



Less Praying More Relaying

Detecting NTLM relay mitigations for ~~MSSQL~~ multiple protocols



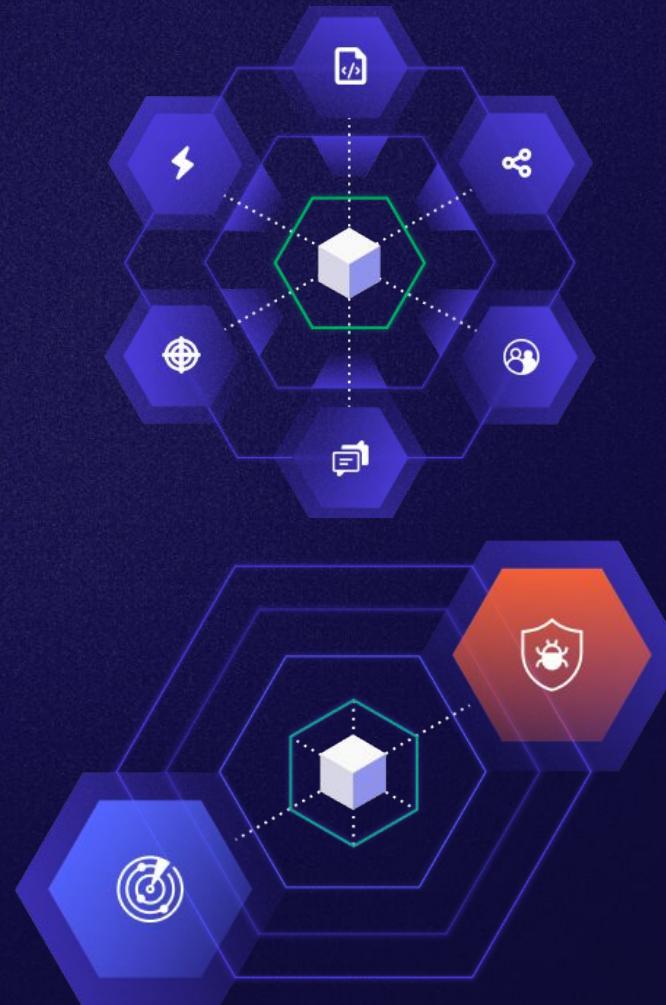
Intros

- **Nick Powers**
 - Service Architect, Adversary Simulation
 - Interested in NTLM relay tradecraft, initial access techniques, and payload development
 - **@zyn3rgy**
- **Matt Creel**
 - Senior Consultant, Adversary Simulation
 - Interested in post-exploitation tradecraft and tool development
 - **@Tw1sm**

The “What” and “Why”

What are we covering?

- Extended Protection for Authentication (EPA) is an increasingly common integrity protection that can mitigate popular authentication relay attack primitives
- We will demonstrate new ways to determine the enforcement state of this integrity protection for arbitrary target services, from the attacker’s perspective

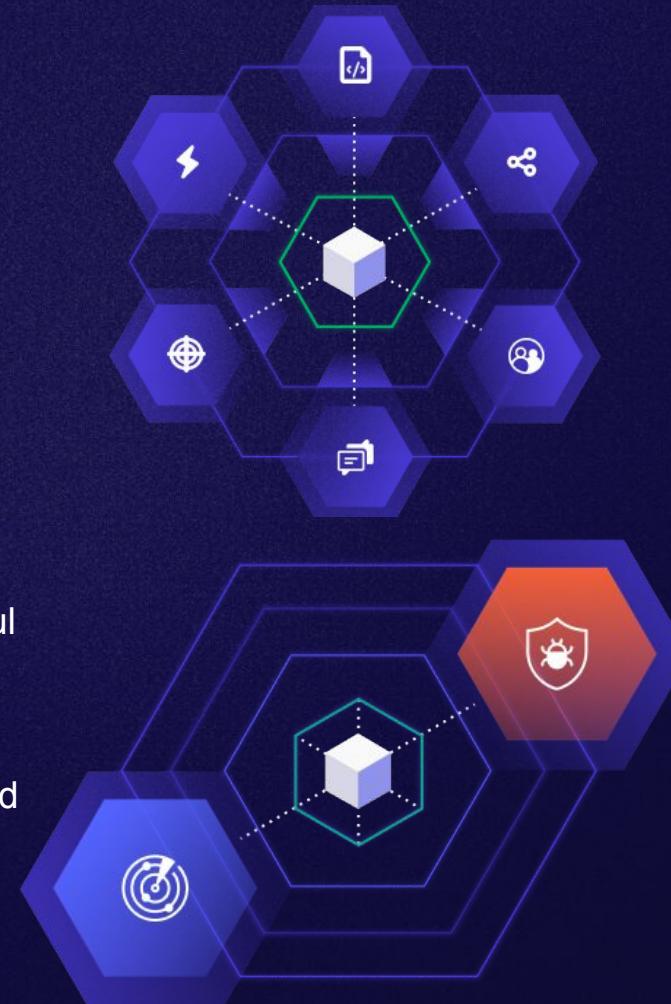


The “What” and “Why”

Why are we covering it?

- Mature orgs have had SMB signing enforced for years
- Popular relays targeting LDAP have thrust LDAP protections into the spotlight
 - *KrbRelayUp* style local privilege escalation
 - Targeted, remote WebClient abuse
- Many orgs have addressed ADCS ESC8
 - Web enrollment service removal
 - EPA
- Misconfiguration Manager TAKEOVER-1 (MSSQL) is still often successful
 - Microsoft added EPA support to SCCM in Dec 2024
- Significant overhead for relay attacks through C2
 - Can we determine the protections in place before lab simulation and attack setup?

