HITB SECCONF 2023 Phuket

How NTLM Relay Ruins Your Exchange Servers

Tianze Ding (@D1iv3)

Whoami

Tianze Ding (@D1iv3)

Senior Security Researcher, Tencent Security Xuanwu Lab

Focusing on Active Directory Security / Cloud Security / Web Security

2022 MSRC Most Valuable Researchers

DEFCON / Black Hat Asia Speaker

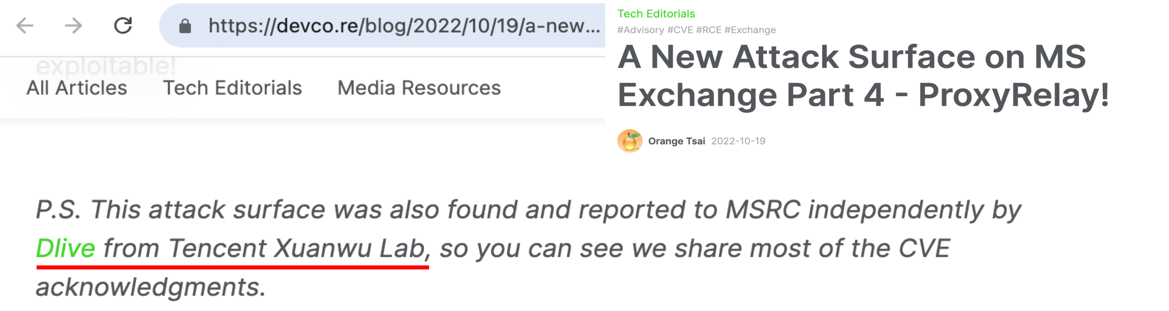
Former CTF Player, 2018 HITB Singapore CTF Final 1st runner-up

腾讯安全玄武实验室
TENCENT SECURITY XUANWU LAB

Vulnerabilities in this talk

Vulnerabilities	Affected Components	Impact
CVE-2021-33768	Exchange Front End	Arbitrary Mailbox Takeover
CVE-2022-21980	Exchange Front End	Arbitrary Mailbox Takeover
CVE-2022-24516	Exchange Front End	Arbitrary Mailbox Takeover
CVE-2022-24477	Exchange Back End	Arbitrary Mailbox Takeover Remote Code Execution
CVE-2021-26414	Windows DCOM	Remote Code Execution
Won’t Fix	Exchange & Active Directory	Privilege Escalation to Domain Admin

Tianze Ding (@D1iv3)



https://devco.re/blog/2022/10/19/a-new...

exploitable!

All Articles Tech Editorials Media Resources

Tech Editorials

#Advisory #CVE #RCE #Exchange

A New Attack Surface on MS Exchange Part 4 - ProxyRelay!

Orange Tsai 2022-10-19

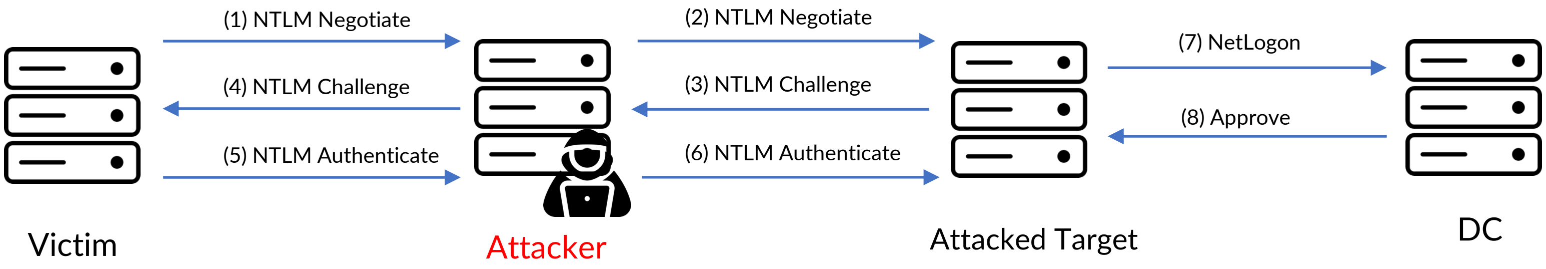
P.S. This attack surface was also found and reported to MSRC independently by Dlive from Tencent Xuanwu Lab, so you can see we share most of the CVE acknowledgments.

Orange Tsai (@orange_8361)

Some of vulnerabilities (CVE-2021-33768/CVE-2022-21979) in this attack surface was also found and reported to MSRC independently by Orange Tsai (@orange_8361) , and named ProxyRelay

NTLM Relay

NTLM Relay is a quite old MITM attack, but still very popular among Active Directory attacks.

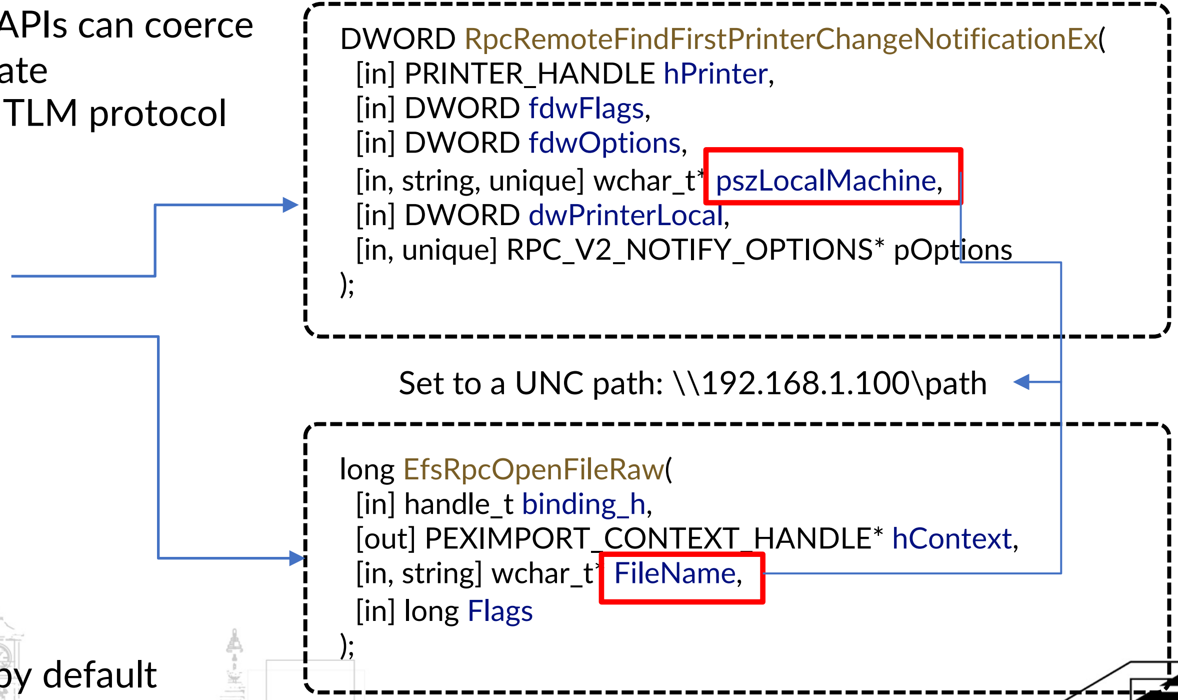


Attackers need to trick/coerce a victim user/computer to authenticate

Coerced Authentication Methods

Many well-known MS-RPC APIs can coerce machine accounts authenticate
to an arbitrary target with NTLM protocol (over SMB)

- MS-PRPNN (PrinterBug)
- MS-EFSR (Petitpotam)
- MS-PAR
- MS-DFSNM
- MS-FSRVP



```
DWORD RpcRemoteFindFirstPrinterChangeNotificationEx(
  [in] PRINTER_HANDLE hPrinter,
  [in] DWORD fdwFlags,
  [in] DWORD fdwOptions,
  [in, string, unique] wchar_t* pszLocalMachine,
  [in] DWORD dwPrinterLocal,
  [in, unique] RPC_V2_NOTIFY_OPTIONS* pOptions
);
```

Set to a UNC path: \\192.168.1.100\path

```
long EfsRpcOpenFileRaw(
  [in] handle_t binding_h,
  [out] PEXIMPORT_CONTEXT_HANDLE* hContext,
  [in, string] wchar_t* FileName,
  [in] long Flags
);
```

Some MS-RPC are enabled by default

Any low-privileged domain users / machine accounts can call these APIs remotely

NTLM Relay

- Authentication
  - Attacked target supports NTLM authentication
  - Relayed credentials need to be considered valid by target services
  - All domain users/computers can authenticate to all AD integration services and authentication will be accepted
- Authorization
  - Relayed credentials need to have special privileges to perform attacks, e.g., local administrator privileges
  - In most cases, low-privileged credentials (regular domain users / machine accounts) can’t do much harm

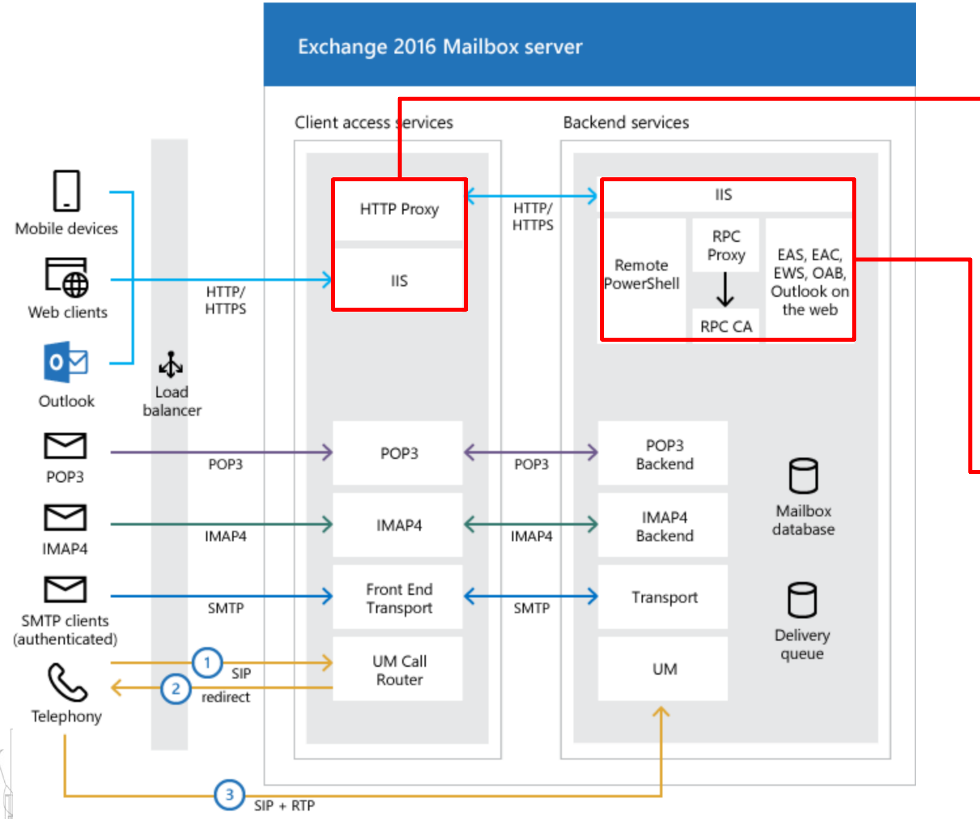
NTLM Relay Mitigations

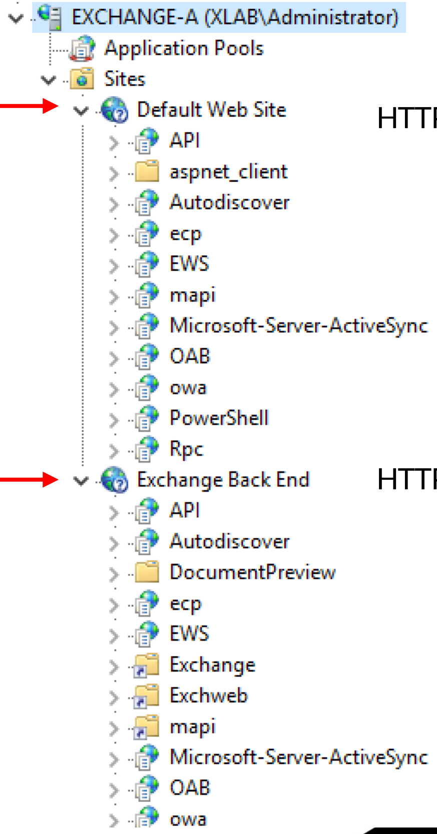
- NTLM reflection protection
- SMB Signing
- LDAP Signing
- EPA (Extended Protection for Authentication)
    - Channel Binding
    - Service Binding
- Signing / Sealing restrictions implemented by individual services
    - Some MS-RPC Services

Why Exchange Server

- Exchange Server is the world's most famous enterprise mail solution
- Highly integrated with Active Directory
  - support AD authentication methods , NTLM/Kerberos
  - some Exchange users and groups have powerful privileges in Active Directory
  - ideal entry point for attackers to break Active Directory
- Complex implementation, software and network architecture

Architecture Overview





- EXCHANGE-A (XLAB\Administrator)
  - Application Pools
  - Sites
    - Default Web Site
      - API
      - aspnet_client
      - Autodiscover
      - ecp
      - EWS
      - mapi
      - Microsoft-Server-ActiveSync
      - OAB
      - owa
      - PowerShell
      - Rpc
    - Exchange Back End
      - API
      - Autodiscover
      - DocumentPreview
      - ecp
      - EWS
      - Exchange
      - Exchweb
      - mapi
      - Microsoft-Server-ActiveSync
      - OAB
      - owa

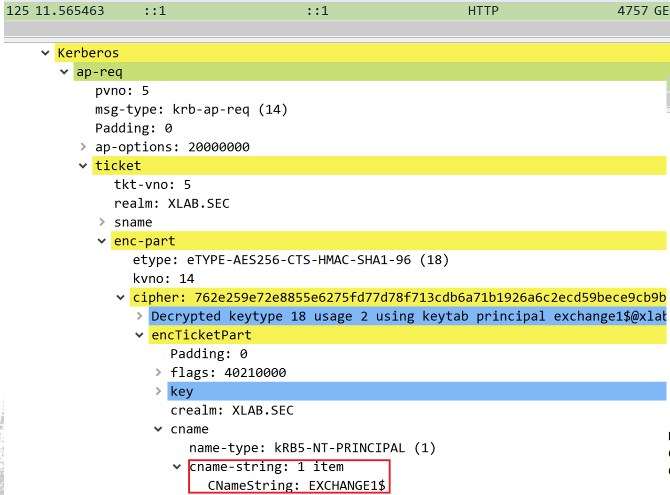
HTTP 80 / HTTPS 443

HTTP 81 / HTTPS 444

Exchange Server - Standalone

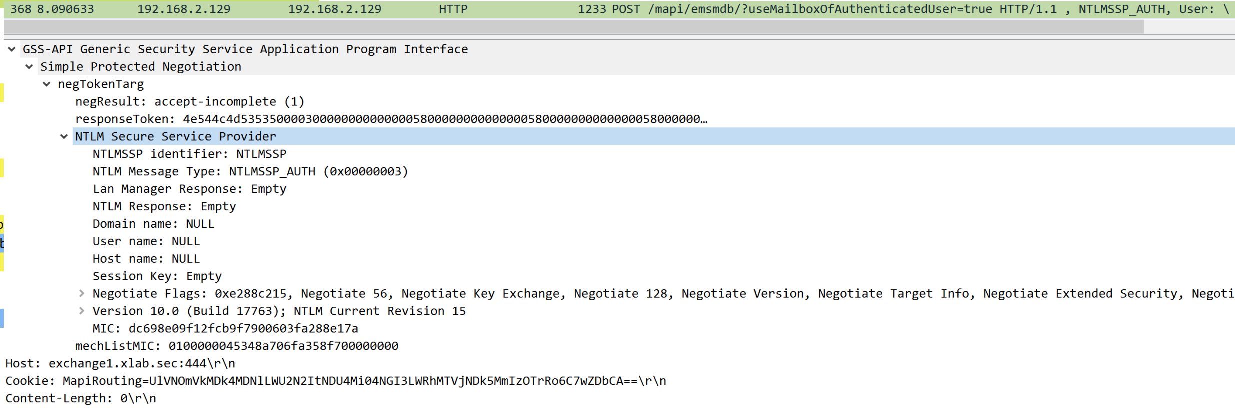
The Front End HttpProxy forward client access requests to The Back End

The Front End authenticates to Back End with machine account and SYSTEM account



```
125 11.565463     ::1                    ::1                   HTTP                4757 GET /owa/ HTTP/1.1

 ⌄ Kerberos
   ⌄ ap-req
       pvno: 5
       msg-type: krb-ap-req (14)
       Padding: 0
     › ap-options: 20000000
     ⌄ ticket
         tkt-vno: 5
         realm: XLAB.SEC
       › sname
       ⌄ enc-part
           etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
           kvno: 14
         ⌄ cipher: 762e259e72e8855e6275fd77d78f713cdb6a71b1926a6c2ecd59bece9cb9b
           › Decrypted keytype 18 usage 2 using keytab principal exchange1$@xlab
           ⌄ encTicketPart
               Padding: 0
             › flags: 40210000
             › key
               crealm: XLAB.SEC
             ⌄ cname
                 name-type: kRB5-NT-PRINCIPAL (1)
               ⌄ cname-string: 1 item
                   CNameString: EXCHANGE1$
```

