LAPS ensures every computer has a unique, regularly rotated local admin password

```
# See AD-006 for full LAPS deployment instructions
```

Step 2: Immediate mitigation -- randomize existing passwords

PowerShell script to set random local admin passwords (stopgap until LAPS)

```
# Run remotely against each host:
$computers = Get-ADComputer -Filter {OperatingSystem -like "*Windows*"}
foreach ($comp in $computers) {
  $newPass = -join ((65..90)+(97..122)+(48..57)+(33..38) | Get-Random -Count 20 | ForEach-Object ...
  Invoke-Command -ComputerName $comp.Name -ScriptBlock {
    param($p) net user Administrator $p
  } -ArgumentList $newPass
  # Store the password in a secure vault
}
```

Note: This is a temporary measure -- deploy LAPS for sustainable password management

GOTCHAS

- ! Randomizing passwords without tracking them will lock teams out -- use LAPS or a vault
- ! Some deployment tools (SCCM task sequences, MDT) set the local admin password during imaging -- update the image
- ! Remember to check both the default 'Administrator' account and any additional local admin accounts

VERIFICATION

Test that local admin hashes differ across machines

```
# From audit box -- compare NTLM hashes across systems:
nxc smb targets.txt -u Administrator -H <hash> --local-auth
# Should only work on the source machine, not others
```

REFERENCES

https://attack.mitre.org/techniques/T1078/003/

HOST-002 RDP Without Network Level Authentication (NLA)

MEDIUM | Effort: Low | Requires: System Admin, GPO Admin

DESCRIPTION

Remote Desktop Protocol without Network Level Authentication allows attackers to establish an RDP session before authenticating, exposing the system to credential relay attacks, BlueKeep-style vulnerabilities, and brute force attacks without rate limiting.

IMPACT

Without NLA, the RDP server allocates resources and presents a login screen before authentication, making it vulnerable to pre-authentication exploits and enabling more effective brute force attacks.

REMEDIATION STEPS

Step 1: Enable NLA via Group Policy

GPO setting for NLA

```
Computer Configuration > Policies > Administrative Templates >
Windows Components > Remote Desktop Services > Remote Desktop Session Host >
Security:
  Require user authentication for remote connections by using NLA -> Enabled
```

PowerShell -- enable immediately on individual hosts

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tc...
  -Name "UserAuthentication" -Value 1
```

Step 2: Also enforce TLS 1.2 for RDP encryption

Set minimum encryption level

```
# Same GPO path > Security:
# Set client connection encryption level -> High Level
# Require use of specific security layer for remote (RDP) connections -> SSL
# Ensure TLS 1.2 is the minimum via SCHANNEL settings (see WEB-001)
```

GOTCHAS

- ! NLA requires the connecting user to have 'Remote Desktop Users' group membership before connection
- ! Password-expired users cannot connect with NLA -- they must change their password via another method first
- ! Some older RDP clients (Windows XP, old thin clients) may not support NLA
- ! NLA uses CredSSP -- ensure CredSSP is not configured in a vulnerable manner (CVE-2018-0886)

VERIFICATION

Verify NLA is enabled

```
Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" |
  Select-Object UserAuthentication  # Should be 1
```

REFERENCES

https://attack.mitre.org/techniques/T1021/001/

HOST-003 PowerShell v2 Enabled

MEDIUM | Effort: Low | Requires: System Admin, GPO Admin

DESCRIPTION

PowerShell v2 lacks security features present in newer versions, including ScriptBlock Logging, Module Logging, AMSI (Anti-Malware Scan Interface), and Constrained Language Mode. Attackers explicitly downgrade to PowerShell v2 to bypass these security controls.

IMPACT

With PowerShell v2 available, attackers can bypass all PowerShell security logging and AMSI-based detection by simply running "powershell -version 2", effectively making their PowerShell-based attacks invisible to security tools.

REMEDIATION STEPS

Step 1: Remove PowerShell v2

Disable the PowerShell 2.0 feature

```
# PowerShell (run as admin):
Disable-WindowsOptionalFeature -Online -FeatureName MicrosoftWindowsPowerShellV2Root
Disable-WindowsOptionalFeature -Online -FeatureName MicrosoftWindowsPowerShellV2
```

GPO -- disable via Features on Demand