This content and subsequent tool release will address that by better informing your relay decisions



Agenda

- EPA Background and Fundamentals
- Server EPA Implementation Nuances
- Client Error-Based Enumeration Strategy
- Tool Releases
- Conclusion and Key Takeaways

EPA Background and Fundamentals

EPA Background and Fundamentals

Introduction and Basics

- Extended Protection for Authentication (EPA)
 - Security feature to protect the integrity of authentication mechanisms such as NTLM and Kerberos
 - Does this by **binding** the authentication to the intended **channel** and/or **service**
 - Focus for this presentation will be NTLM
 - Configured in one of three enforcement levels
 - Disabled
 - Allowed / Accepted / When Supported
 - Required
- Focus for this talk will be specific to NTLM(v2) implementation
 - And “interesting” server implementation design decisions (more on this later)

EPA Background and Fundamentals

Introduction and Basics

- EPA consists of *channel binding and service binding*
 - Common confusion that EPA = channel binding (not the case)
 - Determined by the server's implementation exactly when to check what
- Where do these “bindings” exist?
 - NTLM negotiation consists of three messages, and the values that are validated for EPA are sent in the third message from client → server
 - Server validates based on AV pair contents within NTLM Type 3 / NTLM_AUTH
 - Channel binding - **MsvAvChannelBindings**
 - Service binding - **MsvAvTargetName**

EPA Background and Fundamentals

Enforcement Settings (Server Expectations)



Disabled

Server ignores presence, or lack thereof, the **MsvAvTargetName** and **MsvAvChannelBindings** AV pairs



Allowed / Accepted / When Supported

Server accepts auth where the relevant AV pair is not present

Server treats the mere presence of the relevant AV pair as proof the client is aware of EPA and will enforce validity

Windows NTLM SSP will always include these AV pairs



Required

Server requires the relevant channel binding or service binding AV pair to be present and valid

EPA Background and Fundamentals

Enforcement Settings (Common Client Behavior)

- Common NTLM relays begin poisoning or coercing traffic
 - broadcast / multicast poisoning (LLMNR, mDNS, DHCPv6, etc)
 - authentication coercion (MS-EFSR [petitpotam], [MS-RPRN] printerbug, etc)
- Modern Windows SSPI (NTLM SSP) includes both relevant av pairs by default
 - Both options above mostly commonly result in the Windows constructing the NTLMv2 client blob
 - Makes sense for **MsvAvTargetName**, target name often appears in NTLMv2 clients
 - Importantly though, **MsvAvChannelBindings** will **always appear** regardless of the target protocol's support for channel binding

EPA Background and Fundamentals

Enforcement Settings (Common Client Behavior)

Wireshark · Packet 123 · Ethernet

```
> Attribute: DNS computer name: castelblack.north.sevenkingdoms.]▲
> Attribute: DNS tree name: sevenkingdoms.local
> Attribute: Timestamp
> Attribute: Flags
> Attribute: Restrictions
▼ Attribute: Channel Bindings
    NTLMV2 Response Item Type: Channel Bindings (0x000a)
    NTLMV2 Response Item Length: 16
    Channel Bindings: 00000000000000000000000000000000
> Attribute: Target Name: cifs/10.8.10.22
> Attribute: End of list
```

SMB(v2) / HTTP (WebDav)

Server EPA Implementation Nuances

Server EPA Implementation Nuances

What We Would Expect

- EPA can protect both encrypted and unencrypted connections
 - Encrypted connection
 - Channel binding - calculated by some aspect of TLS negotiation
 - tls-server-end-point (IIS), tls-unique (MSSQL), tls-unique-for-telnet, tls-exporter
 - Unencrypted connection
 - Service binding - included by majority of NTLMv2 clients and determined by the client
- When this feature is enabled, there should be coverage for both by default... right?

Wrong

Server EPA Implementation Nuances

MSSQL



MSSQL - Encrypted

When EPA is set to *allowed* or *required* the channel bindings AV pair value is validated and the target name AV pair is ignored

```
▼ Attribute: Channel Bindings
  NTLMV2 Response Item Type: Channel Bindings (0x000a)
  NTLMV2 Response Item Length: 16
  Channel Bindings: 996fc2637144aeacd4435b09bc1619db
```



MSSQL - Unencrypted

When EPA is set to *allowed* or *required* the target name AV pair value is validated and the channel bindings AV pair is ignored

```
▼ Attribute: Target Name: MSSQLSvc/castelblack.north.sevenkingdoms.local:1433
  NTLMV2 Response Item Type: Target Name (0x0009)
  NTLMV2 Response Item Length: 102
  Target Name: MSSQLSvc/castelblack.north.sevenkingdoms.local:1433
```

Server EPA Implementation Nuances

Visual

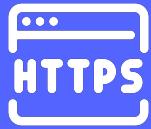
		MSSQL	
		EPA On	EPA Off
Encrypted	CB	None	
Unencrypted	SB	None	

CB = Channel Binding

SB = Service Binding

Server EPA Implementation Nuances

IIS



IIS - Encrypted / HTTPS

EPA set to *accept* or *required* through the IIS GUI will enable channel binding for HTTPS



IIS - Unencrypted / HTTP

EPA set through the IIS GUI *DOES NOT* enable any protections for unencrypted connections

Service binding can be configured (instead of channel binding) manually in the server's *applicationHost.config* file

“A Study on Windows HTTP Authentication” by Pierre Milioni & Geoffrey Bertoli from Synacktiv

Server EPA Implementation Nuances

Visual

	MSSQL		IIS HTTP	
	EPA On	EPA Off	EPA On	EPA Off
Encrypted	CB	None	CB	None
Unencrypted	SB	None	None	None

CB = Channel Binding SB = Service Binding

Server EPA Implementation Nuances

Visual

MSSQL		IIS HTTP		IIS HTTP (Manual SB Config)		
	EPA On	EPA Off	EPA On	EPA Off	EPA On	EPA Off
Encrypted	CB	None	CB	None	SB	None
Unencrypted	SB	None	None	None	SB	None

CB = Channel Binding

SB = Service Binding

Internet Information Services (IIS) Manager

BRAAVOS > Sites > Default Web Site > CertSrv

File View Help

Connections

- Start Page
- BRAAVOS (BRAAVOS\localuse)
 - Application Pools
 - Sites
 - Default Web Site
 - aspnet_client
 - CertEnroll
 - CertSrv

Authentication

Group by: No Grouping

Name	Status	Advanced Settings
Anonymous Authentication	Disabled	
ASP.NET Impersonation	Disabled	
Forms Authentication	Disabled	
Windows Authentication	Enabled	Extended Protection: Required Click here for more information online

Alerts

Click here to learn how to configure Extended Protection.