Kerberos AP-REQ to Back-End with EXCHANGE$

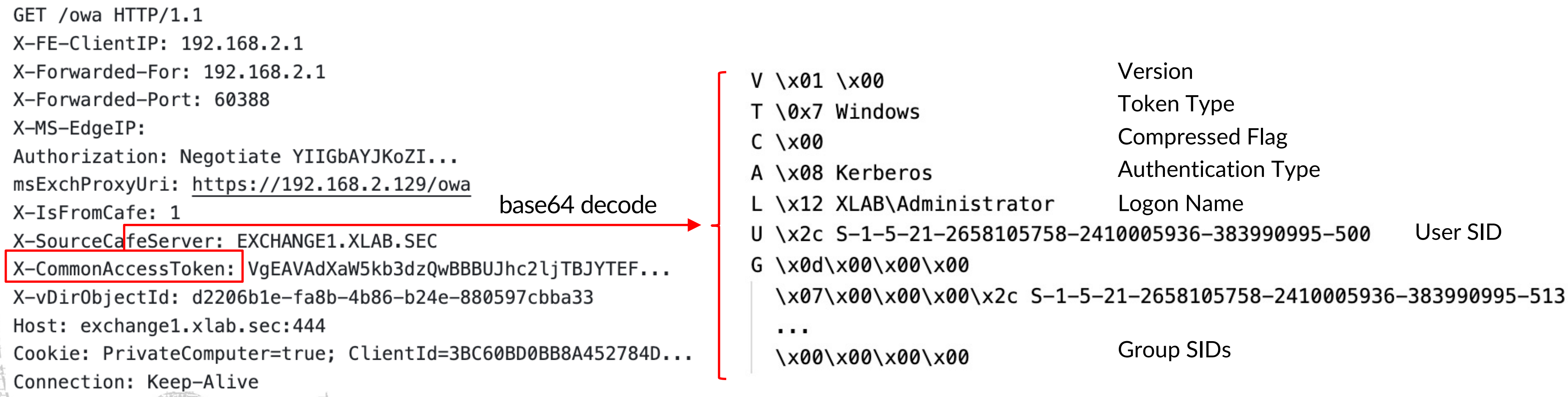


```
368 8.090633      192.168.2.129      192.168.2.129        HTTP            1233 POST /mapi/emsmdb/?useMailboxOfAuthenticatedUser=true HTTP/1.1 , NTLMSSP_AUTH, User: \

⌄ GSS-API Generic Security Service Application Program Interface
  ⌄ Simple Protected Negotiation
    ⌄ negTokenTarg
        negResult: accept-incomplete (1)
        responseToken: 4e544c4d53535000030000000000000058000000000000005800000000000000058000000…
      ⌄ NTLM Secure Service Provider
          NTLMSSP identifier: NTLMSSP
          NTLM Message Type: NTLMSSP_AUTH (0x00000003)
          Lan Manager Response: Empty
          NTLM Response: Empty
          Domain name: NULL
          User name: NULL
          Host name: NULL
          Session Key: Empty
        › Negotiate Flags: 0xe288c215, Negotiate 56, Negotiate Key Exchange, Negotiate 128, Negotiate Version, Negotiate Target Info, Negotiate Extended Security, Negoti
        › Version 10.0 (Build 17763); NTLM Current Revision 15
          MIC: dc698e09f12fcb9f7900603fa288e17a
        mechListMIC: 0100000045348a706fa358f700000000
Host: exchange1.xlab.sec:444\r\n
Cookie: MapiRouting=UlVNOmVkMDk4MDNlLWU2N2ItNDU4Mi04NGI3LWRhMTVjNDk5MmIzOTrRo6C7wZDbCA==\r\n
Content-Length: 0\r\n
```

Local NTLM authentication with SYSTEM account

X-CommonAccessToken

The Front End and the Back End synchronize user identities through X-CommonAccessToken



```
GET /owa HTTP/1.1
X-FE-ClientIP: 192.168.2.1
X-Forwarded-For: 192.168.2.1
X-Forwarded-Port: 60388
X-MS-EdgeIP:
Authorization: Negotiate YIIGbAYJKoZI...
msExchProxyUri: https://192.168.2.129/owa
X-IsFromCafe: 1
X-SourceCafeServer: EXCHANGE1.XLAB.SEC
X-CommonAccessToken: VgEAVAdXaW5kb3dzQwBBBUJhc2ljTBJYTEF...
X-vDirObjectId: d2206b1e-fa8b-4b86-b24e-880597cbba33
Host: exchange1.xlab.sec:444
Cookie: PrivateComputer=true; ClientId=3BC60BD0BB8A452784D...
Connection: Keep-Alive
```

base64 decode

```
V \x01 \x00
T \0x7 Windows
C \x00
A \x08 Kerberos
L \x12 XLAB\Administrator
U \x2c S-1-5-21-2658105758-2410005936-383990995-500
G \x0d\x00\x00\x00
  \x07\x00\x00\x00\x2c S-1-5-21-2658105758-2410005936-383990995-513
  ...
  \x00\x00\x00\x00
```

Version

Token Type

Compressed Flag

Authentication Type

Logon Name

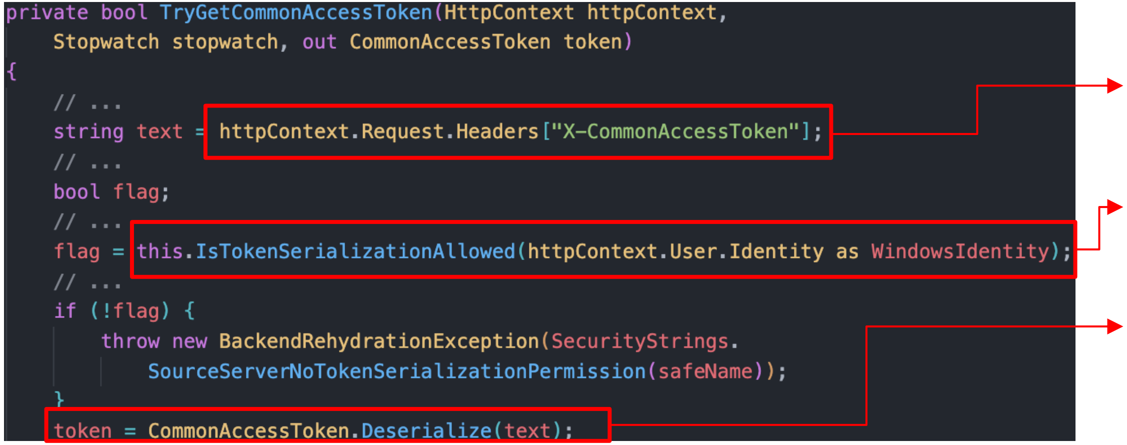
User SID

Group SIDs

The Back End use the X-CommonAccessToken to create a new user token

X-CommonAccessToken

Microsoft.Exchange.Security/Exchange/Security/Authentication/BackendRehydrationModule.cs



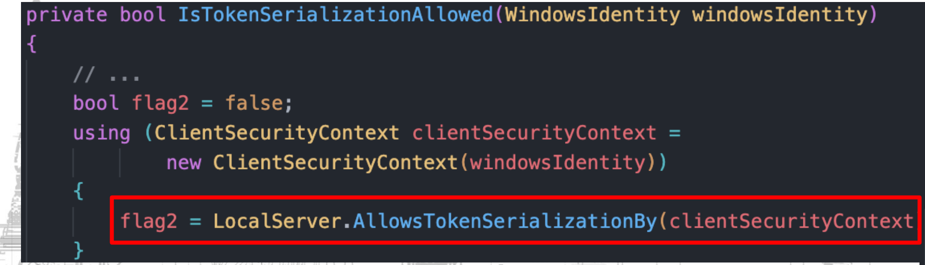
```
private bool TryGetCommonAccessToken(HttpContext httpContext,
    Stopwatch stopwatch, out CommonAccessToken token)
{
    // ...
    string text = httpContext.Request.Headers["X-CommonAccessToken"];
    // ...
    bool flag;
    // ...
    flag = this.IsTokenSerializationAllowed(httpContext.User.Identity as WindowsIdentity);
    // ...
    if (!flag) {
        throw new BackendRehydrationException(SecurityStrings.
            SourceServerNoTokenSerializationPermission(safeName));
    }
    token = CommonAccessToken.Deserialize(text);
```

Back End gets X-CommonAccessToken from HTTP header

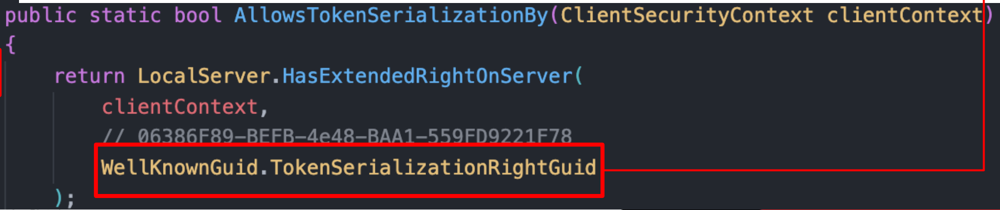
Check if the current user has TokenSerialization right

Deserialize it to create a new user token

Use the new user token to access Back End endpoints



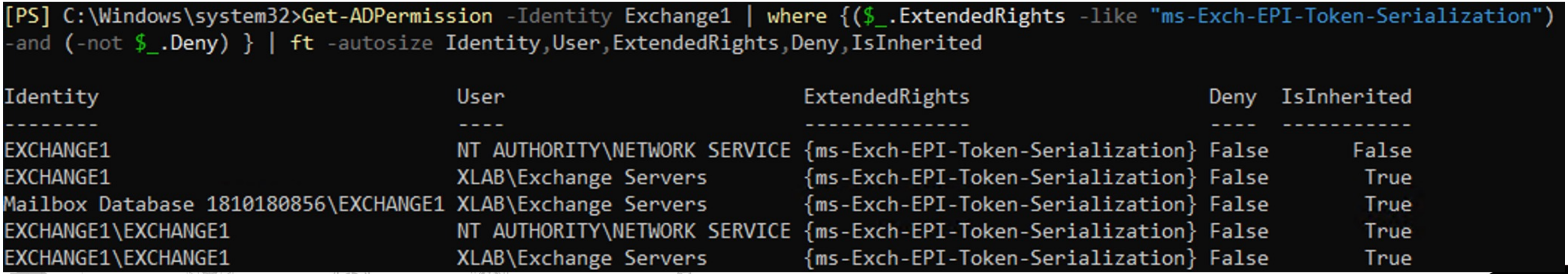
```
private bool IsTokenSerializationAllowed(WindowsIdentity windowsIdentity)
{
    // ...
    bool flag2 = false;
    using (ClientSecurityContext clientSecurityContext =
            new ClientSecurityContext(windowsIdentity))
    {
        flag2 = LocalServer.AllowsTokenSerializationBy(clientSecurityContext
    }
```



```
public static bool AllowsTokenSerializationBy(ClientSecurityContext clientContext)
{
    return LocalServer.HasExtendedRightOnServer(
        clientContext,
        // 06386F89-BEFB-4e48-BAA1-559FD9221F78
        WellKnownGuid.TokenSerializationRightGuid
    );
```

Exchange Server Machine Account

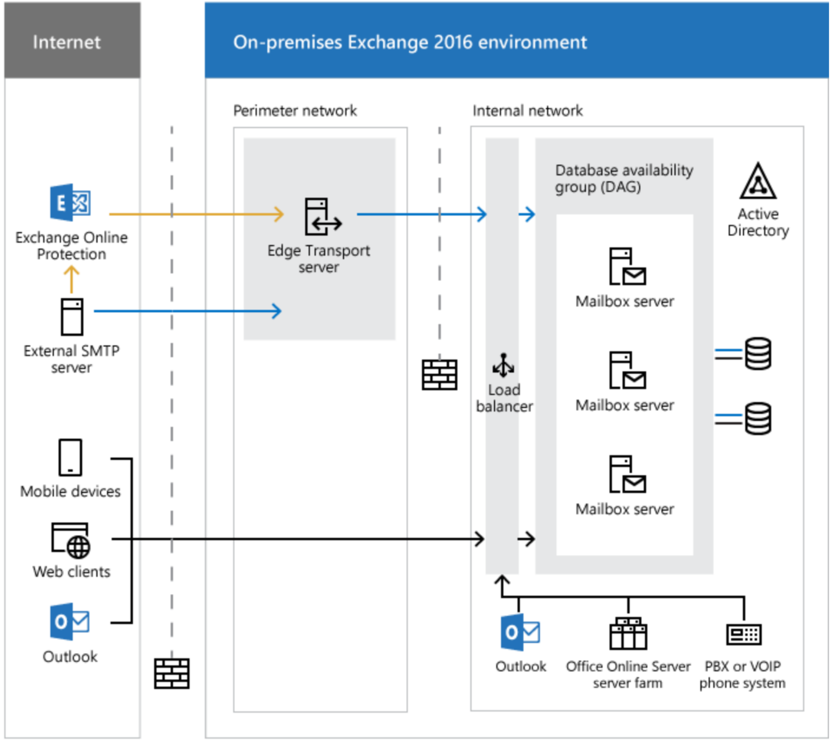
- TokenSerialization ExtendedRight
  - ms-Exch-EPI-Token-Serialization (06386F89-BEFB-4e48-BAA1-559FD9221F78)
- Members of the Exchange Servers group have this right on Exchange Servers
- Exchange machine accounts will be added to Exchange Servers group when installing Exchange Servers



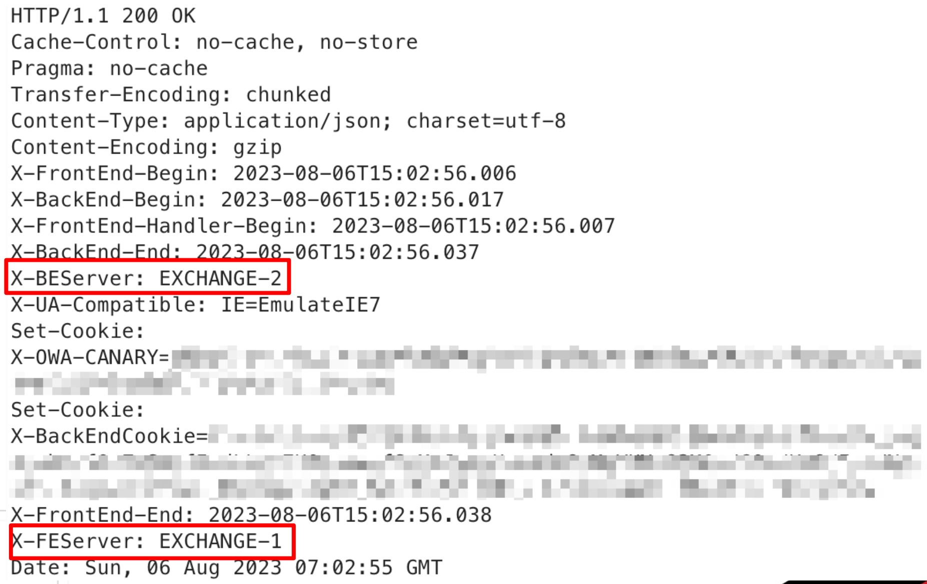
```
[PS] C:\Windows\system32>Get-ADPermission -Identity Exchange1 | where {($_.ExtendedRights -like "ms-Exch-EPI-Token-Serialization")
-and (-not $_.Deny) } | ft -autosize Identity,User,ExtendedRights,Deny,IsInherited

Identity                             User                         ExtendedRights                    Deny  IsInherited
--------                             ----                         --------------                    ----  -----------
EXCHANGE1                            NT AUTHORITY\NETWORK SERVICE {ms-Exch-EPI-Token-Serialization} False       False
EXCHANGE1                            XLAB\Exchange Servers        {ms-Exch-EPI-Token-Serialization} False        True
Mailbox Database 1810180856\EXCHANGE1 XLAB\Exchange Servers       {ms-Exch-EPI-Token-Serialization} False        True
EXCHANGE1\EXCHANGE1                  NT AUTHORITY\NETWORK SERVICE {ms-Exch-EPI-Token-Serialization} False        True
EXCHANGE1\EXCHANGE1                  XLAB\Exchange Servers        {ms-Exch-EPI-Token-Serialization} False        True
```

Exchange Server - Cluster

Cluster architecture is widely used in enterprise environments for high availability.



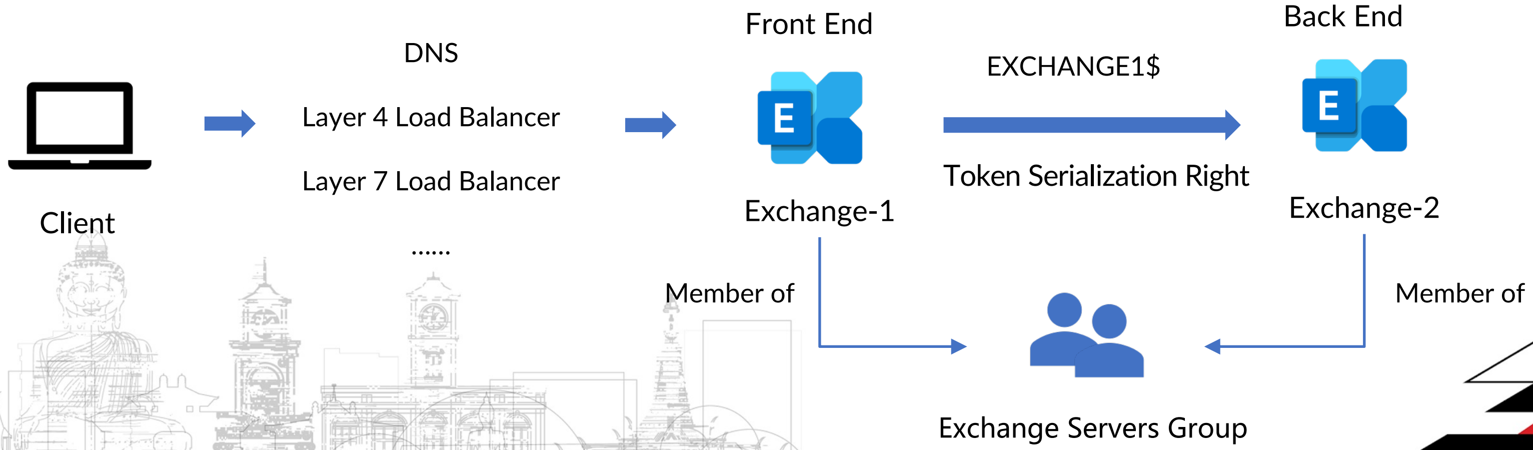
The Front End and the Back End can be on different Exchange Servers



```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store
Pragma: no-cache
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Content-Encoding: gzip
X-FrontEnd-Begin: 2023-08-06T15:02:56.006
X-BackEnd-Begin: 2023-08-06T15:02:56.017
X-FrontEnd-Handler-Begin: 2023-08-06T15:02:56.007
X-BackEnd-End: 2023-08-06T15:02:56.037
X-BEServer: EXCHANGE-2
X-UA-Compatible: IE=EmulateIE7
Set-Cookie:
X-OWA-CANARY=[redacted]
Set-Cookie:
X-BackEndCookie=[redacted]
X-FrontEnd-End: 2023-08-06T15:02:56.038
X-FEServer: EXCHANGE-1
Date: Sun, 06 Aug 2023 07:02:55 GMT
```

Exchange Server - Cluster

- Exchange machine accounts will be used when Frontends authenticate to other Backends in different Exchange Servers
- All Exchange machine accounts are in the same group and have same privileges on all Exchange servers in the AD
- Exchange1 can access the backend of Exchange2 just like the backend of itself



Client

DNS

Layer 4 Load Balancer

Layer 7 Load Balancer

......