```
# DISM:
dism /online /Disable-Feature /FeatureName:MicrosoftWindowsPowerShellV2Root /NoRestart
```

Verify removal on servers

```
Get-WindowsFeature PowerShell-V2 | Select-Object Name, InstallState  # Should be "Removed"
```

GOTCHAS

- ! Some legacy scripts may depend on PowerShell v2 -- test in a pilot group first
- ! Exchange 2010 management tools required PowerShell v2 -- upgrade Exchange if still running 2010
- ! .NET Framework 2.0 is a dependency of PowerShell v2 -- removing PSv2 does not remove .NET 2.0

VERIFICATION

Confirm PowerShell v2 is not available

```
powershell -version 2 -command "Write-Host test" 2>&1
# Should return an error about the feature not being enabled
```

REFERENCES

https://attack.mitre.org/techniques/T1059/001/

HOST-004 Windows Firewall Disabled

MEDIUM | Effort: Low | Requires: GPO Admin

DESCRIPTION

Windows Defender Firewall provides host-based network filtering that limits lateral movement even after network perimeter breach. When disabled, all ports and services on the host are exposed to the internal network.

IMPACT

Without host-based firewalling, any service running on a workstation is accessible from the network. This dramatically

increases the attack surface and enables lateral movement techniques like WMI, PsExec, and WinRM access that would otherwise be blocked.

REMEDIATION STEPS

Step 1: Enable Windows Firewall via GPO

Enable for all profiles

```
Computer Configuration > Policies > Windows Settings > Security Settings >
Windows Defender Firewall with Advanced Security:
  Domain Profile: Firewall state -> On
  Private Profile: Firewall state -> On
  Public Profile: Firewall state -> On
```

PowerShell -- enable immediately

```
Set-NetFirewallProfile -Profile Domain,Private,Public -Enabled True
```

Step 2: Configure baseline rules

Block inbound by default, allow specific management traffic

```
# Allow RDP only from admin subnet:
New-NetFirewallRule -DisplayName "Allow RDP from Admin Subnet" `
  -Direction Inbound -Protocol TCP -LocalPort 3389 `
  -RemoteAddress 10.0.1.0/24 -Action Allow
# Block all other inbound RDP:
New-NetFirewallRule -DisplayName "Block RDP from Other" `
  -Direction Inbound -Protocol TCP -LocalPort 3389 -Action Block
```

GOTCHAS

- ! Third-party firewalls may conflict with Windows Firewall -- typically only one should be active
- ! Some applications require specific inbound rules -- audit running applications before enabling
- ! Test firewall rules in a pilot group to identify applications that break
- ! Domain Profile applies when connected to domain network -- ensure this is the primary profile on workstations

VERIFICATION

Verify firewall is enabled

```
Get-NetFirewallProfile | Select-Object Name, Enabled | Format-Table
```

REFERENCES

https://attack.mitre.org/techniques/T1562/004/

HOST-005 Outdated or End-of-Life Operating Systems

CRITICAL | Effort: High | Requires: IT Management, System Admin

DESCRIPTION

Systems running end-of-life operating systems (Windows 7, Server 2008/2012, etc.) no longer receive security updates and are vulnerable to known, publicly exploited vulnerabilities including EternalBlue (MS17-010), BlueKeep (CVE-2019-0708), and many others.

IMPACT

End-of-life systems can be compromised using publicly available exploits that require no authentication. A single unpatched system provides reliable initial access and can serve as a pivot point for attacking the rest of the network.

REMEDIATION STEPS

Step 1: Inventory end-of-life systems

PowerShell -- find all domain computers with OS version

```
Get-ADComputer -Filter * -Properties OperatingSystem,OperatingSystemVersion,LastLogonDate |
  Select-Object Name, OperatingSystem, OperatingSystemVersion, LastLogonDate |
  Sort-Object OperatingSystem |
  Export-Csv -Path "OS_Inventory.csv" -NoTypeInformation
```

Step 2: Plan and execute upgrades

Priority order for upgrades