Capturing from Ethernet

Capturing from Ethernet

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port == 80

No.	Time	Source	Destination	Protocol	Length	Info
10742	875.31...	10.8.10.23	10.8.10.254	TCP	66	80 → 54432 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
10743	875.31...	10.8.10.254	10.8.10.23	TCP	60	54430 → 80 [ACK] Seq=1105 Ack=6275 Win=65280 Len=0
10744	875.31...	10.8.10.254	10.8.10.23	TCP	60	54432 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
10745	875.31...	10.8.10.254	10.8.10.23	HTTP	203	GET /certsrv/ HTTP/1.1
10746	875.31...	10.8.10.23	10.8.10.254	HTTP	1564	HTTP/1.1 401 Unauthorized (text/html)
10747	875.31...	10.8.10.254	10.8.10.23	TCP	60	54432 → 80 [ACK] Seq=150 Ack=1511 Win=65280 Len=0
10748	875.31...	10.8.10.254	10.8.10.23	HTTP	281	GET /certsrv/ HTTP/1.1 , NTLMSSP_NEGOTIATE
10749	875.32...	10.8.10.23	10.8.10.254	HTTP	861	HTTP/1.1 401 Unauthorized , NTLMSSP_CHALLENGE (text/html)
10750	875.32...	10.8.10.254	10.8.10.23	HTTP	781	GET /certsrv/ HTTP/1.1 , NTLMSSP_AUTH, User: ESSOS\daenerys.targaryen
10793	875.33...	10.8.10.23	10.8.10.254	TCP	54	80 → 54432 [ACK] Seq=2318 Ack=1104 Win=2102016 Len=0
11274	875.40...	10.8.10.23	10.8.10.254	HTTP	4010	HTTP/1.1 200 OK (text/html)

> Attribute: DNS computer name: braavos.essos.local
 > Attribute: DNS tree name: essos.local
 > Attribute: Timestamp
> Attribute: Target Name: blah/blah
 > Attribute: Flags

Frame (781 bytes) NTLMSSP / GSSAPI Data (416 bytes)

Packets: 11389 · Displayed: 240 (2.1%) · Profile: Default

Text item (text), 22 bytes

Server EPA Implementation Nuances

Implementation Takeaways

- Servers can take unexpected liberties with how they implement EPA
- Understanding how a server implements this protection is important
 - What to manipulate when enumerating EPA enforcement
 - What is possible regarding NTLM relays
- Server implementations often result in unique responses for EPA validation failure
 - Manipulating the relevant values during authentication can help us enumerate enforcement and identify potential gaps

Client Error-Based Enumeration Strategy

Client Error-Based Enumeration

Where It Started

49 9.1... 10.0.0.21 10.0.0.19 TCP 60 62970 → 636 [ACK] Seq=751 Ack=2111 Win=2101760 Len=0 74 9.3... 10.0.0.21 10.0.0.19 LDAP 459 bindRequest(6) "<ROOT>" , NTLMSSP_AUTH, User: citadel.lab\guest 75 9.3... 10.0.0.19 10.0.0.21 LDAP 235 bindResponse(6) invalidCredentials (80090346: LdapErr: DSID-0C0906B0, comment: AcceptSecurityCont 77 9.3... 10.0.0.21 10.0.0.19 TCP 60 62970 → 636 [ACK] Seq=1156 Ack=2292 Win=2101504 Len=0 78 15... 10.0.0.21 10.0.0.19 TCP 60 62970 → 636 [ACK] Seq=1156 Ack=2292 Win=2101504 Len=0	> Frame 75: 235 bytes on wire (1880 bits), 235 bytes captured (1880 bits) on interface \Device\NPF_{EFE0C3FA-A6D3-4D7C-8C16-94281295016F}, id 0 > Ethernet II, Src: VMware_39:69:36 (00:0c:29:39:69:36), Dst: VMware 0d:50:c4 (00:0c:29:0d:50:c4) > Internet Protocol Version 4, Src: 10.0.0.19, Dst: 10.0.0.21 > Transmission Control Protocol, Src Port: 636, Dst Port: 62970, Invalid credentials supplied w/ channel binding enforced and LDAP client not sending CBT > Transport Layer Security > Lightweight Directory Access Protocol LDAPMessage bindResponse(6) invalidCredentials (80090346: LdapErr: DSID-0C0906B0, comment: AcceptSecurityContext error, data 80090346, v4f7c) messageID: 6 protocolOp: bindResponse (1) bindResponse resultCode: invalidCredentials (49) matchedDN: errorMessage: 80090346: LdapErr: DSID-0C0906B0, comment: AcceptSecurityContext error, data 80090346 v4f7c [Response To: 74]
COMPARED TO...	
98 10... 10.0.0.21 10.0.0.19 TCP 60 61713 → 636 [ACK] Seq=751 Ack=2111 Win=2101760 Len=0 99 10... 10.0.0.21 10.0.0.19 LDAP 459 bindRequest(9) "<ROOT>" , NTLMSSP_AUTH, User: citadel.lab\guest 100 10... 10.0.0.19 10.0.0.21 LDAP 219 bindResponse(9) invalidCredentials (8009030C: LdapErr: DSID-0C0906B0, comment: AcceptSecurityCont 101 10... 10.0.0.21 10.0.0.19 TCP 60 61713 → 636 [ACK] Seq=751 Ack=2111 Win=2101760 Len=0	> Frame 100: 219 bytes on wire (1752 bits), 219 bytes captured (1752 bits) on interface \Device\NPF_{EFE0C3FA-A6D3-4D7C-8C16-94281295016F}, id 0 > Ethernet II, Src: VMware_39:69:36 (00:0c:29:39:69:36), Dst: VMware 0d:50:c4 (00:0c:29:0d:50:c4) > Internet Protocol Version 4, Src: 10.0.0.19, Dst: 10.0.0.21 > Transmission Control Protocol, Src Port: 636, Dst Port: 61713 > Transport Layer Security > Lightweight Directory Access Protocol LDAPMessage bindResponse(9) invalidCredentials (8009030C: LdapErr: DSID-0C0906B0, comment: AcceptSecurityContext error, data 52e, v4f7c) messageID: 9 protocolOp: bindResponse (1) bindResponse resultCode: invalidCredentials (49) matchedDN: errorMessage: 8009030C: LdapErr: DSID-0C0906B0, comment: AcceptSecurityContext error, data 52e v4f7c [Response To: 99] [Time: 0.001678000 seconds]