Front End

Exchange-1

EXCHANGE1$

Token Serialization Right

Back End

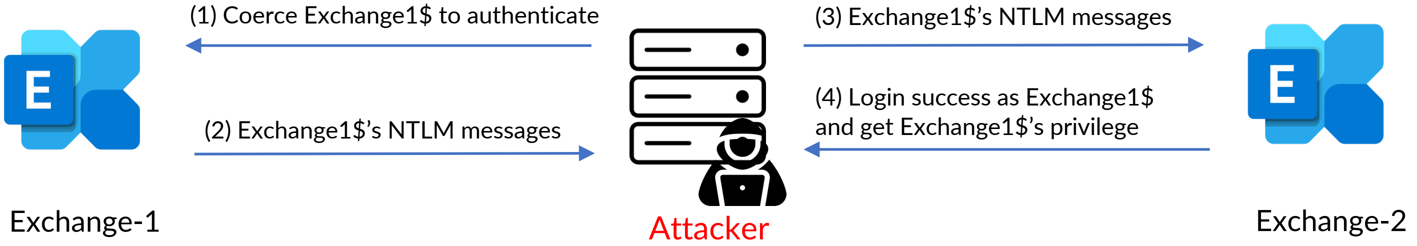
Exchange-2

Member of

Member of

Exchange Servers Group

NTLM Relay to Exchange Server



(1) Coerce Exchange1$ to authenticate
(2) Exchange1$'s NTLM messages
(3) Exchange1$'s NTLM messages
(4) Login success as Exchange1$ and get Exchange1$'s privilege
Exchange-1
Attacker
Exchange-2

Exchange Endpoints

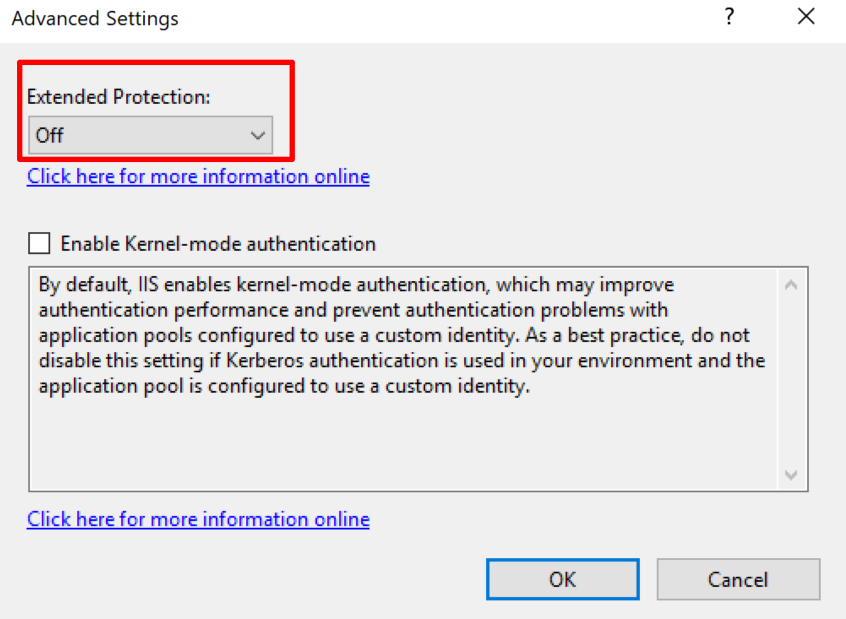
Endpoints	Description	Mail Access	Management
OWA	Outlook Web App	✓	-
EWS	Exchange Web Services, used by Outlook for macOS and Outlook add-ins	✓	-
API	REST API, available in Exchange 2016 CU3 or newer	✓	-
Microsoft-Server-ActiveSync	ActiveSync let you synchronize a mobile device with your Exchange mailbox	✓	-
MAPI	MAPI over HTTP, used by modern Microsoft Outlook	✓	-
RPC	Outlook Anywhere, used by Microsoft Outlook 2013, Outlook 2010, or Outlook 2007	✓	-
Powershell	Used by Exchange PowerShell Cmdlets	✓	✓
ECP	Exchange Control Panel	-	✓
Autodiscover	Used by client application to configure itself	-	-
OAB	Offline Address Book	-	-

Front End endpoints NTLM support

Frontend Endpoints	Authentication Methods
/EWS/	Kerberos, NTLM
/mapi/emsmdb	Kerberos, NTLM
/API/	Kerberos, NTLM
/owa/Integrated/	Web Form, Kerberos, NTLM
/Microsoft-Server-ActiveSync/Proxy	Basic, Kerberos, NTLM
/rpc/rpcproxy.dll	Basic, Kerberos, NTLM
/autodiscover/	Kerberos, NTLM, Basic
/oab/	Kerberos, NTLM
/ecp/	Web Form
/Powershell/	Kerberos

NTLM Relay to the Front End

- The EPA is disabled on all Front-End endpoints by default



Advanced Settings

Extended Protection:

Off

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 application pools configured to use a custom identity. As a best practice, do not disable this setting if Kerberos authentication is used in your environment and the application pool is configured to use a custom identity.

Click here for more information online

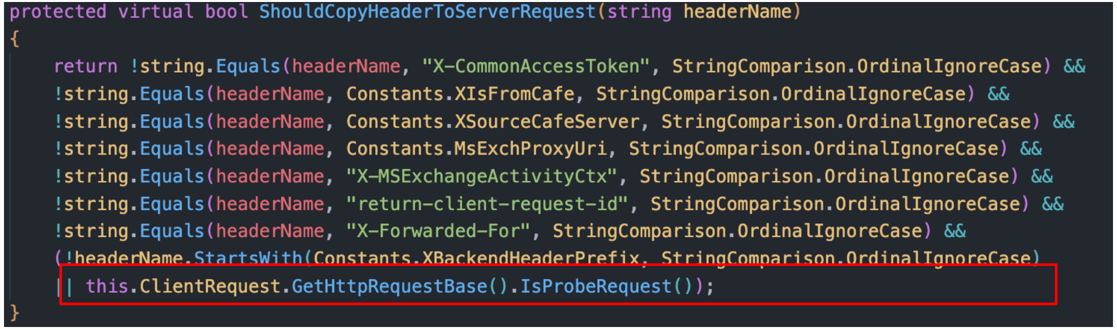
EPA is not compatible well with 7-layer load balancer by default

Enterprise IT administrators usually do not enable EPA on Exchange Servers

NTLM Relay to the Font End

Can we use X-CommonAccessToken to impersonate arbitrary user when we relay to the Front End?

Microsoft.Exchange.FrontEndHttpProxy.dll\HttpProxy\ProxyRequestHandler.cs

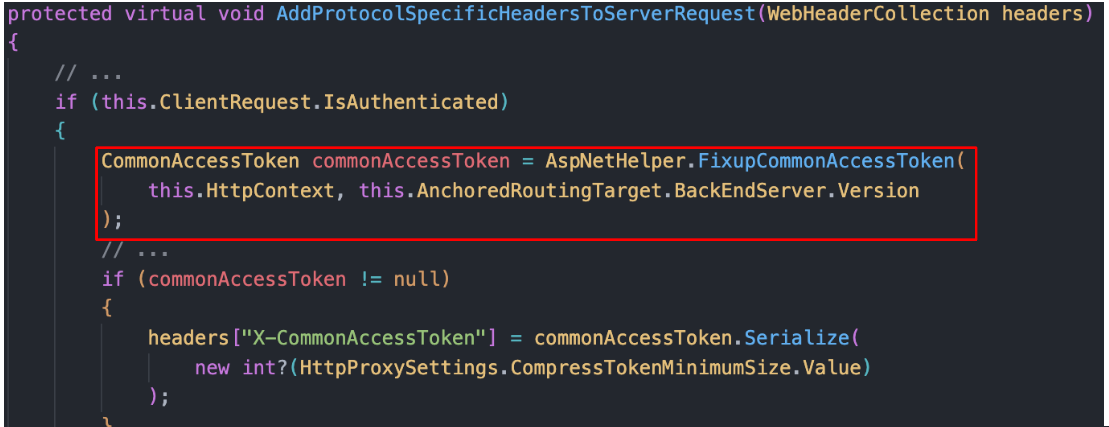


```
protected virtual bool ShouldCopyHeaderToServerRequest(string headerName)
{
    return !string.Equals(headerName, "X-CommonAccessToken", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.XIsFromCafe, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.XSourceCafeServer, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.MsExchProxyUri, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "X-MSExchangeActivityCtx", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "return-client-request-id", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "X-Forwarded-For", StringComparison.OrdinalIgnoreCase) &&
    (!headerName.StartsWith(Constants.XBackendHeaderPrefix, StringComparison.OrdinalIgnoreCase)
    || this.ClientRequest.GetHttpRequestBase().IsProbeRequest());
}
```

```
PrepareServerRequest
|__ CopyHeadersToServerRequest
    |__ ShouldCopyHeaderToServerRequest
```

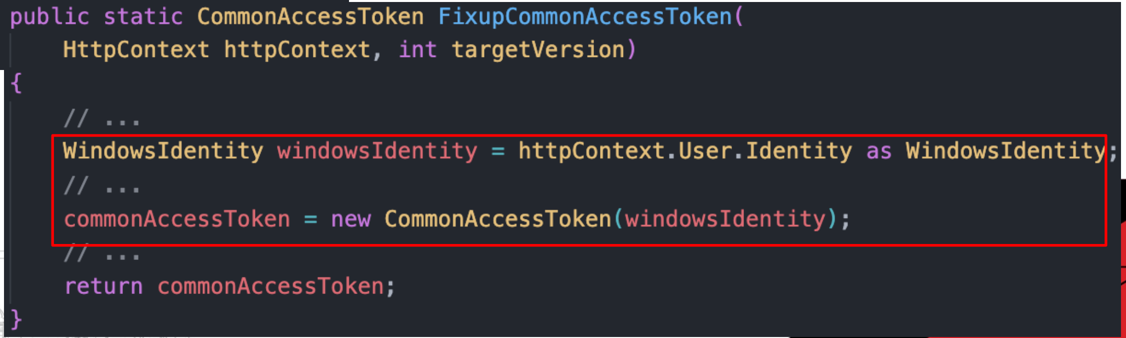
X-CommonAccessToken in the client request doesn't be allowed to forward to the server request

NTLM Relay to the Font End



```
protected virtual void AddProtocolSpecificHeadersToServerRequest(WebHeaderCollection headers)
{
    // ...
    if (this.ClientRequest.IsAuthenticated)
    {
        CommonAccessToken commonAccessToken = AspNetHelper.FixupCommonAccessToken(
            this.HttpContext, this.AnchoredRoutingTarget.BackEndServer.Version
        );
        // ...
        if (commonAccessToken != null)
        {
            headers["X-CommonAccessToken"] = commonAccessToken.Serialize(
                new int?(HttpProxySettings.CompressTokenMinimumSize.Value)
            );
        }
    }
}
```

The FrontEnd proxy will create a new X-CommonAccessToken with current user‘s identity



```
public static CommonAccessToken FixupCommonAccessToken(
    HttpContext httpContext, int targetVersion)
{
    // ...
    WindowsIdentity windowsIdentity = httpContext.User.Identity as WindowsIdentity;
    // ...
    commonAccessToken = new CommonAccessToken(windowsIdentity);
    // ...
    return commonAccessToken;
}
```

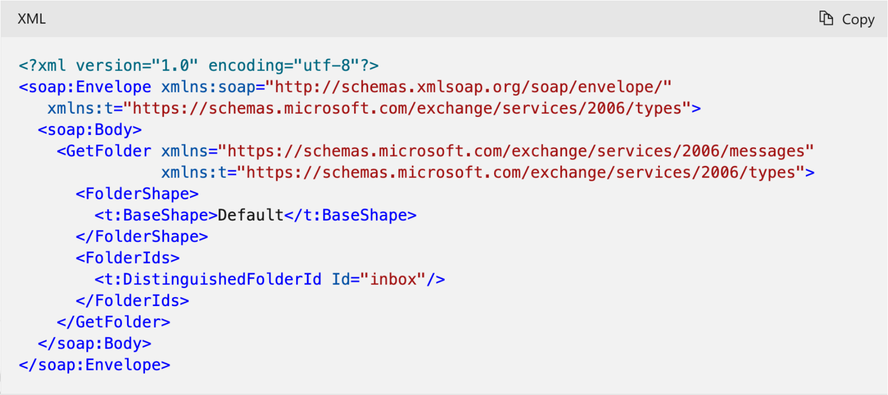
Can we impersonate other users without X-CommonAccessToken?

Exchange EWS

Exchange Web Services API

- used by Outlook for macOS and Outlook add-ins by default
- provide full-ability SOAP API for accessing and manipulating emails, attachments, contacts, calendar events, etc.

Endpoint: /EWS/Exchange.asmx , with NTLM support

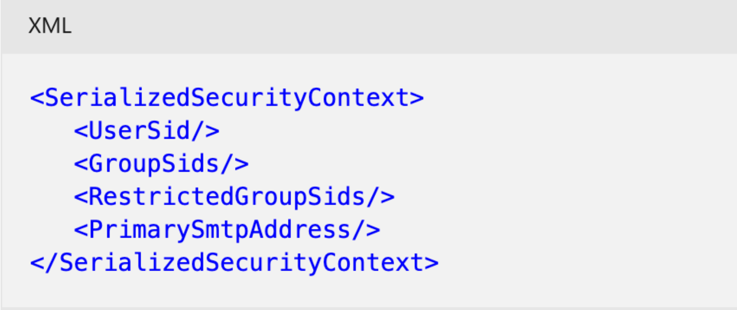


XML

Copy

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:t="https://schemas.microsoft.com/exchange/services/2006/types">
  <soap:Body>
    <GetFolder xmlns="https://schemas.microsoft.com/exchange/services/2006/messages"
               xmlns:t="https://schemas.microsoft.com/exchange/services/2006/types">
      <FolderShape>
        <t:BaseShape>Default</t:BaseShape>
      </FolderShape>
      <FolderIds>
        <t:DistinguishedFolderId Id="inbox"/>
      </FolderIds>
    </GetFolder>
  </soap:Body>
</soap:Envelope>
```

EWS API support token serialization natively



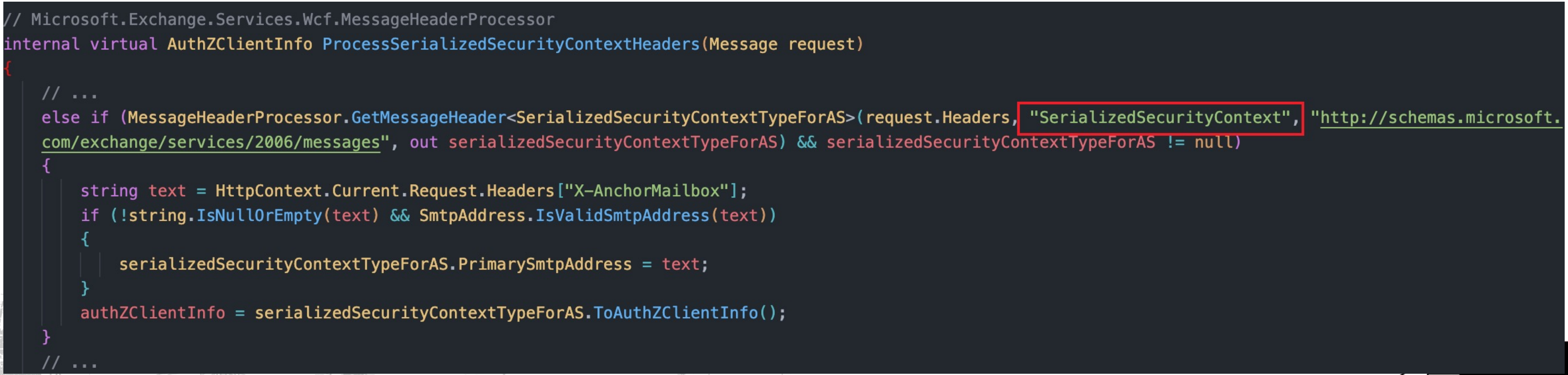
XML

```
<SerializedSecurityContext>
   <UserSid/>
   <GroupSids/>
   <RestrictedGroupSids/>
   <PrimarySmtpAddress/>
</SerializedSecurityContext>
```

Exchange EWS

SerializedSecurityContext

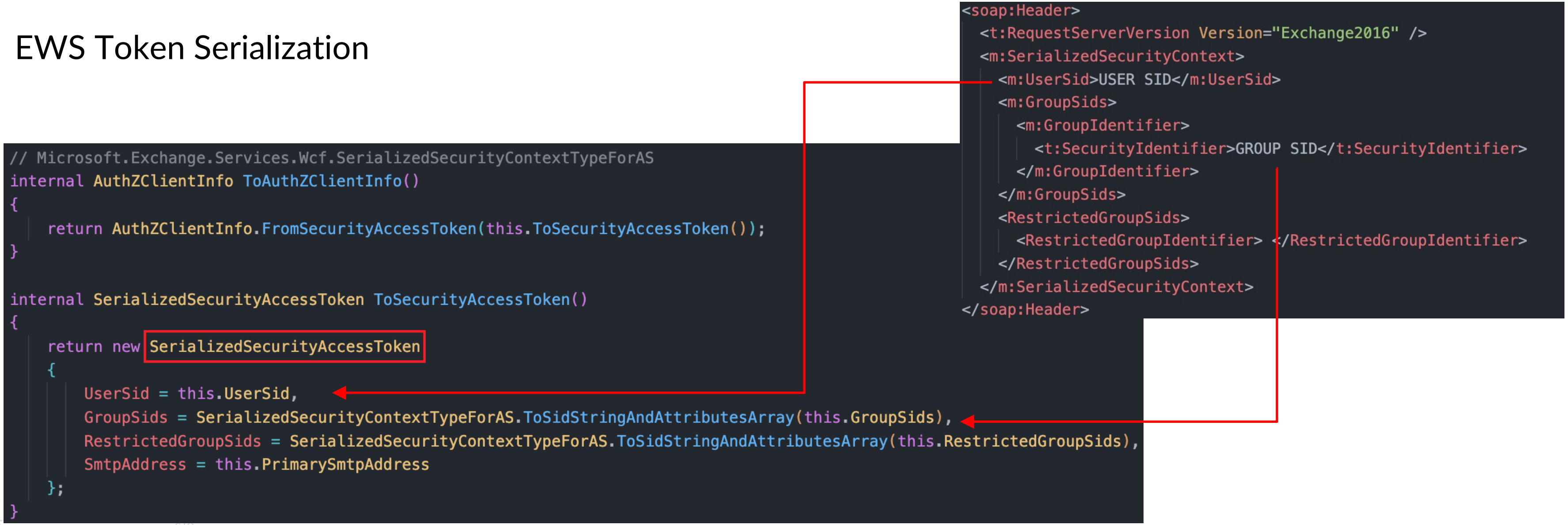
The SerializedSecurityContext element is used in the SOAP header for token serialization in server-to-server authentication.