```
# 1. Internet-facing systems (highest priority)
# 2. Domain Controllers and infrastructure servers
# 3. Systems with sensitive data (file servers, databases)
# 4. Workstations
# 5. Isolated lab/test systems (lowest priority)
```

Step 3: Isolate systems that cannot be immediately upgraded

Network segmentation for legacy systems

```
# Place legacy systems in a dedicated VLAN with restrictive ACLs:
# - Block all inbound access except required management ports
# - Block outbound internet access
# - Log all traffic to/from the legacy VLAN
# - Apply additional monitoring (IDS/IPS signatures for known exploits)
```

Note: Isolation is a compensating control, not a fix -- track these for upgrade

GOTCHAS

- ! Some LOB applications only run on legacy OS -- engage application vendors for upgrade paths
- ! Extended Security Updates (ESU) are available for some EOL systems at additional cost
- ! OT/ICS environments may have systems that cannot be upgraded -- isolate and monitor
- ! Disable unnecessary services on legacy systems to reduce attack surface (especially SMBv1)

VERIFICATION

Re-run inventory to confirm no EOL systems remain in production

```
Get-ADComputer -Filter {OperatingSystem -like "*2003*" -or OperatingSystem -like "*2008*" -or Ope...
  Select-Object Name, OperatingSystem
```

REFERENCES

https://attack.mitre.org/techniques/T1190/

HOST-006 Unrestricted Outbound Internet Access

MEDIUM | Effort: High | Requires: Network Admin, Firewall Admin

DESCRIPTION

When internal systems have unrestricted outbound internet access, attackers can easily exfiltrate data, download additional tools, establish command and control (C2) channels, and pivot through the network without restriction.

IMPACT

Unrestricted outbound access enables attackers to maintain persistent access via C2 channels, exfiltrate sensitive data, and

download additional attack tools -- all critical phases of a real-world attack.

REMEDIATION STEPS

Step 1: Implement outbound firewall rules

Default deny outbound with explicit allowlists

```
# At the network perimeter firewall:
# 1. Block all outbound traffic by default
# 2. Allow specific ports/protocols:
#    - HTTPS (443) through web proxy only
#    - DNS (53) to internal DNS servers only
#    - NTP (123) to internal NTP servers only
# 3. Block direct internet access from servers and workstations
```

Step 2: Deploy a web proxy for HTTP/HTTPS

Force all web traffic through an authenticated proxy

```
# Configure proxy via GPO:
User Configuration > Preferences > Windows Settings > Registry:
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings:
  ProxyEnable = 1
  ProxyServer = proxy.domain.local:8080
# Or deploy PAC file via WPAD DNS entry
```

Step 3: Implement DNS filtering

Block direct DNS and force internal resolution

```
# Firewall rule: Block all outbound UDP/TCP 53 except from DNS servers
# Configure DNS servers to forward to filtered upstream (Umbrella, Quad9, etc.)
# Monitor for DNS tunneling patterns
```

GOTCHAS

- ! Many cloud services require HTTPS outbound -- build allowlists carefully
- ! Certificate pinning in some applications may conflict with SSL inspection proxies
- ! Blocking outbound may break Windows Update -- configure WSUS or direct update paths
- ! DevOps teams need outbound for package managers (npm, pip, NuGet) -- provide proxy access

VERIFICATION

Test outbound connectivity from a workstation

```
# From a standard workstation, these should be blocked:
Test-NetConnection -ComputerName 8.8.8.8 -Port 53       # Direct DNS  --  should fail
Test-NetConnection -ComputerName evil.com -Port 443      # Direct HTTPS  --  should fail
# Only proxy-based access should work
```

REFERENCES

https://attack.mitre.org/techniques/T1048/

SSL/TLS AND WEB

WEB-001 SSL/TLS Weak Configurations

MEDIUM | Effort: Medium | Requires: System Admin, Application Owner

DESCRIPTION

Weak SSL/TLS configurations including support for deprecated protocols (SSLv3, TLS 1.0, TLS 1.1), weak cipher suites (RC4, DES, 3DES, NULL), and short key lengths expose encrypted communications to downgrade attacks and potential decryption.

IMPACT

Attackers can exploit weak TLS configurations to downgrade connections, intercept encrypted traffic, or perform padding oracle attacks to decrypt sensitive data in transit.

REMEDIATION STEPS

Step 1: Disable legacy protocols

Disable SSLv3, TLS 1.0, and TLS 1.1 via registry