Client Error-Based Enumeration

Same Problem, New Protocols

- Control of the client gives us many variables to manipulate in search of unique responses
 - Manipulation of EPA relevant AV pair
 - In some cases, removal of AV pairs entirely
- Any difference in a response, specific to EPA enforcement, will suffice
 - Data in response
 - Error codes
 - Response codes
 - Timing of responses

Client Error-Based Enumeration

MSSQL Analysis

- MSSQL servers can support unencrypted and encrypted authentication
 - **Force Encryption** can be configured to only allow encrypted connections
- Encryption will determine what type of EPA is validated
 - Unencrypted (if allowed) = service binding
 - Encrypted = channel binding
- How do we ensure this is the case?

Client Error-Based Enumeration

MSSQL Analysis

33 9.907751 10.5.10.22 10.5.10.31 TDS 361 Response, NTLMSSP_CHALLENGE	34 9.908259 10.5.10.31 10.5.10.22 TDS 718 SSPI message, NTLMSSP_AUTH, User: SEVENKINGDOMS\jaime.lannister	37 9.931696 10.5.10.22 10.5.10.31 TCP 54 1433 → 50429 [ACK] Seq=1440 Ack=1558 Win=2102272 L	55 10.128964 10.5.10.22 10.5.10.31 TDS 238 Response
<p>Token - Error</p> <p>Token length: 164</p> <p>SQL Error Number: 18456</p> <p>State: 1</p> <p>Class (Severity): 14</p> <p>Error message length: 54 characters</p> <p>Error message: Login failed for user 'SEVENKINGDOMS\jaime.lannister'.</p> <p>Server name length: 22 characters</p> <p>Server name: CASTELBLACK\SQLEXPRESS</p> <p>Process name length: 0 characters</p> <p>Line number: 1</p> <p>Token - Done</p>			
709 205.335096 10.5.10.22 10.5.10.31 TDS 361 Response, NTLMSSP_CHALLENGE	710 205.335578 10.5.10.31 10.5.10.22 TDS 662 SSPI message, NTLMSSP_AUTH, User: SEVENKINGDOMS\jaime.lannister	725 205.340382 10.5.10.22 10.5.10.31 TDS 334 Response	
<p>Response to invalid target name / CBT when EPA is off (valid domain credentials)</p> <p>0000 bc 24 11 52 85 5a b 0010 00 e0 76 21 40 00 8 0020 0a 1f 05 99 c4 fd 7 0030 20 14 29 11 00 00 0 0040 00 18 48 00 00 01 0 0050 00 6e 00 20 00 66 0 0060 00 20 00 66 00 6f 0 0070 00 72 00 20 00 27 0 0080 00 4b 00 49 00 4e 0 0090 00 5c 00 6a 00 61 0</p>			
<p>▼ Token - Error</p> <p>Token length: 260</p> <p>SQL Error Number: 18452</p> <p>State: 1</p> <p>Class (Severity): 14</p> <p>Error message length: 102 characters</p> <p>Error message: Login failed. The login is from an untrusted domain and cannot be used with Integrated authentication.</p> <p>Server name length: 22 characters</p> <p>Server name: CASTELBLACK\SQLEXPRESS</p> <p>Process name length: 0 characters</p> <p>Line number: 1</p> <p>Token - Done</p>			
<p>Response to invalid target name / CBT when EPA is enabled (valid domain credentials)</p>			

Client Error-Based Enumeration

MSSQL Analysis

- MSSQL EPA enumeration requirements
 - Any valid NTLM authentication for the domain
- MSSQL EPA enumeration logic
 - (Optionally) Validate credentials
 - Send TDS pre-login message to determine encryption requirement
 - If encryption is required
 - Send authentication with an invalid channel bindings AV pair value
 - Send authentication without a channel bindings AV pair
 - If encryption is not required
 - Send authentication with an invalid target name AV pair value
 - Send authentication without a target name AV pair

Client Error-Based Enumeration

HTTP Analysis

- HTTP/S EPA enumeration requirements
 - Valid NTLM authentication *that results in 200 response code*
- HTTP/S EPA enumeration logic
 - Prerequisite check with valid channel binding and service binding where applicable
 - Validate a 200 response code is received
 - If HTTPS, authenticate with:
 - **invalid channel binding** value (and valid target name)
 - **stripped channel binding** AV pair (and valid target name)
 - **stripped target name** AV pair (and correct channel binding)
 - **invalid target name** (and correct channel binding)
 - If HTTP, authenticate with:
 - **stripped target name** AV pair (and correct channel binding)
 - **invalid target name** (and correct channel binding)
- Counterintuitive logic for HTTPS used
 - IIS observations
 - Coverage across multiple server implementations