```
// Microsoft.Exchange.Services.Wcf.MessageHeaderProcessor
internal virtual AuthZClientInfo ProcessSerializedSecurityContextHeaders(Message request)
{
    // ...
    else if (MessageHeaderProcessor.GetMessageHeader<SerializedSecurityContextTypeForAS>(request.Headers, "SerializedSecurityContext", "http://schemas.microsoft.
    com/exchange/services/2006/messages", out serializedSecurityContextTypeForAS) && serializedSecurityContextTypeForAS != null)
    {
        string text = HttpContext.Current.Request.Headers["X-AnchorMailbox"];
        if (!string.IsNullOrEmpty(text) && SmtpAddress.IsValidSmtpAddress(text))
        {
            serializedSecurityContextTypeForAS.PrimarySmtpAddress = text;
        }
        authZClientInfo = serializedSecurityContextTypeForAS.ToAuthZClientInfo();
    }
    // ...
```

Exchange EWS

EWS Token Serialization



```xml
<soap:Header>
  <t:RequestServerVersion Version="Exchange2016" />
  <m:SerializedSecurityContext>
    <m:UserSid>USER SID</m:UserSid>
    <m:GroupSids>
      <m:GroupIdentifier>
        <t:SecurityIdentifier>GROUP SID</t:SecurityIdentifier>
      </m:GroupIdentifier>
    </m:GroupSids>
    <RestrictedGroupSids>
      <RestrictedGroupIdentifier> </RestrictedGroupIdentifier>
    </RestrictedGroupSids>
  </m:SerializedSecurityContext>
</soap:Header>
```

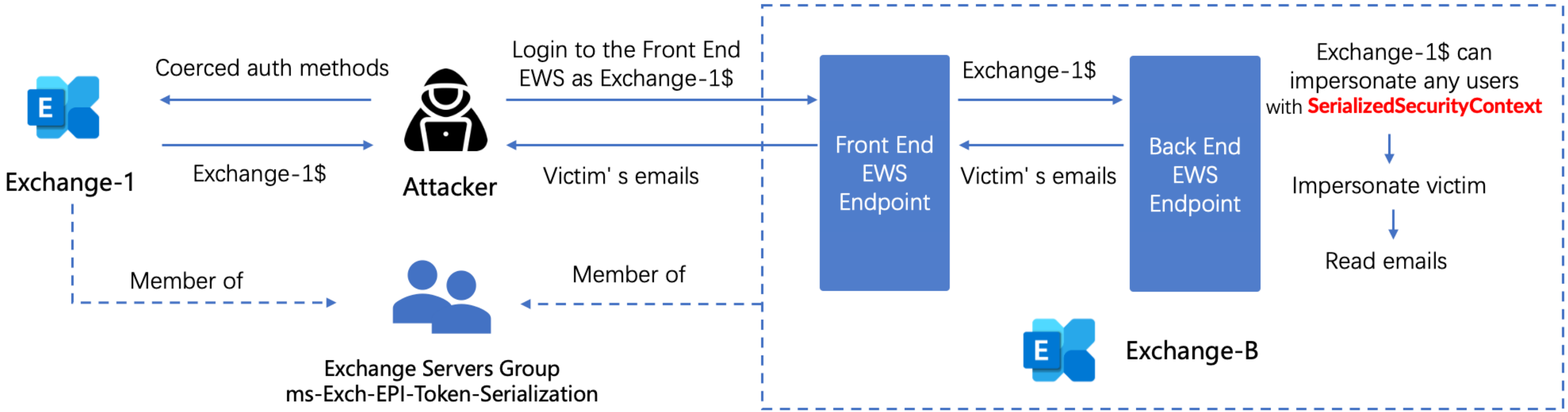
```csharp
// Microsoft.Exchange.Services.Wcf.SerializedSecurityContextTypeForAS
internal AuthZClientInfo ToAuthZClientInfo()
{
    return AuthZClientInfo.FromSecurityAccessToken(this.ToSecurityAccessToken());
}

internal SerializedSecurityAccessToken ToSecurityAccessToken()
{
    return new SerializedSecurityAccessToken
    {
        UserSid = this.UserSid,
        GroupSids = SerializedSecurityContextTypeForAS.ToSidStringAndAttributesArray(this.GroupSids),
        RestrictedGroupSids = SerializedSecurityContextTypeForAS.ToSidStringAndAttributesArray(this.RestrictedGroupSids),
        SmtpAddress = this.PrimarySmtpAddress
    };
}
```

EWS will create user token with the UserSid and GroupSids in the SerializedSecurityContext

Users with token serialization right can impersonate any Exchange users on the EWS endpoint

NTLM Relay to the Font End - CVE-2021-33768

The attacker can perform NTLM relay to the Front-End EWS, impersonate arbitrary Exchange users to send emails, read emails, download attachments, do anything EWS supports.

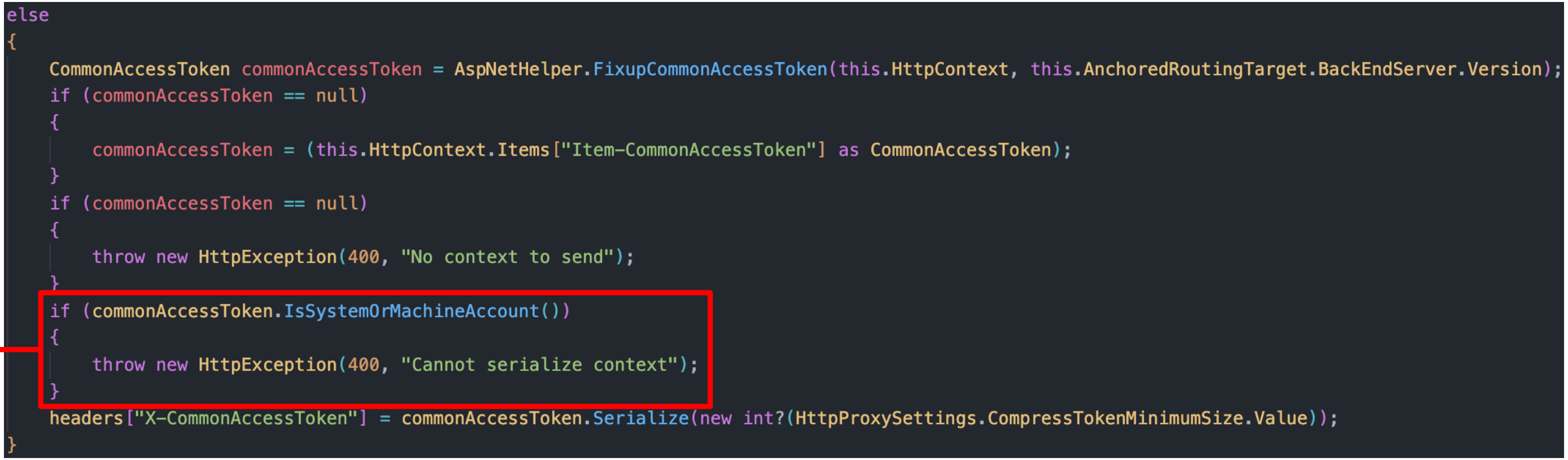


NTLM Relay to the Font End - CVE-2021-33768

DEMO: https://youtu.be/I_HOLSztI4Q

CVE-2021-33768 – Patch Analysis

Microsoft.Exchange.FrontEndHttpProxy.dll\HttpProxy\ProxyRequestHandler.cs!AddProtocolSpecificHeadersToServerRequest

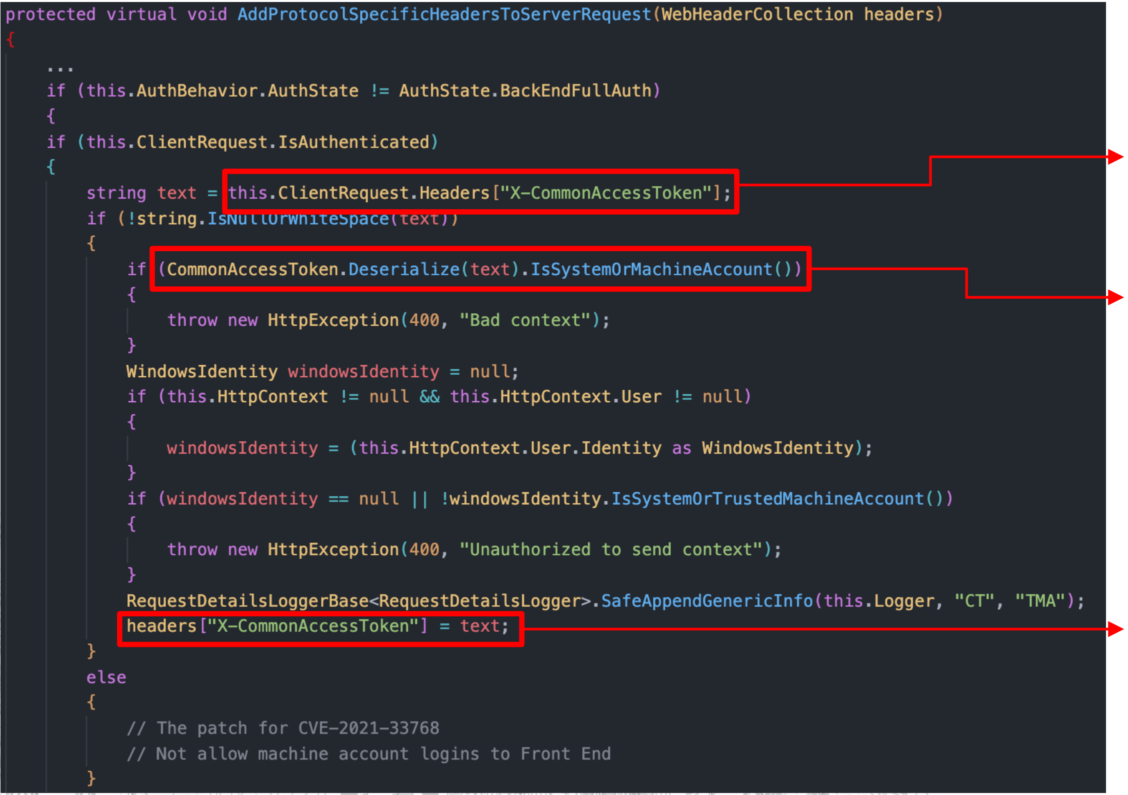


```
else
{
    CommonAccessToken commonAccessToken = AspNetHelper.FixupCommonAccessToken(this.HttpContext, this.AnchoredRoutingTarget.BackEndServer.Version);
    if (commonAccessToken == null)
    {
        commonAccessToken = (this.HttpContext.Items["Item-CommonAccessToken"] as CommonAccessToken);
    }
    if (commonAccessToken == null)
    {
        throw new HttpException(400, "No context to send");
    }
    if (commonAccessToken.IsSystemOrMachineAccount())
    {
        throw new HttpException(400, "Cannot serialize context");
    }
    headers["X-CommonAccessToken"] = commonAccessToken.Serialize(new int?(HttpProxySettings.CompressTokenMinimumSize.Value));
}
```

Not allow machine account logins to the Front End anymore

CVE-2021-33768 – Patch Bypass

This branch is introduced in the same Security Update with the patch



```
protected virtual void AddProtocolSpecificHeadersToServerRequest(WebHeaderCollection headers)
{
    ...
    if (this.AuthBehavior.AuthState != AuthState.BackEndFullAuth)
    {
    if (this.ClientRequest.IsAuthenticated)
    {
        string text = this.ClientRequest.Headers["X-CommonAccessToken"];
        if (!string.IsNullOrWhiteSpace(text))
        {
            if (CommonAccessToken.Deserialize(text).IsSystemOrMachineAccount())
            {
                throw new HttpException(400, "Bad context");
            }
            WindowsIdentity windowsIdentity = null;
            if (this.HttpContext != null && this.HttpContext.User != null)
            {
                windowsIdentity = (this.HttpContext.User.Identity as WindowsIdentity);
            }
            if (windowsIdentity == null || !windowsIdentity.IsSystemOrTrustedMachineAccount())
            {
                throw new HttpException(400, "Unauthorized to send context");
            }
            RequestDetailsLoggerBase<RequestDetailsLogger>.SafeAppendGenericInfo(this.Logger, "CT", "TMA");
            headers["X-CommonAccessToken"] = text;
        }
        else
        {
            // The patch for CVE-2021-33768
            // Not allow machine account logins to Front End
        }
```

Read X-CommonAccessToken from clientRequest directly

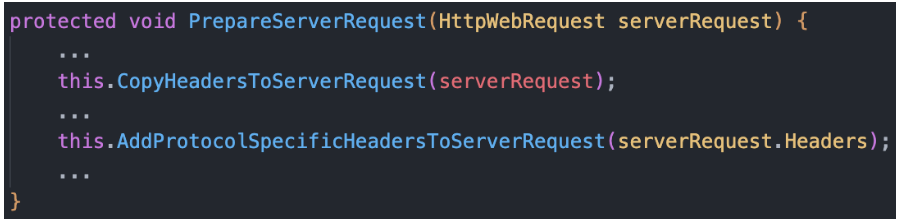
Deserialize X-CommonAccessToken,
check if the identity is a machine account

If the user in the X-CommonAccessToken is not
a machine account

X-CommonAccessToken from client request will be
added to serverRequest.Headers

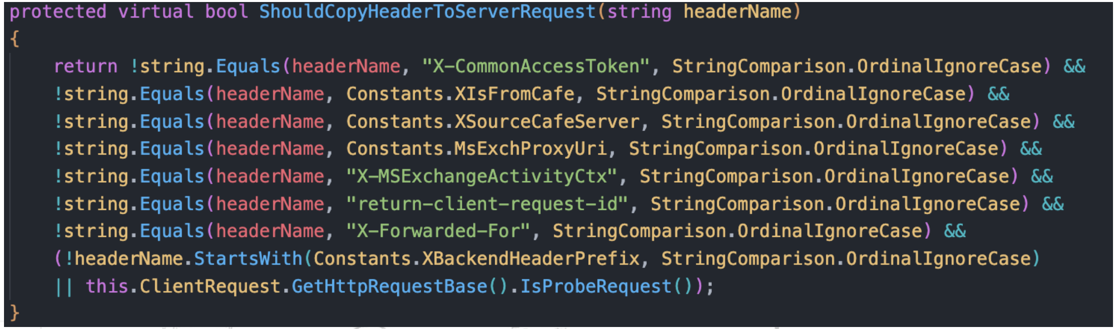
CVE-2021-33768 – Patch Bypass

Microsoft.Exchange.FrontEndHttpProxy.dll\HttpProxy\ProxyRequestHandler.cs



```
protected void PrepareServerRequest(HttpWebRequest serverRequest) {
    ...
    this.CopyHeadersToServerRequest(serverRequest);
    ...
    this.AddProtocolSpecificHeadersToServerRequest(serverRequest.Headers);
    ...
}
```

PrepareServerRequest
|__ CopyHeadersToServerRequest
    |__ ShouldCopyHeaderToServerRequest
|__ AddProtocolSpecificHeadersToServerRequest