```
# Disable TLS 1.0
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Serve...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Serve...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Clien...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Clien...
# Disable TLS 1.1
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Serve...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Serve...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Clien...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Clien...
# Disable SSLv3
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 3.0\Serve...
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 3.0\Serve...
```

Or use IISCrypto tool for a GUI-based approach

```
# Download IISCrypto from Nartac Software  --  use 'Best Practices' template
```

Step 2: Configure strong cipher suites

GPO -- set cipher suite order

```
Computer Configuration > Administrative Templates > Network >
SSL Configuration Settings > SSL Cipher Suite Order -> Enabled
# Recommended order (TLS 1.2+):
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
```

Step 3: Enable TLS 1.3 where supported

Windows Server 2022+ and Windows 11 support TLS 1.3 natively

```
reg add "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.3\Serve...
```

GOTCHAS

! Disabling TLS 1.0 may break connectivity to legacy systems -- inventory before disabling

! SQL Server 2008-2014 may require TLS 1.0 -- check vendor support for TLS 1.2
! Some payment processing integrations require specific TLS versions -- verify with vendors
! .NET applications may need registry settings to use TLS 1.2: SchUseStrongCrypto

VERIFICATION

Test SSL/TLS configuration with nmap

```
nmap --script ssl-enum-ciphers -p 443 target.domain.local
```

Test with testssl.sh

```
testssl.sh --protocols --ciphers target.domain.local:443
```

REFERENCES

https://learn.microsoft.com/en-us/windows-server/security/tls/tls-registry-settings

WEB-002 Default Web Application Pages and Headers

LOW | Effort: Low | Requires: System Admin, Application Owner

DESCRIPTION

Default web server pages (IIS welcome page, Apache test page, Tomcat default application) and verbose HTTP headers (Server, X-Powered-By, X-AspNet-Version) reveal technology stack information that helps attackers identify specific exploits to use.

IMPACT

Information disclosure through default pages and headers enables attackers to fingerprint the technology stack and target version-specific vulnerabilities, reducing the time needed for reconnaissance.

REMEDIATION STEPS

Step 1: Remove default pages

IIS -- remove default page

```
# Remove the default IIS page:
Remove-Item "C:\inetpub\wwwroot\iisstart.htm" -Force
Remove-Item "C:\inetpub\wwwroot\iisstart.png" -Force
```

Apache -- disable default site

```
a2dissite 000-default && systemctl reload apache2
```

Tomcat -- remove default applications

```
rm -rf /opt/tomcat/webapps/ROOT
rm -rf /opt/tomcat/webapps/docs
rm -rf /opt/tomcat/webapps/examples
rm -rf /opt/tomcat/webapps/manager      # Or restrict access
rm -rf /opt/tomcat/webapps/host-manager  # Or restrict access
```

Step 2: Remove version headers

IIS -- remove Server header

```
# web.config:
# <system.webServer>
#   <security>
#     <requestFiltering removeServerHeader="true" />
#   </security>
#   <httpProtocol>
#     <customHeaders>
#       <remove name="X-Powered-By" />
#     </customHeaders>
#   </httpProtocol>
# </system.webServer>
Import-Module WebAdministration
Set-WebConfigurationProperty -Filter "system.webServer/httpProtocol/customHeaders" `
  -PSPath "IIS:\Sites\Default Web Site" -Name "." `
  -Value @{name='X-Powered-By'; value=''}
```

Nginx -- hide version

```
# In nginx.conf: server_tokens off;
```

Apache -- hide version

```
# In httpd.conf or apache2.conf:
ServerTokens Prod
ServerSignature Off
```

GOTCHAS

- ! Some load balancers add their own Server headers -- check at the load balancer level too
- ! WAF/CDN services may expose version information -- configure at the CDN level
- ! Custom error pages should also avoid revealing technology details

VERIFICATION

Check HTTP headers

```
curl -sI https://target.domain.local | grep -iE 'server|x-powered|x-aspnet'
```

REFERENCES

https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/01-Information_Gathering/02-Fingerprint_Web_Server