Client Error-Based Enumeration

HTTPS Analysis

valid credential, correct CBT

HTTP 267 GET /certsrv/ HTTP/1.1	HTTP 1627 HTTP/1.1 401 Unauthorized (text/html)	TCP 60 52499 → 443 [ACK] Seq=697 Ack=2613 Win=65280 Len=0
HTTP 347 GET /certsvr/ HTTP/1.1 , NTLMSSP_AUTH, User: ESSOS\daenerys.targaryen	HTTP 923 HTTP/1.1 401	
HTTP 891 GET /certsrv/ HTTP/1.1 , NTLMSSP_AUTH, User: ESSOS\daenerys.targaryen	TCP 54 443 → 52499 [ACK] Seq=3482 Ack=1827 Win=213040 Len=0	
HTTP 4075 HTTP/1.1 200 OK (text/html)	TCP 60 52499 → 443 [ACK] Seq=1827 Ack=5962 Win=65280 Len=0	
TCP 60 52499 → 443 [ACK] Seq=1827 Ack=7503 Win=65280 Len=0		

NTLMV2 Response Item Length: 8
Timestamp: Aug 22, 2025 19:27:37.54714600 Eastern Daylight

Attribute: Channel Bindings
NTLMV2 Response Item Type: Channel Bindings (0x000a)
NTLMV2 Response Item Length: 16
Channel Bindings: 0a724d360ce1240dff579471a9d2ed7a

Attribute: Target Name: http://10.8.10.23
NTLMV2 Response Item Type: Target Name (0x0009)
NTLMV2 Response Item Length: 30

NTLMV2 Response Item Length: 13

NTLMV2 Response Item Length: 13

** EPA allowed / required

valid credential, incorrect CBT value

HTTP 267 GET /certsrv/ HTTP/1.1	HTTP 1627 HTTP/1.1 401 Unauthorized (text/html)	TCP 60 52499 → 443 [ACK] Seq=1827 Ack=5055 Win=65280 Len=0
HTTP 347 GET /certsvr/ HTTP/1.1 , NTLMSSP_CHALLENGE (text/html)	HTTP 923 HTTP/1.1 401 Unauthorized (text/html)	
HTTP 891 GET /certsrv/ HTTP/1.1 , NTLMSSP_AUTH, User: ESSOS\daenerys.targaryen	HTTP 1627 HTTP/1.1 401 Unauthorized (text/html)	
TCP 60 52094 → 443 [ACK] Seq=1827 Ack=5055 Win=65280 Len=0		

DNS Type Name: essos.local

Attribute: Timestamp
NTLMV2 Response Item Type: Timestamp (0x0007)
NTLMV2 Response Item Length: 8
Timestamp: Aug 22, 2025 19:39:02.650355900 Eastern Daylight

Attribute: Channel Bindings
NTLMV2 Response Item Type: Channel Bindings (0x000a)
NTLMV2 Response Item Length: 16
Channel Bindings: eb7cf0b0a637a8a5415c3efc6e29aa99

0120 f3 f0 bd 13 d ^
0130 a8 a5 41 5c 3
0140 74 00 74 00 7
0150 2e 00 31 00 3
0160 02 00 00 00 0
0170 53 00 4f 00 5
0180 72 00 79 00 7
0190 61 00 72 00 7
01a0 4b 00 53 00 5 ^
\$SAPI Data (448 bytes) < >

Client Error-Based Enumeration

IIS Analysis (EPA configured using GUI only)



Internet Information Services (IIS) Manager

BRAAVOS > Sites > Default Web Site > CertSrv

File View Help

Connections

Start Page

BRAAVOS (BRAAVOS\localuse

Application Pools

Sites

Default Web Site

aspnet_client

CertEnroll

CertSrv

Authentication

Group by: No Grouping

Name	Status	Response
Anonymous Authentication	Disabled	
ASP.NET Impersonation	Disabled	
Forms Authentication	Disabled	HTTP 302
Windows Authentication	Enabled	HTTP 401

Advanced Settings

Extended Protection:

Required

[Click here for more information online](#)

Enable Kernel-mode authentication

By default, IIS enables kernel-mode authentication, which may improve authentication performance and prevent authentication problems with

Capturing from Ethernet

tcp.port == 80

No.	Time	Source	Destination	Protocol	Length	Info
2340	6.8651...	10.8.10.254	10.8.10.23	HTTP	212	GET /certsrv/ HTTP/1.1
2341	6.8653...	10.8.10.23	10.8.10.254	HTTP	1564	HTTP/1.1 401 Unauthorized (text/html)
2342	6.8682...	10.8.10.254	10.8.10.23	TCP	60	62717 → 80 [ACK] Seq=159 Ack=1511 Win=652
2343	6.8709...	10.8.10.254	10.8.10.23	HTTP	290	GET /certsrv/ HTTP/1.1 , NTLMSSP_NEGOTIATE
2344	6.8711...	10.8.10.23	10.8.10.254	HTTP	861	HTTP/1.1 401 Unauthorized , NTLMSSP_CHALL
2346	6.8761...	10.8.10.254	10.8.10.23	HTTP	790	GET /certsrv/ HTTP/1.1 , NTLMSSP_AUTH Us
2457	6.8984...	10.8.10.23	10.8.10.254	TCP	54	80 → 62717 [ACK] Seq=2318 Ack=1131 Win=21
2876	6.9483...	10.8.10.23	10.8.10.254	HTTP	4010	HTTP/1.1 200 OK (text/html)

Time: Aug 22, 2025 23:54:29.510234400 UTC
NTLMv2 Client Challenge: 332242c054080361
Z: 00000000
> Attribute: NetBIOS domain name: ESSOS
> Attribute: NetBIOS computer name: BRAAVOS
> Attribute: DNS domain name: essos.local
> Attribute: DNS computer name: braavos.essos.local
> Attribute: DNS tree name: essos.local
> Attribute: Timestamp
> Attribute: Target Name: blah/blah