```
protected virtual bool ShouldCopyHeaderToServerRequest(string headerName)
{
    return !string.Equals(headerName, "X-CommonAccessToken", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.XIsFromCafe, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.XSourceCafeServer, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, Constants.MsExchProxyUri, StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "X-MSExchangeActivityCtx", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "return-client-request-id", StringComparison.OrdinalIgnoreCase) &&
    !string.Equals(headerName, "X-Forwarded-For", StringComparison.OrdinalIgnoreCase) &&
    (!headerName.StartsWith(Constants.XBackendHeaderPrefix, StringComparison.OrdinalIgnoreCase)
    || this.ClientRequest.GetHttpRequestBase().IsProbeRequest());
}
```

ShouldCopyHeaderToServerRequest doesn't allow X-CommonAccessToken in the client request to be forwarded to the Back End

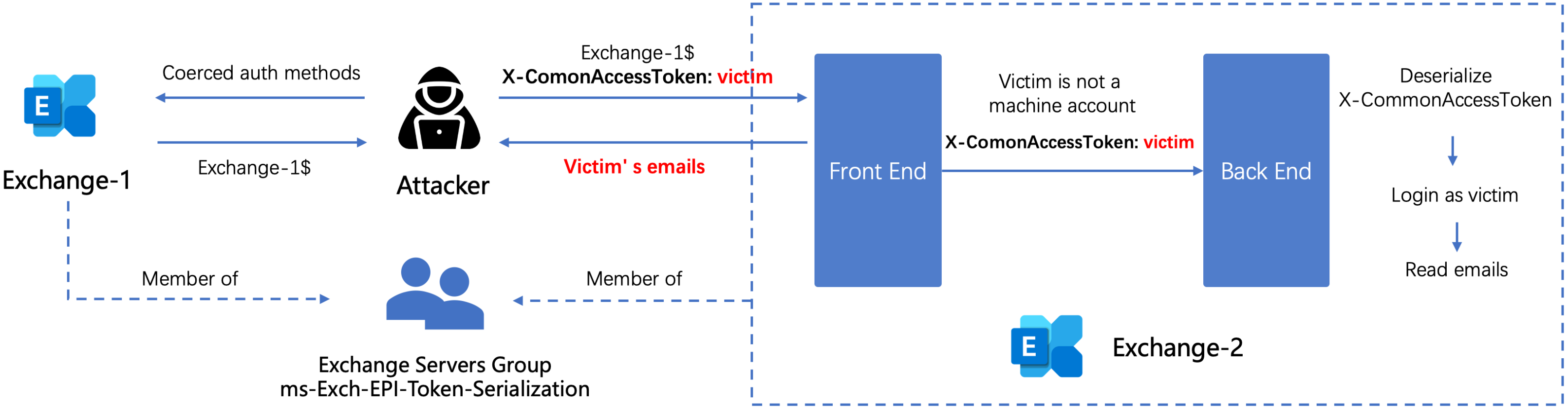
AddProtocolSpecificHeadersToServerRequest is called after ShouldCopyHeaderToServerRequest

Attackers can forge any mailbox user's X-CommonAccessToken, the Front End will forward it directly to the Back End

CVE-2022-21980

The Security Update for CVE-2021-33768 bring us a new and more powerful vulnerability

Attackers can perform NTLM relay to all Front End endpoints have NTLM support and impersonate arbitrary Exchange user



NTLM Relay to the Front End - CVE-2022-21980

All endpoints that support NTLM authentication and support email access are exploitable

- add X-CommonAccessToken to impersonate Exchange users
- implement the corresponding protocol to operate the mail

EWS
- /EWS/Exchange.asmx
- SOAP XML over HTTPS
  - FindFolder / FindItem / GetItem

OWA
- JSON with HTTPS
- /owa/Integrated/service.svc?action=FindConversation
- /owa/Integrated/service.svc?action=GetConversationItems

API
- JSON with HTTPS
- /api/v2.0/users/victim@xlab.sec/mailFolders/inbox/messages

ActiveSync
- /Microsoft-Server-ActiveSync/Proxy
- WBXML over HTTPS
  - FolderSync to dump collectionIds
  - Sync to dump email contents

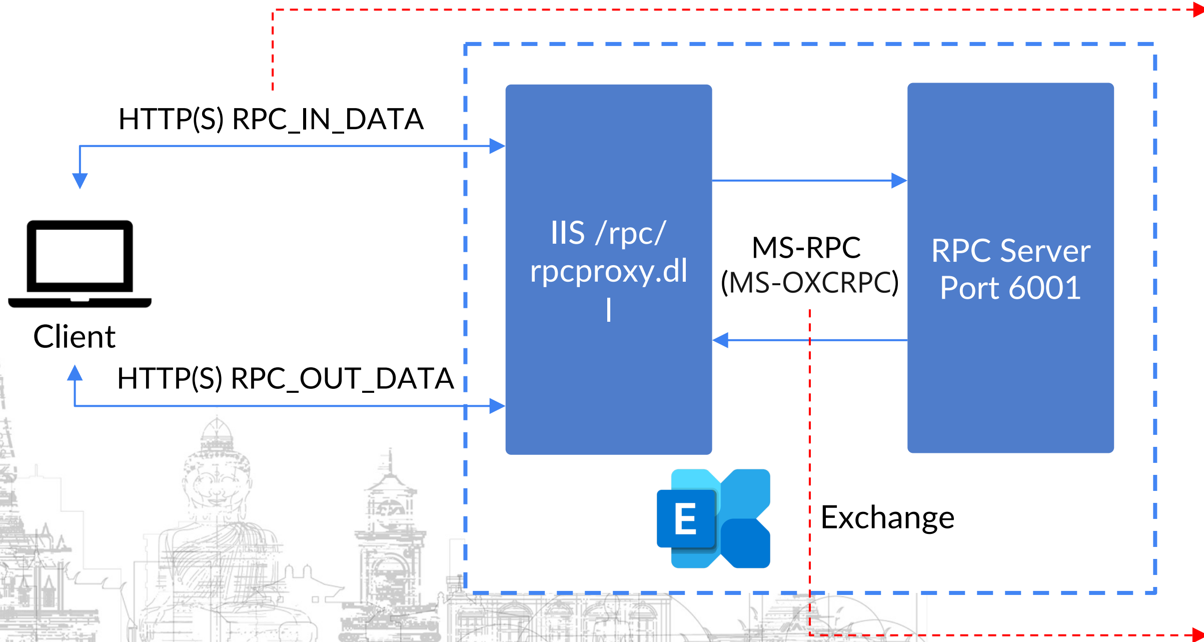
MAPI
- /mapi/emsmdb/?MailboxId=victimmailboxid
- MS-OXPROPS over HTTPS
  - RopGetPropertiesListRequest
  - RopGetPropertiesSpecificRequest

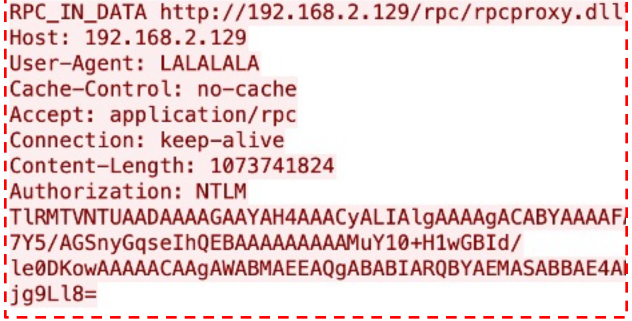
RPC
- /rpc/rpcproxy.dll
- MS-OXPROPS over MS-RPC over HTTP(S)

RPC (Outlook Anywhere)

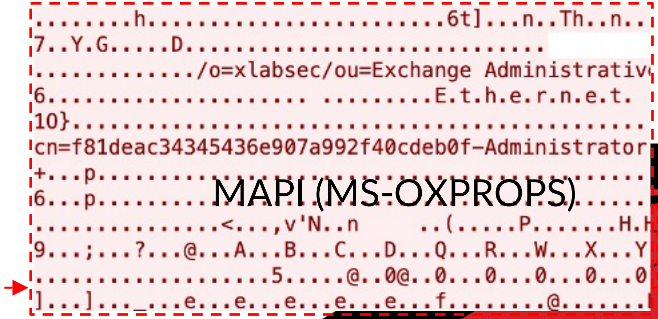
Endpoint: /rpc/rpcproxy.dll, works as an RPC proxy

- Requires two connections RPC_IN_DATA and RPC_OUT_DATA
- MAPI (MS-OXPROPS) over MS-RPC (MS-OXCRPC) over HTTP(S)





```
RPC_IN_DATA http://192.168.2.129/rpc/rpcproxy.dll
Host: 192.168.2.129
User-Agent: LALALALA
Cache-Control: no-cache
Accept: application/rpc
Connection: keep-alive
Content-Length: 1073741824
Authorization: NTLM
TlRMTVNTUAADAAAAGAAYAH4AAACyALIAlgAAAAgACABYAAAAF
7Y5/AGSnyGqseIhQEBAAAAAAAAAMuY10+H1wGBId/
le0DKowAAAAACAAgAWABMAEEAQgABABIARQBYAEMASABBAE4A
jg9Ll8=
```



```
........h.........................6t]...n..Th..n..
7..Y.G.....D..............................
............../o=xlabsec/ou=Exchange Administrativ
6..................... ..........E.t.h.e.r.n.e.t.
10}...........................................
cn=f81deac34345436e907a992f40cdeb0f-Administrator
+...p.........................................
6...p.........................................
.................<...,v'N..n      ..(.....P.......H.H
9...;...?...@...A...B...C...D...Q...R...W...X...Y
....................5.....@..0@..0...0...0...0...0
]...]..._...e...e...e...e...e...f.........@.......
```

MAPI (MS-OXPROPS)

RPC (Outlook Anywhere)

RPC authentication

- The RPC proxy allows client to skip authentication on the RPC level to get a faster connection
- RPC auth type RPC_C_AUTHN_NONE
- RPC auth level RPC_C_AUTHN_LEVEL_NONE
- No NTLM relay protection on the RPC level

NTLM relay to the Front-End /rpc/rpcproxy.dll

- Trigger NTLM relay twice, login to RPC_IN_DATA and RPC_OUT_DATA as Exchange machine account
- Add X-CommonAccessToken HTTP Header to impersonate arbitrary Exchange user
- Skip RPC authentication to prevent signing / sealing on the RPC level
- Use MAPI(MS-OXPROPS) protocol to access arbitrary emails, attachments, etc.

NTLM Relay to the Front End - CVE-2022-21980

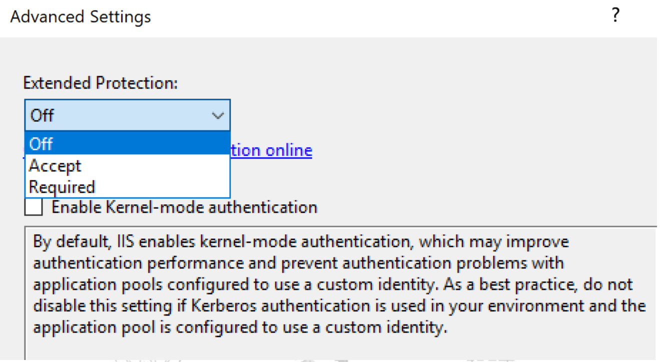
Frontend Endpoints	Authentication Methods	Affected	Exploitable	Impact
/EWS/	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/mapi/emsmdb	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/API/	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/owa/Integrated/	Web Form, Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/Microsoft-Server-ActiveSync/Proxy	Basic, Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/rpc/rpcproxy.dll	Basic, Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/autodiscover/	Kerberos, NTLM, Basic	✓	-	-
/oab/	Kerberos, NTLM	✓	-	-
/ecp/	Web Form	-	-	-
/Powershell/	Kerberos	-	-	-

NTLM Relay to the Front End - CVE-2022-21980

DEMOS: https://www.youtube.com/playlist?list=PLtZO9vwOND910VlyxVOEPoTJNZMWBtv9y

NTLM Relay to the Back End

- Everyone can access the Back End port 444 (no IP/Host whitelist by default)
- All Back End endpoints support NTLM authentication, and the EPA is disabled by default
- The EPA on the Backend needs to be disabled by design, if enabled it will break the communication between the Front-End and the Back-End

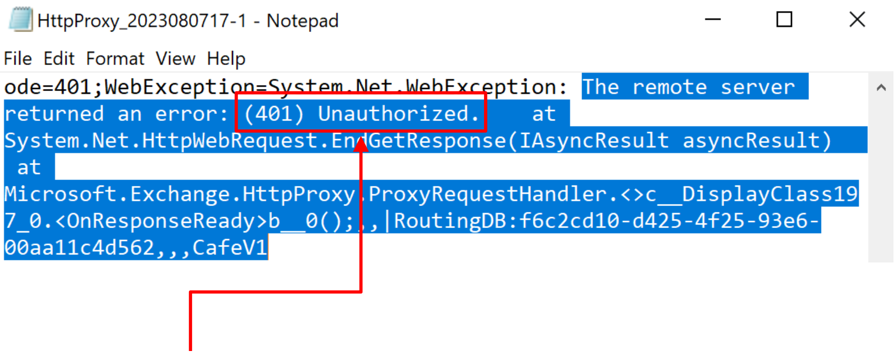


Advanced Settings ?

Extended Protection: Off

□ Enable Kernel-mode authentication

By default, IIS enables kernel-mode authentication, which may improve authentication performance and prevent authentication problems with application pools configured to use a custom identity. As a best practice, do not disable this setting if Kerberos authentication is used in your environment and the application pool is configured to use a custom identity.

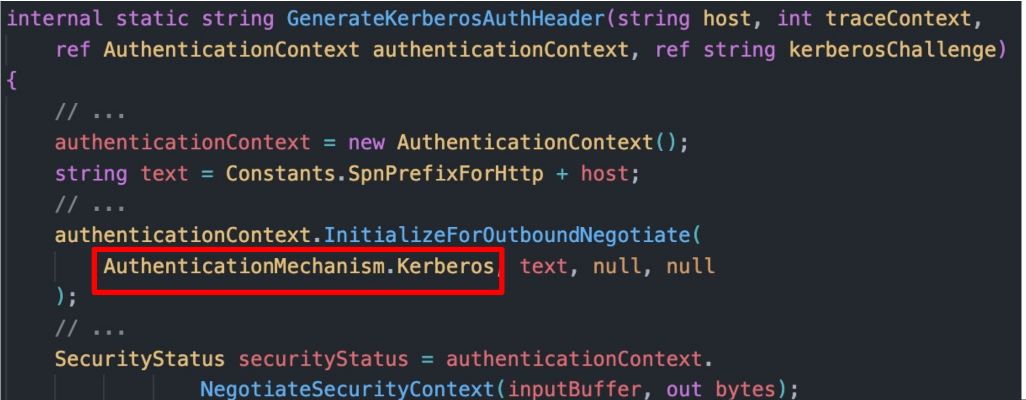


HttpProxy_2023080717-1 - Notepad

```
ode=401;WebException=System.Net.WebException: The remote server
returned an error: (401) Unauthorized.    at
System.Net.HttpWebRequest.EndGetResponse(IAsyncResult asyncResult)
 at
Microsoft.Exchange.HttpProxy.ProxyRequestHandler.<>c__DisplayClass19
7_0.<OnResponseReady>b__0();,,|RoutingDB:f6c2cd10-d425-4f25-93e6-
00aa11c4d562,,,CafeV1
```

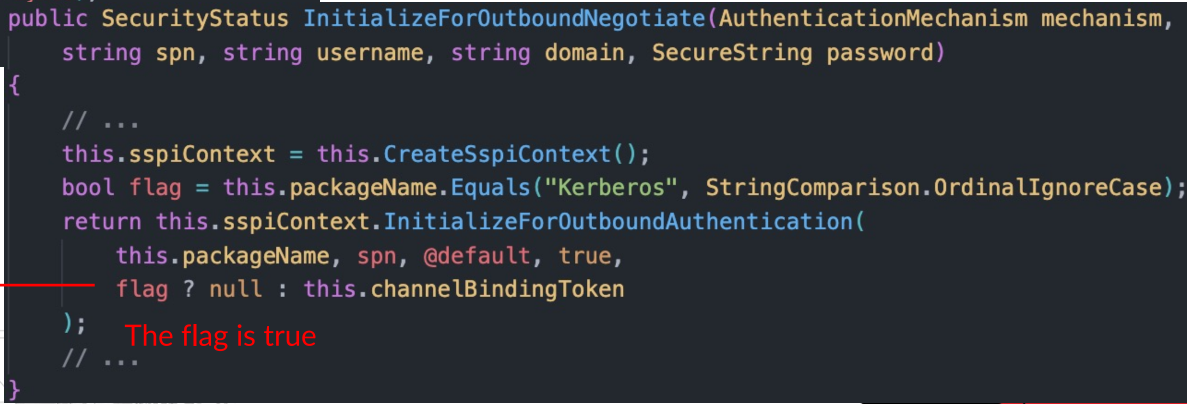
If EPA is set to accept or required, the frontend will fail (401) to authenticate to the backend

EPA on Back End Endpoints



```
internal static string GenerateKerberosAuthHeader(string host, int traceContext,
    ref AuthenticationContext authenticationContext, ref string kerberosChallenge)
{
    // ...
    authenticationContext = new AuthenticationContext();
    string text = Constants.SpnPrefixForHttp + host;
    // ...
    authenticationContext.InitializeForOutboundNegotiate(
        AuthenticationMechanism.Kerberos, text, null, null
    );
    // ...
    SecurityStatus securityStatus = authenticationContext.
                NegotiateSecurityContext(inputBuffer, out bytes);
    // ...
}
```

ProxyRequestHandler.PrepareServerRequest
|__ KerberosUtilities.GenerateKerberosAuthHeader
    |__ Authentication.InitializeForOutboundNegotiate



```
public SecurityStatus InitializeForOutboundNegotiate(AuthenticationMechanism mechanism,
    string spn, string username, string domain, SecureString password)
{
    // ...
    this.sspiContext = this.CreateSspiContext();
    bool flag = this.packageName.Equals("Kerberos", StringComparison.OrdinalIgnoreCase);
    return this.sspiContext.InitializeForOutboundAuthentication(
        this.packageName, spn, @default, true,
        flag ? null : this.channelBindingToken
    );
    // ...
}
```

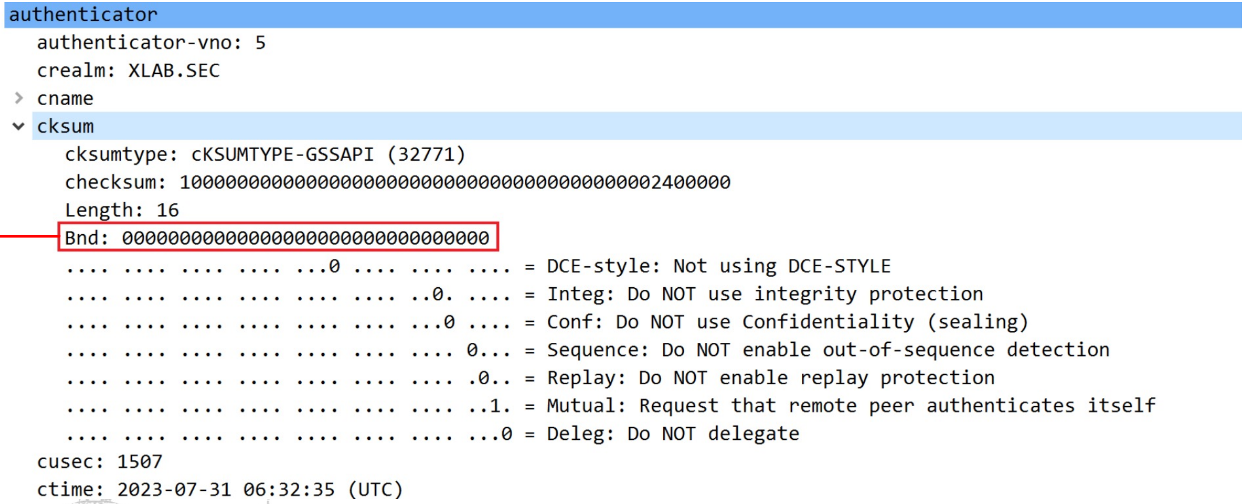
The flag is true

No channel binding token when generating Kerberos AP-REQ

EPA is not supported in the code level by design

EPA on Back End Endpoints

- No Channel Binding Token (CBT) when the Front End authenticates to the Back End

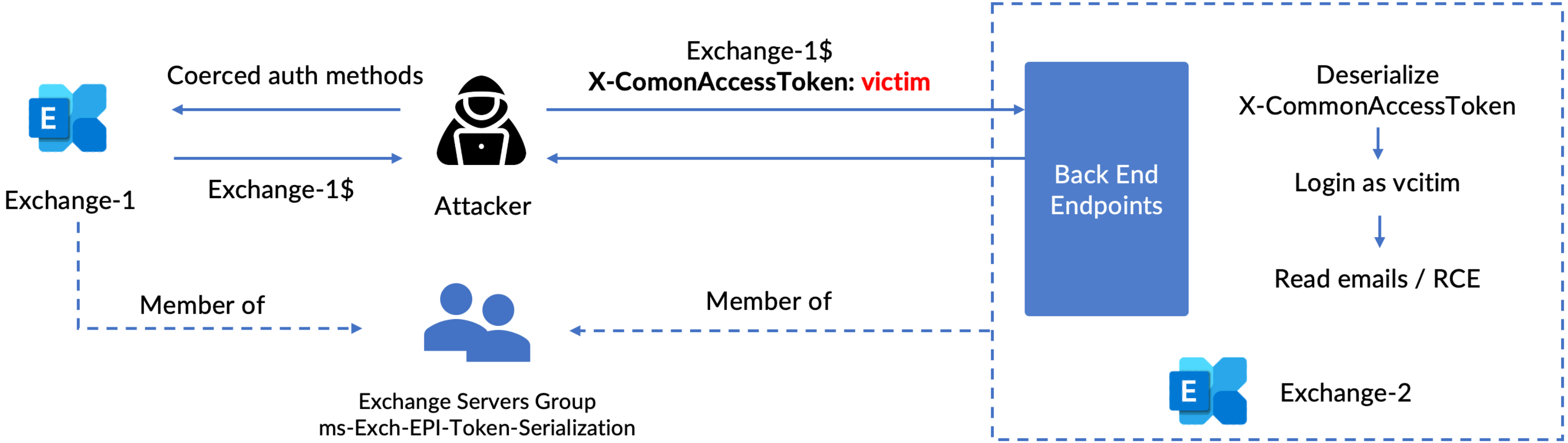