00f0 6f 00 73 00 2 ▲
0100 05 00 16 00 6
0110 6c 00 6f 00 6
0120 67 19 c0 13 d
0130 68 00 2f 00 6
0140 02 00 00 00 0
0150 53 00 4f 00 5
0160 72 00 79 00 7
0170 61 00 72 00 7
0180 4h 00 53 00 5 ▼
NTLMSSP / GSSAPI

Client Error-Based Enumeration

Key Takeaways

- Understanding *what* and *when* a server validates is important
 - Informed by manipulating **MsvAvChannelBindings** and **MsvAvTargetName** on client-side
 - Any difference in response can help identify EPA enforcement
- **More unauthenticated opportunity for edge cases**
 - MSSQL
 - configured to allow GUEST login → unauthenticated EPA enumeration (error-based)
 - IIS
 - default EPA configuration → unauthenticated EPA enumeration (timing-based)

Tool Releases



~Python and BOF Implementations~

RelayInformer

Python Implementation (MSSQL)

```
[2025-08-23 03:55:35] defaultuser ~/dev/relayinformer
$ uv run relayinformer mssql --target 10.8.10.23 --user 'ESSOS/daenerys.targaryen' --password 'BurnThemAll!'
[03:56:02] INFO    Testing EPA enforcement level for MSSQL service at 10.8.10.23 on port 1433 as ESSOS/daenerys.targaryen
[03:56:02] INFO    Server supports but does not require encryption (TDS_ENCRYPT_OFF)
[03:56:03] INFO    Successfully connected to MSSQL server
[03:56:03] INFO    Prereq check passed with an expected response, continuing
[03:56:03] INFO    Conducting logins while manipulating target service av pair over unencrypted connection
[03:56:03] INFO    -----
[03:56:03] INFO        EPA setting - Required
[03:56:03] INFO    -----
[2025-08-23 03:56:03] defaultuser ~/dev/relayinformer
$ uv run relayinformer mssql --target 10.8.10.23 --user 'ESSOS/daenerys.targaryen' --hashes :34534854d33b398b66684072224bb47a
[03:56:37] INFO    Testing EPA enforcement level for MSSQL service at 10.8.10.23 on port 1433 as ESSOS/daenerys.targaryen
[03:56:37] INFO    Server requires encryption (TDS_ENCRYPT_REQ)
[03:56:37] INFO    Successfully connected to MSSQL server
[03:56:37] INFO    Prereq check passed with an expected response, continuing
[03:56:37] INFO    Conducting logins while manipulating channel binding av pair over encrypted connection
[03:56:37] INFO    Successfully connected to MSSQL server
[03:56:37] INFO    -----
[03:56:37] INFO        EPA setting - Allowed
[03:56:37] INFO    -----
```

RelayInformer

Python Implementation (HTTP)

```
[2025-08-23 03:26:10] defaultuser ~/dev/relayinformer
└ $ uv run relayinformer --debug http --url https://10.8.10.23/certsrv/ --user 'ESSOS\daenerys.targaryen' --password 'BurnThemAll!'
[03:26:14] INFO    Testing EPA enforcement level for HTTPS service at https://10.8.10.23/certsrv/ as ESSOS\daenerys.targaryen
[03:26:14] DEBUG   Running prerequisite check with correct parameters
[03:26:14] DEBUG   Successfully authenticated to https://10.8.10.23/certsrv
[03:26:14] INFO    Prereq check passed with correct av pair values
[03:26:14] DEBUG   EPA checks will manipulate channel binding AV pair over encrypted (HTTPS) connection
[03:26:14] DEBUG   Checking with NO CBT and correct target name
[03:26:14] DEBUG   Authentication failed with 401 Unauthorized
[03:26:14] INFO    EPA (channel binding) is REQUIRED
[03:26:14] DEBUG   Checking with CORRECT CBT and STRIPPED target name (service binding)
[03:26:14] DEBUG   Successfully authenticated to https://10.8.10.23/certsrv
[03:26:15] DEBUG   Service binding not required when target name stripped (200). Proceeding to fake target service check
[03:26:15] DEBUG   Checking with CORRECT CBT and FAKE target service name
[03:26:15] DEBUG   Successfully authenticated to https://10.8.10.23/certsrv
[03:26:15] INFO    EPA (service binding) is DISABLED
[2025-08-23 03:26:15] defaultuser ~/dev/relayinformer
└ $ uv run relayinformer --debug http http://10.8.10.23/certsrv/ --user 'ESSOS\daenerys.targaryen' --password 'BurnThemAll!'
[03:26:48] INFO    Testing EPA enforcement level for HTTP service at http://10.8.10.23/certsrv/ as ESSOS\daenerys.targaryen
[03:26:48] DEBUG   Running prerequisite check with correct parameters
[03:26:48] DEBUG   Successfully authenticated to http://10.8.10.23/certsrv
[03:26:48] INFO    Prereq check passed with correct av pair values
[03:26:48] DEBUG   EPA checks will manipulate "target service" AV pair over unencrypted (HTTP) connection
[03:26:48] DEBUG   HTTP check: STRIPPED target name
[03:26:48] DEBUG   Successfully authenticated to http://10.8.10.23/certsrv
[03:26:48] DEBUG   HTTP check: FAKE target service name
[03:26:49] DEBUG   Successfully authenticated to http://10.8.10.23/certsrv
[03:26:49] INFO    EPA (service binding) is DISABLED
```

RelayInformer

Python Implementation (HTTP)

```
[2025-08-23 03:29:27] defaultuser ~/dev/relayinformer
$ uv run relayinformer http --url https://10.8.10.23/certsrv/ --user 'ESSOS\daenerys.targaryen' --password 'BurnThemAll!'
[03:29:39] INFO      Testing EPA enforcement level for HTTPS service at https://10.8.10.23/certsrv/ as ESSOS\daenerys.targaryen
[03:29:39] INFO      Prereq check passed with correct av pair values
[03:29:39] INFO      EPA (channel binding) is REQUIRED
[03:29:39] INFO      EPA (service binding) is DISABLED
[2025-08-23 03:29:39] defaultuser ~/dev/relayinformer
$ uv run relayinformer http --url http://10.8.10.23/certsrv/ --user 'ESSOS\daenerys.targaryen' --password 'BurnThemAll!'
[03:29:46] INFO      Testing EPA enforcement level for HTTP service at http://10.8.10.23/certsrv/ as ESSOS\daenerys.targaryen
[03:29:46] INFO      Prereq check passed with correct av pair values
[03:29:47] INFO      EPA (service binding) is DISABLED
```

RelayInformer

Python Implementation (LDAP)

```
[2025-08-23 03:49:13] defaultuser ~/dev/relayinformer
└$ uv run relayinformer ldap --method LDAPS --dc-ip 10.8.10.12
[03:49:21] INFO      Identified Domain Controllers

-> meereen.essos.local

[03:49:21] INFO      Checking DCs for LDAP NTLM relay protections
[03:49:21] INFO      [meereen.essos.local] (LDAPS) CHANNEL BINDING SET TO NEVER! PARTY TIME!
[2025-08-23 03:49:21] defaultuser ~/dev/relayinformer
└$ uv run relayinformer ldap --method BOTH --dc-ip 10.8.10.12 --user drogon --password Dracarys
[03:49:29] INFO      Identified Domain Controllers

-> meereen.essos.local

[03:49:29] INFO      Checking DCs for LDAP NTLM relay protections
[03:49:29] INFO      [meereen.essos.local] (LDAP) SERVER SIGNING REQUIREMENTS NOT ENFORCED!
[03:49:29] INFO      [meereen.essos.local] (LDAPS) CHANNEL BINDING SET TO NEVER! PARTY TIME!
```

RelayInformer

BOF Design Considerations

- Existing approaches
 - **LdapSignCheck** - Manually calls AcquireCredentialsHandle/InitializeSecurityContext and feed a credential/context handle to protocol-specific APIs
 - **SharpLdapRelayScan** - Manually builds NTLM messages
 - Requires cleartext password to be supplied
- Our design requirements
 - Needs to work with multiple protocols
 - Ex: SSPI context handle can't be passed directly into a SQLDriverConnect call
 - Needs to be capable of leveraging the current logon session

RelayInformer

BOF Implementation

- Trap *sspicli.dll* function calls using hardware breakpoints
 - *AcquireCredentialsHandle*
 - *InitializeSecurityContext*

Syntax

```
C++ Copy  
  
SECURITY_STATUS SEC_Entry AcquireCredentialsHandle(  
    _In_ SEC_CHAR *pszPrincipal,  
    _In_ SEC_CHAR *pszPackage,  
    _In_ ULONG fCredentialUse,  
    _In_ PLUID pvLogonID,  
    _In_ PVOID pAuthData,  
    _In_ SEC_GET_KEY_FN pGetKeyFn,  
    _In_ PVOID pvGetKeyArgument,  
    _Out_ PCredHandle phCredential,  
    _Out_ PTimeStamp ptsExpiry  
);
```

Syntax

```
C++ Copy  
  
KSECDDECLSPEC SECURITY_STATUS SEC_ENTRY InitializeSecurityContextW(  
    [in, optional] PCredHandle phCredential,  
    [in, optional] PCtxtHandle phContext,  
    [in, optional] PSECURITY_STRING pTargetName,  
    [in] unsigned long fContextReq,  
    [in] unsigned long Reserved1,  
    [in] unsigned long TargetDataRep,  
    [in, optional] PSecBufferDesc pInput,  
    [in] unsigned long Reserved2,  
    [in, out, optional] PCtxtHandle phNewContext,  
    [in, out, optional] PSecBufferDesc pOutput,  
    [out] unsigned long *pfContextAttr,  
    [out, optional] PTimeStamp ptsExpiry  
);
```

RelayInformer

BOF Implementation

- Traps work as a drop-in approach across MSSQL, HTTP/S and LDAPS
 - *WLDAP32!ldap_bind_s, ODBC32!SQLDriverConnect, WINHTTP!WinHttpSendRequest* all filter down to ACH/ISC API calls

```
=====>InitializeSecurityContextW trap<=====  
[ISC] TargetName: HTTP/braavos.essos.local ← Modify parameter to change  
[ISC] fContextReq: 0x00000001 TargetName AV pair  
[ISC]   flags: ISC_REQ_DELEGATE  
[ISC] pInput=000000DA6D1FF078  
[ISC] pInput ver=0 c=2 pBuf=000000DA6D1FF0A8  
[ISC] buf[0]: type=14 cb=85 pv=00000232CC79FFB0  
[ISC]   CB appOff=32 appLen=53  
[ISC]   CB appDataHex:  
[ISC]     buf[1]: type=2 c=14 pBuf=000000DA6D1FF0A8 → Modify type 14 SecBuffer to change  
[ISC]       TargetName: HTTP/braavos.essos.local ChannelBindings AV pair  
[ISC]         flags: ISC_REQ_DELEGATE  
[ISC]           pInput=000000DA6D1FF078  
[ISC]           pInput ver=0 c=2 pBuf=000000DA6D1FF0A8  
[ISC]             buf[0]: type=14 cb=85 pv=00000232CC79FFB0  
[ISC]               CB appOff=32 appLen=53  
[ISC]               CB appDataHex: 746C732D7365727665722D656E642D706F696E743AE73054E70F16D6C6F77A6910584C7029FEF0ECFB1  
[ISC]             buf[1]: type=2 cb=208 pv=00000232CC77CB30
```