```
authenticator
    authenticator-vno: 5
    crealm: XLAB.SEC
  > cname
  ⌄ cksum
        cksumtype: cKSUMTYPE-GSSAPI (32771)
        checksum: 100000000000000000000000000000000000000002400000
        Length: 16
        Bnd: 00000000000000000000000000000000
        .... .... .... .... ...0 .... .... .... = DCE-style: Not using DCE-STYLE
        .... .... .... .... .... .... ..0. .... = Integ: Do NOT use integrity protection
        .... .... .... .... .... .... ...0 .... = Conf: Do NOT use Confidentiality (sealing)
        .... .... .... .... .... .... .... 0... = Sequence: Do NOT enable out-of-sequence detection
        .... .... .... .... .... .... .... .0.. = Replay: Do NOT enable replay protection
        .... .... .... .... .... .... .... ..1. = Mutual: Request that remote peer authenticates itself
        .... .... .... .... .... .... .... ...0 = Deleg: Do NOT delegate
    cusec: 1507
    ctime: 2023-07-31 06:32:35 (UTC)
```

Kerberos AP-REQ authenticator generated by the Front End, with CBT set to all zeros

NTLM Relay to the Back End - CVE-2022-24477

Attackers can perform NTLM relay to all Back-End endpoints, and impersonate arbitrary Exchange user with X-CommonAccessToken to read emails / RCE

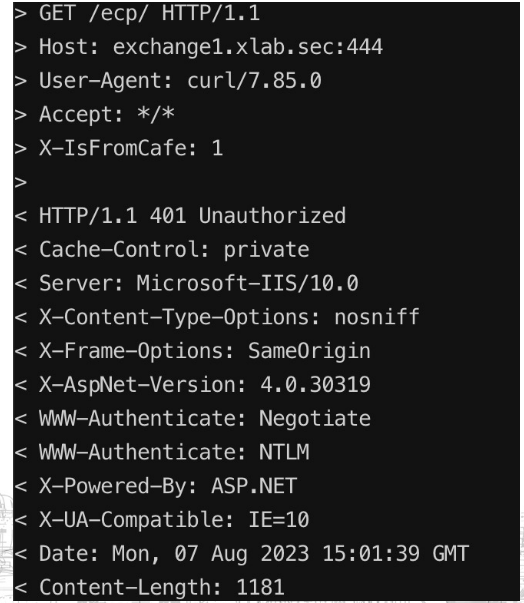


NTLM Relay to the Back End - CVE-2022-24477

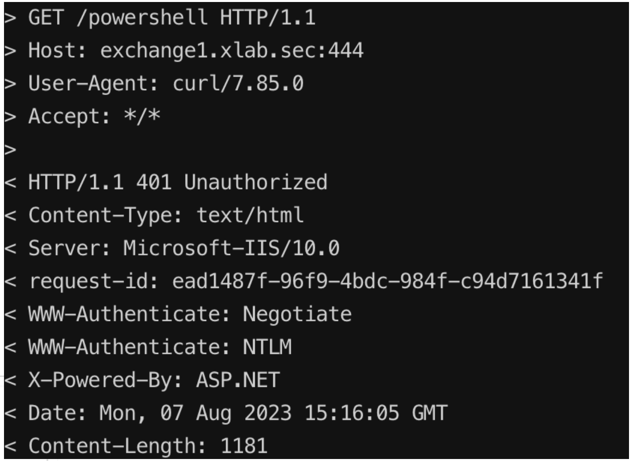
Backend Endpoints	Authentication Methods	Affected	Exploitable	Impact
/Powershell/	Kerberos, NTLM	✓	✓	Remote Code Execution
/ecp/	Kerberos, NTLM	✓	✓	Remote Code Execution
/EWS/	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/mapi/emsmdb	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/API/	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/owa/	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/Microsoft-Server-ActiveSync/Proxy	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/rpc/rpcproxy.dll	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/RpcWithCert/rpcproxy.dll	Kerberos, NTLM	✓	✓	Arbitrary Mailbox Takeover
/autodiscover/	Kerberos, NTLM	✓	-	-
/oab/	Kerberos, NTLM	✓	-	-
/PushNotifications/	Kerberos, NTLM	✓	-	-

NTLM Relay to the Back End – RCE

The /Powershell and /ECP are mainly for Exchange management, support NTLM authentication on the Back End



```
> GET /ecp/ HTTP/1.1
> Host: exchange1.xlab.sec:444
> User-Agent: curl/7.85.0
> Accept: */*
> X-IsFromCafe: 1
>
< HTTP/1.1 401 Unauthorized
< Cache-Control: private
< Server: Microsoft-IIS/10.0
< X-Content-Type-Options: nosniff
< X-Frame-Options: SameOrigin
< X-AspNet-Version: 4.0.30319
< WWW-Authenticate: Negotiate
< WWW-Authenticate: NTLM
< X-Powered-By: ASP.NET
< X-UA-Compatible: IE=10
< Date: Mon, 07 Aug 2023 15:01:39 GMT
< Content-Length: 1181
```



```
> GET /powershell HTTP/1.1
> Host: exchange1.xlab.sec:444
> User-Agent: curl/7.85.0
> Accept: */*
>
< HTTP/1.1 401 Unauthorized
< Content-Type: text/html
< Server: Microsoft-IIS/10.0
< request-id: ead1487f-96f9-4bdc-984f-c94d7161341f
< WWW-Authenticate: Negotiate
< WWW-Authenticate: NTLM
< X-Powered-By: ASP.NET
< Date: Mon, 07 Aug 2023 15:16:05 GMT
< Content-Length: 1181
```

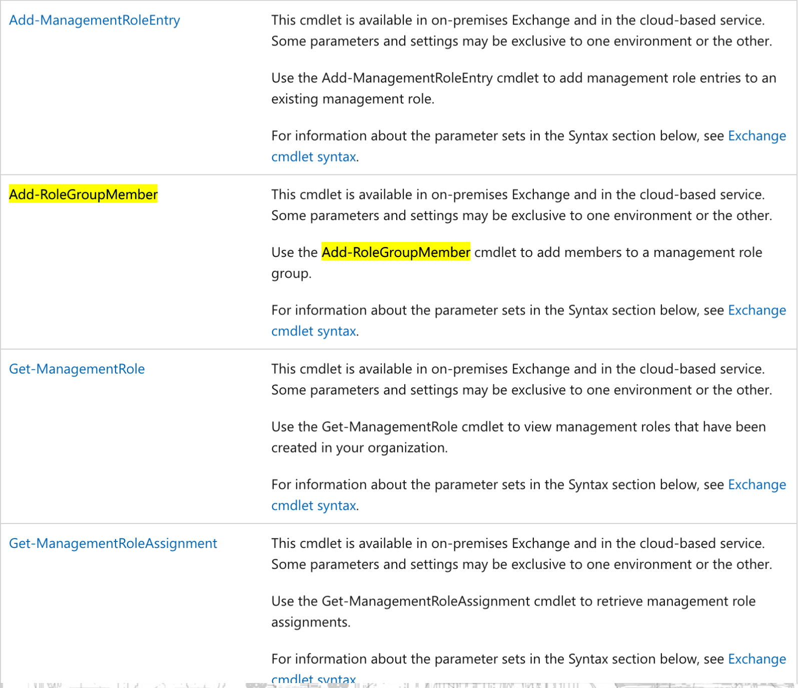
NTLM Relay to the Back End – Powershell

- Attackers can impersonate administrator on the Back-End /Powershell with X-CommonAccessToken, and execute arbitrary Exchange Cmdlet
- Exchange PowerShell doesn’t support to execute Windows commands like the native PowerShell of Windows, it only support cmdlets implemented by Exchange
- How to RCE? Maybe we need found a new Post-Auth RCE on Exchange Cmdlets?

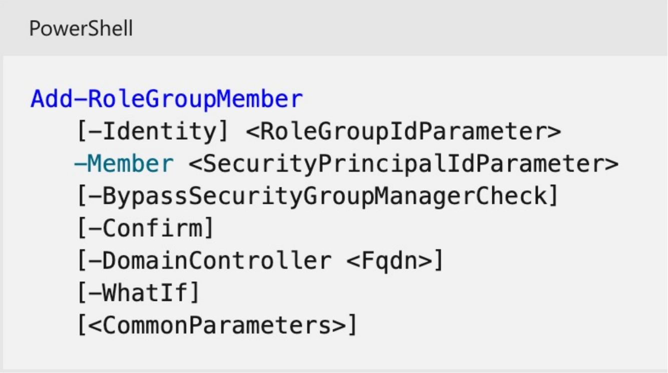
NTLM Relay to the Back End – Powershell

- Role-based access control management Cmdlets

role-based-access-control



Add-ManagementRoleEntry	This cmdlet is available in on-premises Exchange and in the cloud-based service. Some parameters and settings may be exclusive to one environment or the other. Use the Add-ManagementRoleEntry cmdlet to add management role entries to an existing management role. For information about the parameter sets in the Syntax section below, see Exchange cmdlet syntax.
Add-RoleGroupMember	This cmdlet is available in on-premises Exchange and in the cloud-based service. Some parameters and settings may be exclusive to one environment or the other. Use the Add-RoleGroupMember cmdlet to add members to a management role group. For information about the parameter sets in the Syntax section below, see Exchange cmdlet syntax.
Get-ManagementRole	This cmdlet is available in on-premises Exchange and in the cloud-based service. Some parameters and settings may be exclusive to one environment or the other. Use the Get-ManagementRole cmdlet to view management roles that have been created in your organization. For information about the parameter sets in the Syntax section below, see Exchange cmdlet syntax.
Get-ManagementRoleAssignment	This cmdlet is available in on-premises Exchange and in the cloud-based service. Some parameters and settings may be exclusive to one environment or the other. Use the Get-ManagementRoleAssignment cmdlet to retrieve management role assignments. For information about the parameter sets in the Syntax section below, see Exchange cmdlet syntax

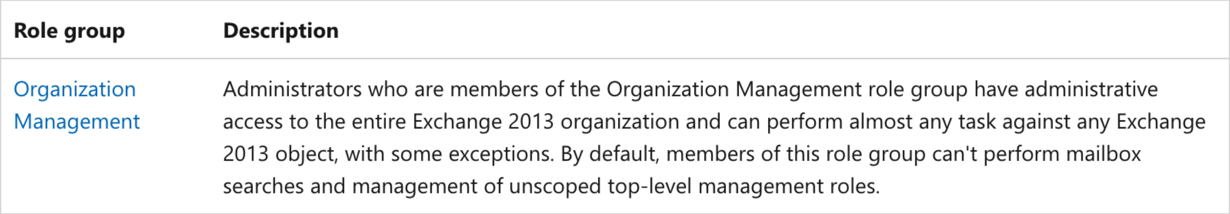


PowerShell

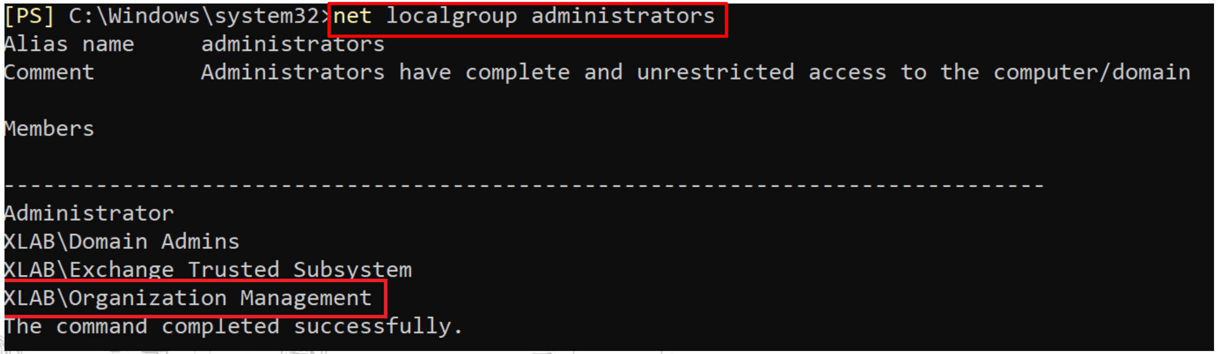
```
Add-RoleGroupMember
    [-Identity] <RoleGroupIdParameter>
    -Member <SecurityPrincipalIdParameter>
    [-BypassSecurityGroupManagerCheck]
    [-Confirm]
    [-DomainController <Fqdn>]
    [-WhatIf]
    [<CommonParameters>]
```

NTLM Relay to the Back End – Powershell

“Organization Management” is one of these built-in role groups, which is also a member of local administrators of Exchange Servers



Role group	Description
Organization Management	Administrators who are members of the Organization Management role group have administrative access to the entire Exchange 2013 organization and can perform almost any task against any Exchange 2013 object, with some exceptions. By default, members of this role group can't perform mailbox searches and management of unscoped top-level management roles.

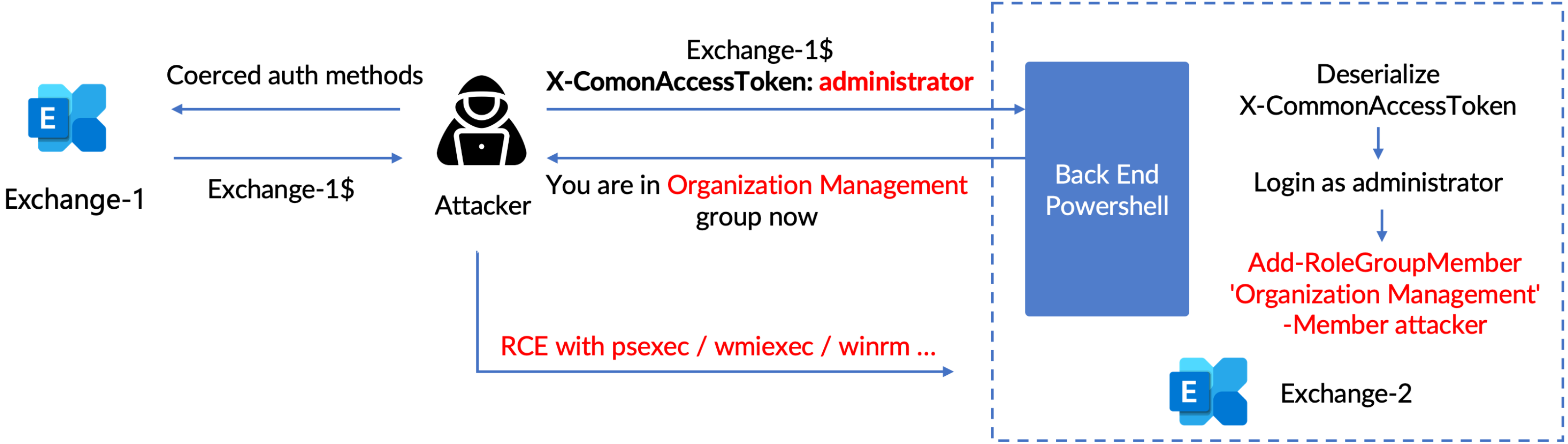