RelayInformer

BOF Implementation Nuances

- ODBC32 calls funnel to ANSI versions of ACH/ISC
- Default ODBC “SQL Driver” does not support channel binding
 - Newer ODBC 17/18 drivers support channel binding, but require installation
- WinHTTP uses threading under the hood
 - At least when using the low security autologon policy
 - Requires arming hardware breakpoints on all threads to catch SSPI calls
- Checking LDAP signing requires stripping certain context flags from ISC calls
 - *WLDAP32!ldap_bind_s* allows auth method specification (can force NTLM without trapping ACH)

RelayInformer

BOF Implementation

- BOFs can detect EPA disabled vs enabled
 - Windows SSP always sends the relevant AV pairs!
- Support for EPA/signing across multiple protocols
 - MSSQL
 - HTTP/S
 - LDAP/S
 - SMB

```
[08/22 15:05:43] beacon> http-relay-informer https://braavos.essos.local/certsrv/  
[08/22 15:05:43] [*] Tasked beacon to inform on HTTP/S protections  
[08/22 15:05:47] [+] host called home, sent: 14615 bytes  
[08/22 15:05:48] [+] received output:  
[*] Targeting URL: https://braavos.essos.local/certsrv/  
[*] Zeroed out channel binding token rejected!  
[*] EPA is ACCEPT or REQUIRED (Channel Binding)
```

```
[08/22 15:05:49] beacon> http-relay-informer http://braavos.essos.local/certsrv/  
[08/22 15:05:49] [*] Tasked beacon to inform on HTTP/S protections  
[08/22 15:05:57] [+] host called home, sent: 14613 bytes  
[08/22 15:05:57] [+] received output:  
[*] Targeting URL: http://braavos.essos.local/certsrv/  
[*] EPA is OFF
```

```
[08/22 15:34:10] beacon> mssql-relay-informer castelblack.north.sevenkingdoms.local 1433 master  
[08/22 15:34:10] [*] Tasked beacon to inform on MSSQL protections  
[08/22 15:34:18] [+] host called home, sent: 14290 bytes  
[08/22 15:34:18] [+] received output:  
[*] Target: mssql://castelblack.north.sevenkingdoms.local:1433  
[*] ForceEncryption is ON, checking channel binding...  
[*] EPA is ALLOWED or REQUIRED
```

Conclusion

Conclusion

Key Takeaways

- EPA can now be enumerated for MSSQL and HTTPS, in addition to LDAPS
 - LDAPS - still does not require authentication
 - MSSQL - require any valid credential, no need for DB access (BOF)
 - HTTP - requires credentials that can authenticate (BOF)
- Understanding server implementation of EPA can lead to identifying gaps
 - IIS configuration of EPA in GUI vs config file

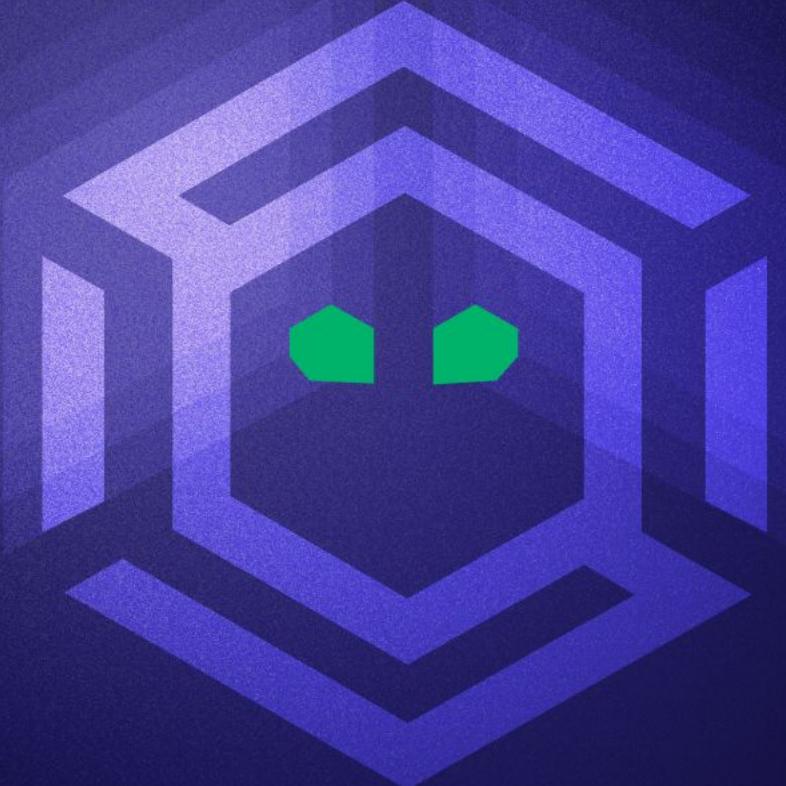
Conclusion

Credit & References

- **Alex DeMine** - initial effort in MSSQL EPA research
- **@Defte_** - “A journey implementation Channel Binding on MSSQLClient.py”
 - <https://sensepost.com/blog/2025/a-journey-implementing-channel-binding-on-mssqlclient.py/>
- **@lowercase_drm** - early open source implementation of LDAP channel binding in LDAP3 library
- **Pierre Milioni and Geoggrey Bertoli** - “A Study on Windows HTTP Authentication (Part 2)”
 - <https://sensepost.com/blog/2025/a-journey-implementing-channel-binding-on-mssqlclient.py/>
- **Adam Crosser** - “Relaying to ADFS Attacks”
 - <https://www.praetorian.com/blog/relaying-to-adfs-attacks/>
- Many open-source developers contributing to libraries such as **Impacket**, **msldap**, **LdapSignCheck**, and more



Questions



Nick Powers | npowers@specterops.io

Matt Creel | mcreel@specterops.io