```
[PS] C:\Windows\system32>net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
XLAB\Domain Admins
XLAB\Exchange Trusted Subsystem
XLAB\Organization Management
The command completed successfully.
```

Add-RoleGroupMember ’Organization Management’ -Member attacker

NTLM Relay to the Back End Powershell - RCE

RCE with PsExec, WmiExec, WinRM ...



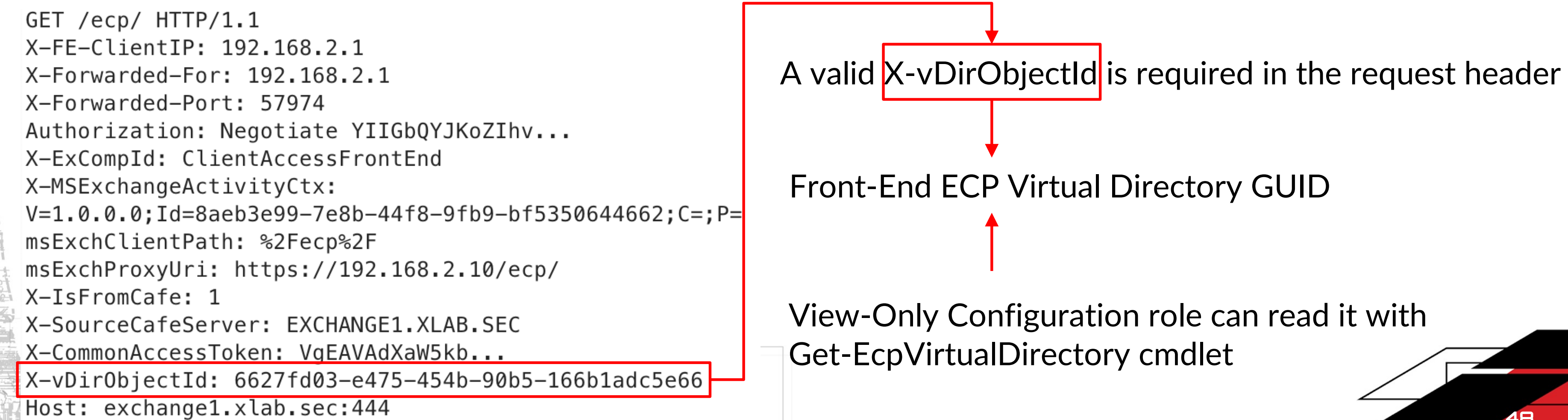
Exchange-1
Coerced auth methods
Exchange-1$
Attacker
Exchange-1$
X-ComonAccessToken: administrator
You are in Organization Management group now
Back End Powershell
Deserialize X-CommonAccessToken
Login as administrator
Add-RoleGroupMember 'Organization Management' -Member attacker
RCE with psexec / wmiexec / winrm ...
Exchange-2

NTLM Relay to the Back End Powershell - RCE

DEMO: https://youtu.be/Y7uVtfZ3jcU

NTLM Relay to the Back End – ECP

- Two methods to impersonate other users on the Back End ECP
  - X-CommonAccessToken HTTP Header
  - /ecp/ProxyLogon.ecp create new user token from the XML in POST body
    - Also used by the well-known ProxyLogon vulnerability



```
GET /ecp/ HTTP/1.1
X-FE-ClientIP: 192.168.2.1
X-Forwarded-For: 192.168.2.1
X-Forwarded-Port: 57974
Authorization: Negotiate YIIGbQYJKoZIhv...
X-ExCompId: ClientAccessFrontEnd
X-MSExchangeActivityCtx:
V=1.0.0.0;Id=8aeb3e99-7e8b-44f8-9fb9-bf5350644662;C=;P=
msExchClientPath: %2Fecp%2F
msExchProxyUri: https://192.168.2.10/ecp/
X-IsFromCafe: 1
X-SourceCafeServer: EXCHANGE1.XLAB.SEC
X-CommonAccessToken: VgEAVAdXaW5kb...
X-vDirObjectId: 6627fd03-e475-454b-90b5-166b1adc5e66
Host: exchange1.xlab.sec:444
```

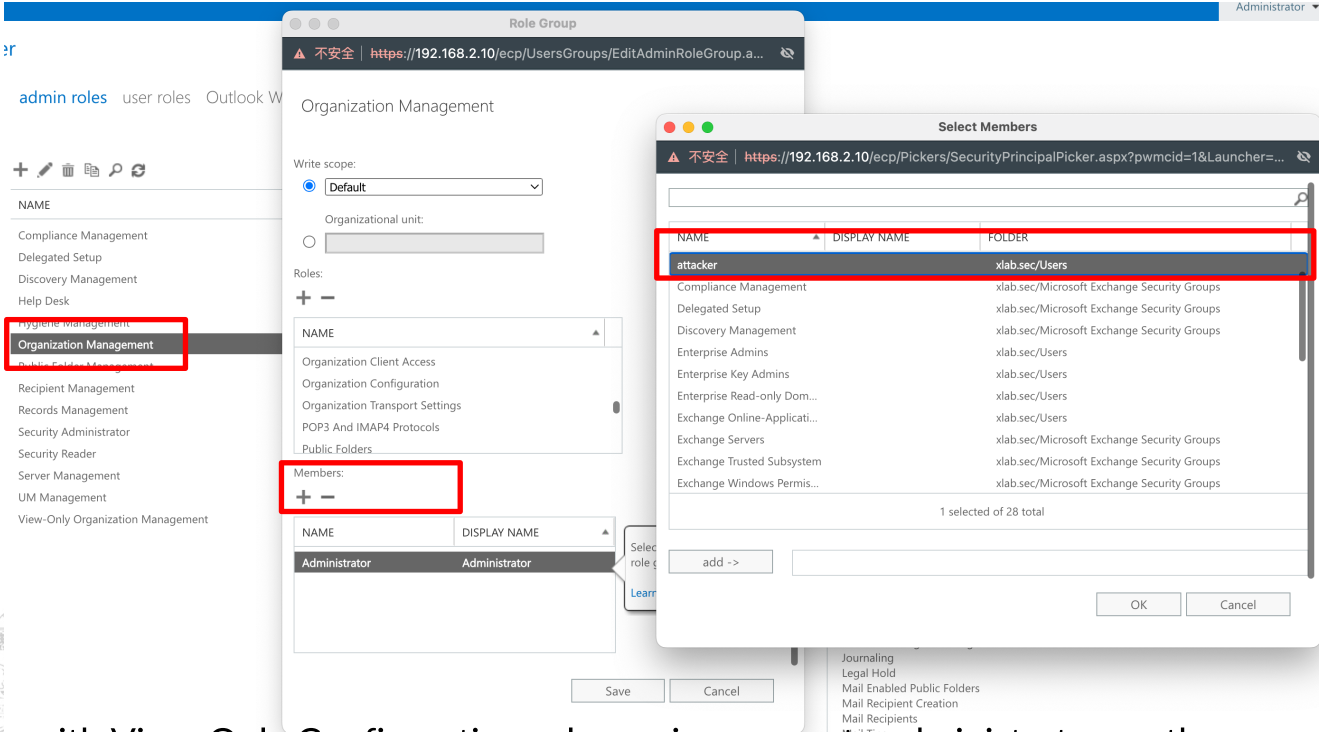
A valid X-vDirObjectId is required in the request header

Front-End ECP Virtual Directory GUID

View-Only Configuration role can read it with Get-EcpVirtualDirectory cmdlet

NTLM Relay to the Back End ECP - RCE

ECP also support add users to Organization Management group



Administrator ▾

er

admin roles   user roles   Outlook W

NAME

- Compliance Management
- Delegated Setup
- Discovery Management
- Help Desk
- Hygiene Management
- Organization Management
- Public Folder Management
- Recipient Management
- Records Management
- Security Administrator
- Security Reader
- Server Management
- UM Management
- View-Only Organization Management

Role Group

⚠ 不安全 | https://192.168.2.10/ecp/UsersGroups/EditAdminRoleGroup.a...

Organization Management

Write scope:

● Default

Organizational unit:

○

Roles:

NAME
Organization Client Access
Organization Configuration
Organization Transport Settings
POP3 And IMAP4 Protocols
Public Folders

Members:

NAME	DISPLAY NAME
Administrator	Administrator

Selec role g Learn

Save   Cancel

Select Members

⚠ 不安全 | https://192.168.2.10/ecp/Pickers/SecurityPrincipalPicker.aspx?pwmcid=1&Launcher=...

NAME	DISPLAY NAME	FOLDER
attacker		xlab.sec/Users
Compliance Management		xlab.sec/Microsoft Exchange Security Groups
Delegated Setup		xlab.sec/Microsoft Exchange Security Groups
Discovery Management		xlab.sec/Microsoft Exchange Security Groups
Enterprise Admins		xlab.sec/Users
Enterprise Key Admins		xlab.sec/Users
Enterprise Read-only Dom...		xlab.sec/Users
Exchange Online-Applicati...		xlab.sec/Users
Exchange Servers		xlab.sec/Microsoft Exchange Security Groups
Exchange Trusted Subsystem		xlab.sec/Microsoft Exchange Security Groups
Exchange Windows Permis...		xlab.sec/Microsoft Exchange Security Groups

1 selected of 28 total

add ->

OK   Cancel

Journaling
Legal Hold
Mail Enabled Public Folders
Mail Recipient Creation
Mail Recipients

Attackers with View-Only Configuration role can impersonate administrator on the Back-End ECP with NTLM relay and add himself to Organization Management, and achieve RCE on Exchange Servers with PsExec, WmiExec, WinRM ...

Relay to the Back End - CVE-2022-24477

DEMOS: https://www.youtube.com/playlist?list=PLtZO9vwOND92_EcfyXo90lHtLng8aIEQT

Patch – Extended Protection

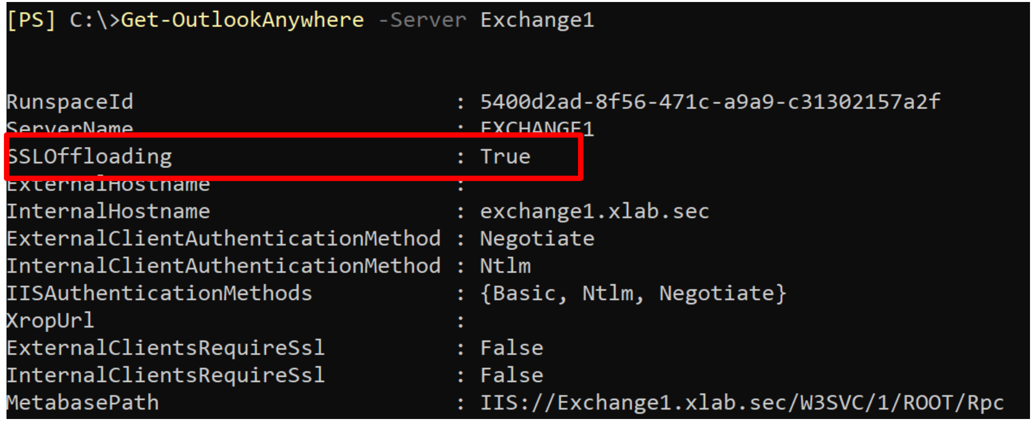
- Extended Protection is supported on Exchange Server 2013, 2016 and 2019 starting with the August 2022 Exchange Server Security Update (SU) releases.
- This protection is accomplished by Channel Binding Token (CBT) and mainly for SSL connections
- Customers need to enable the Extended Protection manually
- All exploitable Front End endpoints and Back End endpoints are recommended to enabled Extended Protection

Patch Guide: https://microsoft.github.io/CSS-Exchange/Security/Extended-Protection/
Patch Script: ExchangeExtendedProtectionManagement.ps1

NTLM Relay to the Front End /RPC over HTTP(80)

Extended Protection only protects HTTPS connections

SSLOffLoading is enabled by default in the Front End /RPC, which means /RPC endpoint also supports HTTP 80



```
[PS] C:\>Get-OutlookAnywhere -Server Exchange1


RunspaceId                         : 5400d2ad-8f56-471c-a9a9-c31302157a2f
ServerName                         : EXCHANGE1
SSLOffloading                      : True
ExternalHostname                   :
InternalHostname                   : exchange1.xlab.sec
ExternalClientAuthenticationMethod : Negotiate
InternalClientAuthenticationMethod : Ntlm
IISAuthenticationMethods           : {Basic, Ntlm, Negotiate}
XropUrl                            :
ExternalClientsRequireSsl          : False
InternalClientsRequireSsl          : False
MetabasePath                       : IIS://Exchange1.xlab.sec/W3SVC/1/ROOT/Rpc
```

CVE-2022-24516

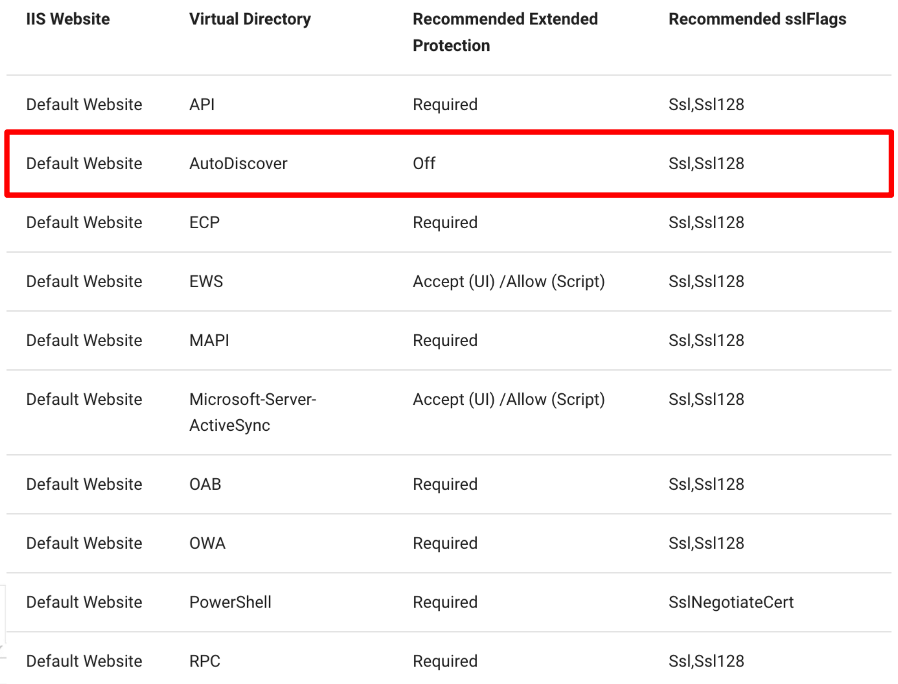
NTLM relay to the Front End /RPC on HTTP 80

can also lead to arbitrary mailbox takeover

The patch script ExtendedProtection ExchangeExtendedProtectionManagement.ps1 will turn off the SSLOffLoading for the Front End /RPC when enable the Extended Protection

Patch Bypass ?

The Extended Protection is still not enabled on the frontend and backend AutoDiscover

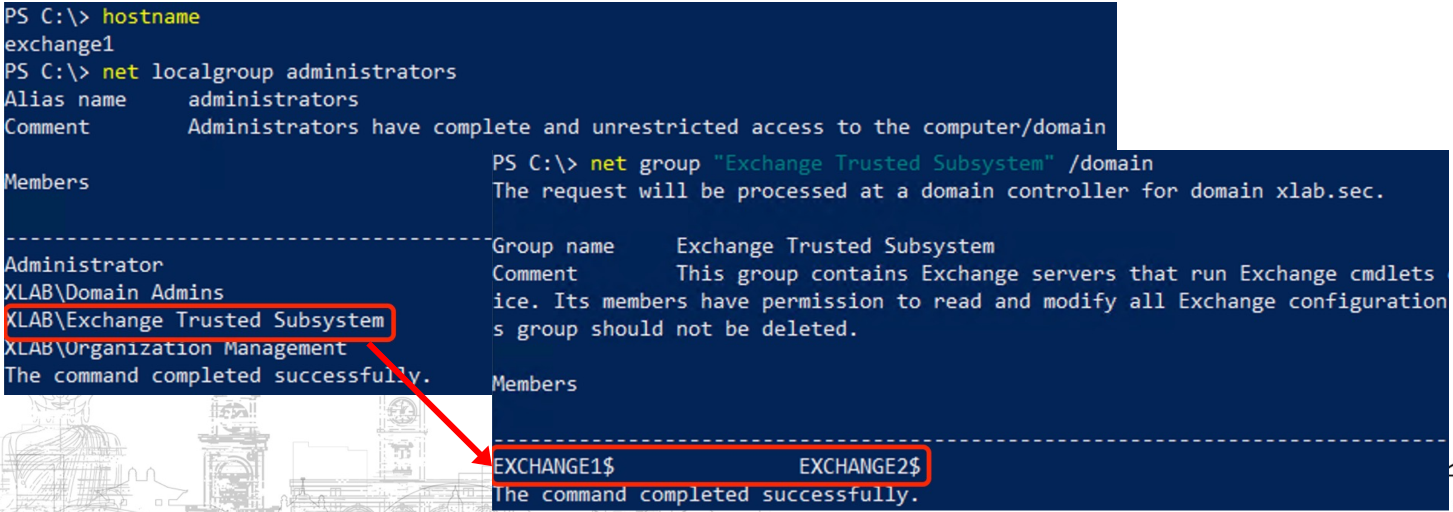


IIS Website	Virtual Directory	Recommended Extended Protection	Recommended sslFlags
Default Website	API	Required	Ssl,Ssl128
Default Website	AutoDiscover	Off	Ssl,Ssl128
Default Website	ECP	Required	Ssl,Ssl128
Default Website	EWS	Accept (UI) /Allow (Script)	Ssl,Ssl128
Default Website	MAPI	Required	Ssl,Ssl128
Default Website	Microsoft-Server-ActiveSync	Accept (UI) /Allow (Script)	Ssl,Ssl128
Default Website	OAB	Required	Ssl,Ssl128
Default Website	OWA	Required	Ssl,Ssl128
Default Website	PowerShell	Required	SslNegotiateCert
Default Website	RPC	Required	Ssl,Ssl128

If you found an SSRF on AutoDiscover endpoint (like ProxyNotShell), this attack will still work

Exchange Server Machine Account

- All members of Exchange Trusted Subsystem have local administrator privileges on all Exchange Servers.
- All Exchange machine accounts will be added to this group during Exchange Server installation.

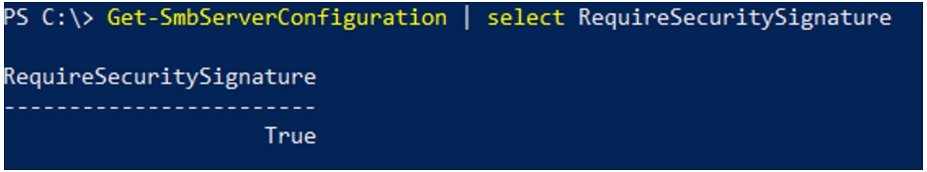


```
PS C:\> hostname
exchange1
PS C:\> net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
XLAB\Domain Admins
XLAB\Exchange Trusted Subsystem
XLAB\Organization Management
The command completed successfully.
```

```
PS C:\> net group "Exchange Trusted Subsystem" /domain
The request will be processed at a domain controller for domain xlab.sec.

Group name     Exchange Trusted Subsystem
Comment        This group contains Exchange servers that run Exchange cmdlets
ice. Its members have permission to read and modify all Exchange configuration
s group should not be deleted.

Members

-------------------------------------------------------------------------------
EXCHANGE1$                 EXCHANGE2$
The command completed successfully.
```

NTLM Relay to Windows Services

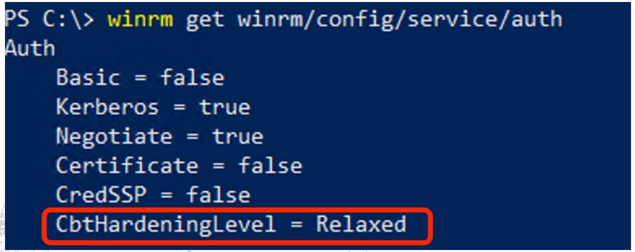
- MS-RPC over SMB (ncacn_np)



```
PS C:\> Get-SmbServerConfiguration | select RequireSecuritySignature

RequireSecuritySignature
------------------------
                    True
```

SMB signing is enabled by default
On Exchange Servers

- WinRM (Powershell Remoting)



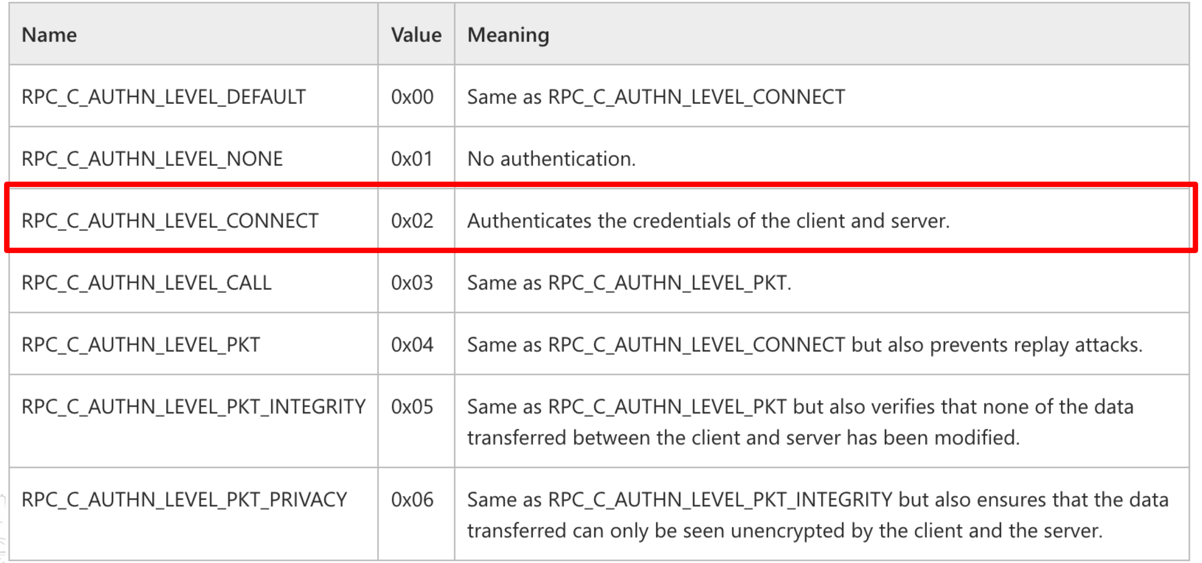
```
PS C:\> winrm get winrm/config/service/auth
Auth
    Basic = false
    Kerberos = true
    Negotiate = true
    Certificate = false
    CredSSP = false
    CbtHardeningLevel = Relaxed
```

HTTP: Signing and Sealing are required
HTTPS: EPA Channel Binding is enabled

- MS-RPC over TCP (ncacn_ip_tcp)
    - Many RPC interfaces support ncacn_ip_tcp transport , MS-PAR， MS-TSCH， DCOM， WMI ...
    - Each RPC interface implement its own security policies

NTLM Relay to MS-RPC (ncacn_ip_tcp)

- RPC clients can set the auth type to RPC_C_AUTHN_WINNT to use NTLMSSP
- RPC authentication level and RPC signing / sealing
  - The RPC_C_AUTHN_LEVEL_CONNECT authentication level indicates that the RPC connection does not need to be sealed and signed



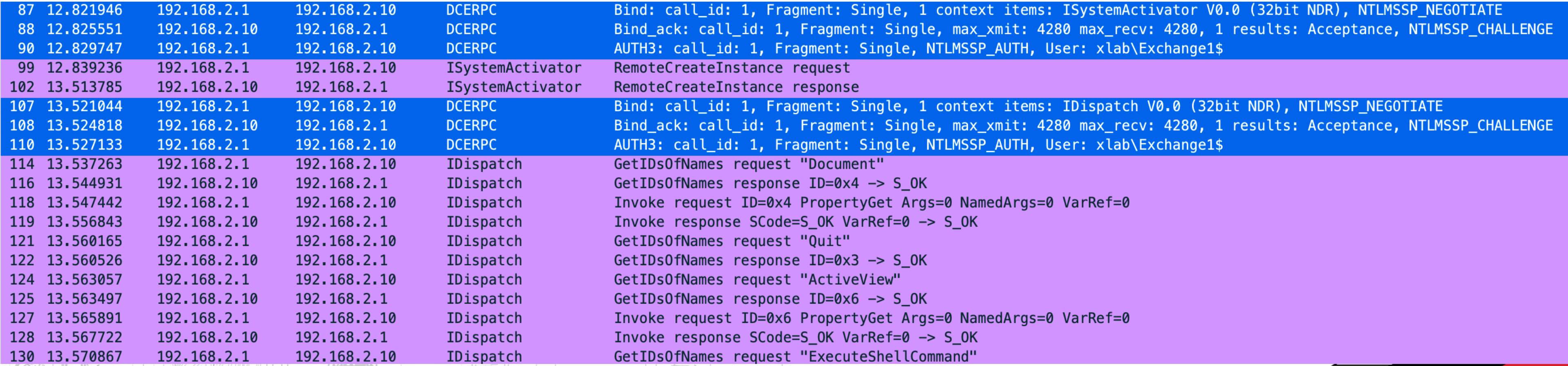
Name	Value	Meaning
RPC_C_AUTHN_LEVEL_DEFAULT	0x00	Same as RPC_C_AUTHN_LEVEL_CONNECT
RPC_C_AUTHN_LEVEL_NONE	0x01	No authentication.
RPC_C_AUTHN_LEVEL_CONNECT	0x02	Authenticates the credentials of the client and server.
RPC_C_AUTHN_LEVEL_CALL	0x03	Same as RPC_C_AUTHN_LEVEL_PKT.
RPC_C_AUTHN_LEVEL_PKT	0x04	Same as RPC_C_AUTHN_LEVEL_CONNECT but also prevents replay attacks.
RPC_C_AUTHN_LEVEL_PKT_INTEGRITY	0x05	Same as RPC_C_AUTHN_LEVEL_PKT but also verifies that none of the data transferred between the client and server has been modified.
RPC_C_AUTHN_LEVEL_PKT_PRIVACY	0x06	Same as RPC_C_AUTHN_LEVEL_PKT_INTEGRITY but also ensures that the data transferred can only be seen unencrypted by the client and the server.

NTLM Relay to DCOM

- DCOM utilizes MS-RPC (ncacn_ip_tcp) as its underlying communication protocol to enable remote COM object communication.
- Signing and sealing are not force enabled on DCOM servers
- DCOM clients can set the RPC authentication level to RPC_C_AUTHN_LEVEL_CONNECT to avoid signing and sealing, which can disable the protection for NTLM relay
- DCOM clients communicate with RPC servers using port 135 and a dynamic port assigned by EPM(endpoint mapper), both connections require NTLM authentication

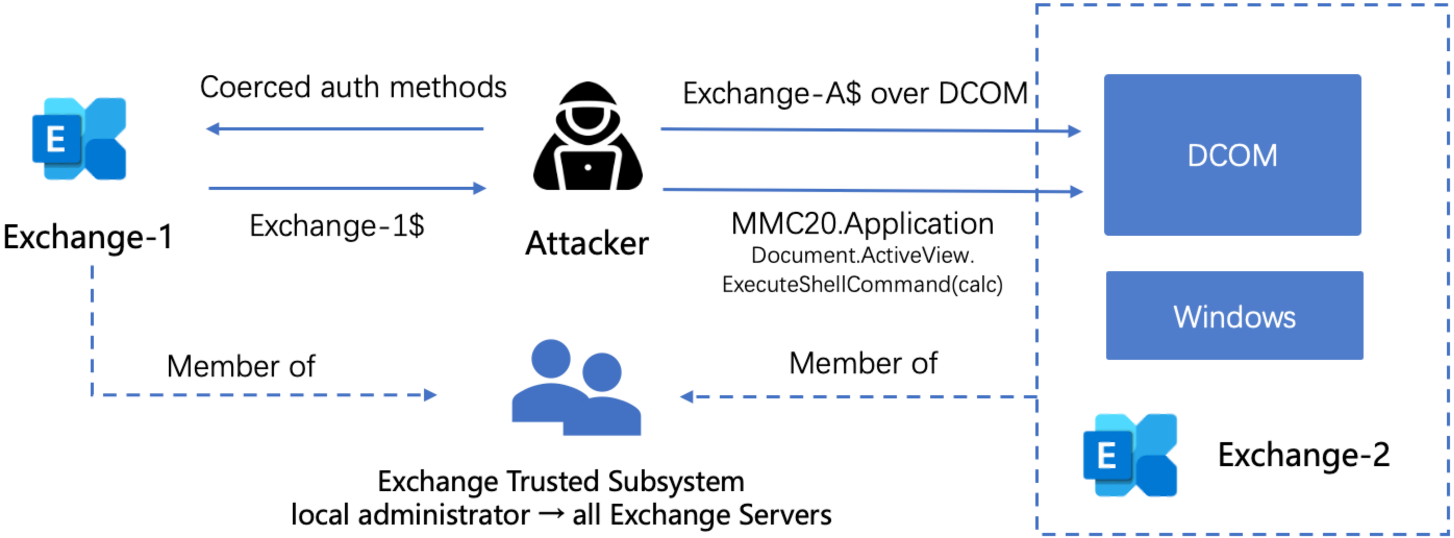
NTLM Relay to DCOM

- MMC20.Application COM object
  - CLSID 49B2791A-B1AE-4C90-9B8E-E860BA07F889
  - has the Document.ActiveView.ExecuteShellCommand method which support to execute commands remotely



```
 87  12.821946   192.168.2.1    192.168.2.10    DCERPC            Bind: call_id: 1, Fragment: Single, 1 context items: ISystemActivator V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
 88  12.825551   192.168.2.10   192.168.2.1     DCERPC            Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
 90  12.829747   192.168.2.1    192.168.2.10    DCERPC            AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: xlab\Exchange1$
 99  12.839236   192.168.2.1    192.168.2.10    ISystemActivator  RemoteCreateInstance request
102  13.513785   192.168.2.10   192.168.2.1     ISystemActivator  RemoteCreateInstance response
107  13.521044   192.168.2.1    192.168.2.10    DCERPC            Bind: call_id: 1, Fragment: Single, 1 context items: IDispatch V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
108  13.524818   192.168.2.10   192.168.2.1     DCERPC            Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
110  13.527133   192.168.2.1    192.168.2.10    DCERPC            AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: xlab\Exchange1$
114  13.537263   192.168.2.1    192.168.2.10    IDispatch         GetIDsOfNames request "Document"
116  13.544931   192.168.2.10   192.168.2.1     IDispatch         GetIDsOfNames response ID=0x4 -> S_OK
118  13.547442   192.168.2.1    192.168.2.10    IDispatch         Invoke request ID=0x4 PropertyGet Args=0 NamedArgs=0 VarRef=0
119  13.556843   192.168.2.10   192.168.2.1     IDispatch         Invoke response SCode=S_OK VarRef=0 -> S_OK
121  13.560165   192.168.2.1    192.168.2.10    IDispatch         GetIDsOfNames request "Quit"
122  13.560526   192.168.2.10   192.168.2.1     IDispatch         GetIDsOfNames response ID=0x3 -> S_OK
124  13.563057   192.168.2.1    192.168.2.10    IDispatch         GetIDsOfNames request "ActiveView"
125  13.563497   192.168.2.10   192.168.2.1     IDispatch         GetIDsOfNames response ID=0x6 -> S_OK
127  13.565891   192.168.2.1    192.168.2.10    IDispatch         Invoke request ID=0x6 PropertyGet Args=0 NamedArgs=0 VarRef=0
128  13.567722   192.168.2.10   192.168.2.1     IDispatch         Invoke response SCode=S_OK VarRef=0 -> S_OK
130  13.570867   192.168.2.1    192.168.2.10    IDispatch         GetIDsOfNames request "ExecuteShellCommand"
```

NTLM Relay to DCOM – Exchange Server RCE



NTLM relay to DCOM – Exchange Server RCE

DEMO: https://youtu.be/AByIzLx7RiQ

Patch – CVE-2021-26414

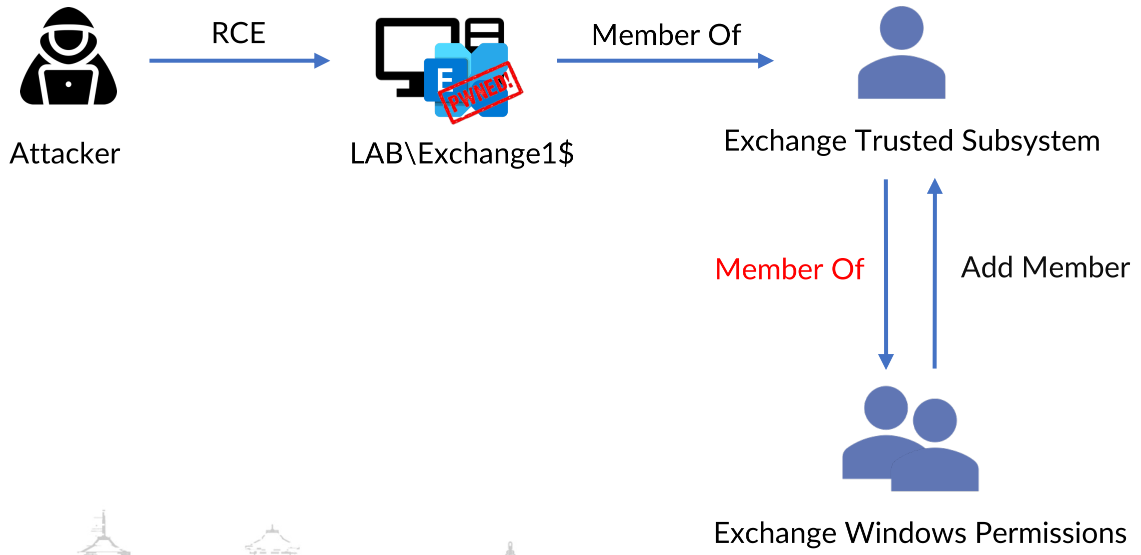
- The patch for this vulnerability was released on Patch Tuesday in June 2021.
  - The minimum authentication level required by DCOM is set to RPC_C_AUTHN_LEVEL_PKT_INTEGRITY
  - But this patch is not enabled by default, customers need to manually set "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Ole\AppCompat RequireIntegrityActivationAuthenticationLevel = 1" to active the patch
- June 2022, Microsoft released a security update to enable the patch by default, but still with the ability to disable it using the registry key.
- March 2023, the patch is enabled by default with no ability to disable it.

Privilege Escalation to Domain Admin

PrivExchange (fixed in 2019)

- Exchange EWS has a feature which can make it authenticate to an attacker with the Exchange machine account
- The Exchange machine account is a member of the Exchange Windows Permissions group
- The Exchange Windows Permissions group has WriteDACL access on the Domain object in Active Directory, an attackers can use these privileges to grant himself DCSync rights
- NTLM relay from HTTP to LDAP to escalate from a mailbox user to Domain Admin

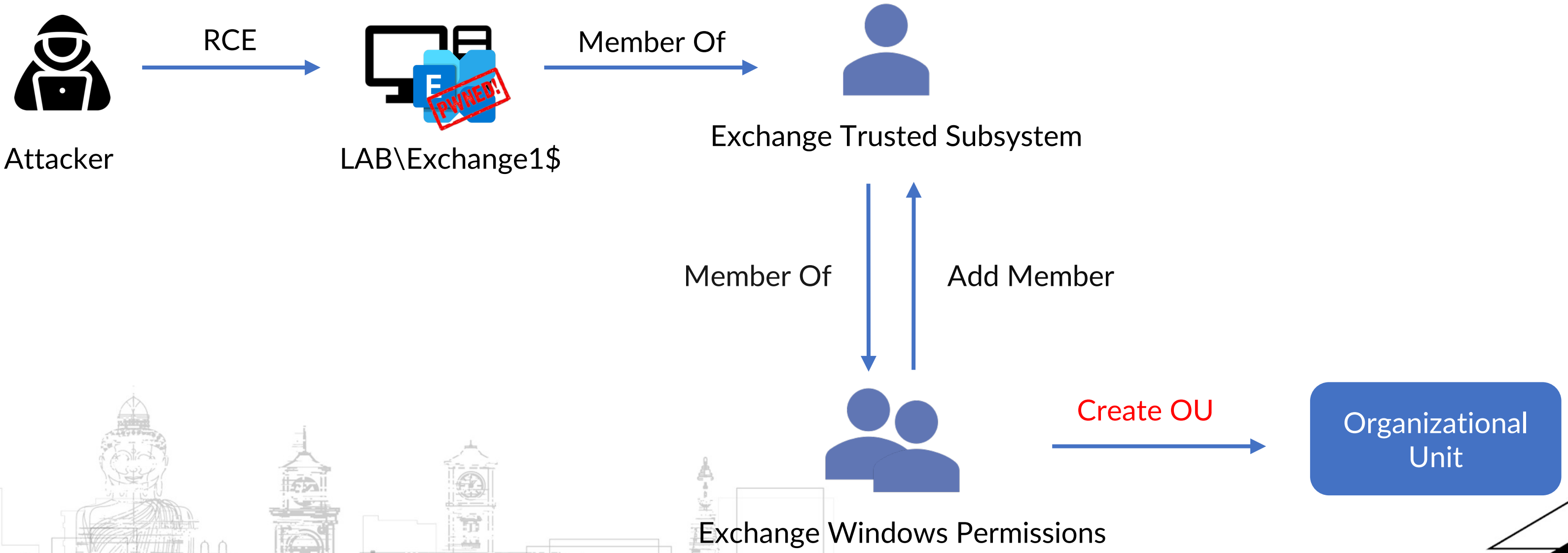
New Attack Path



New Attack Path

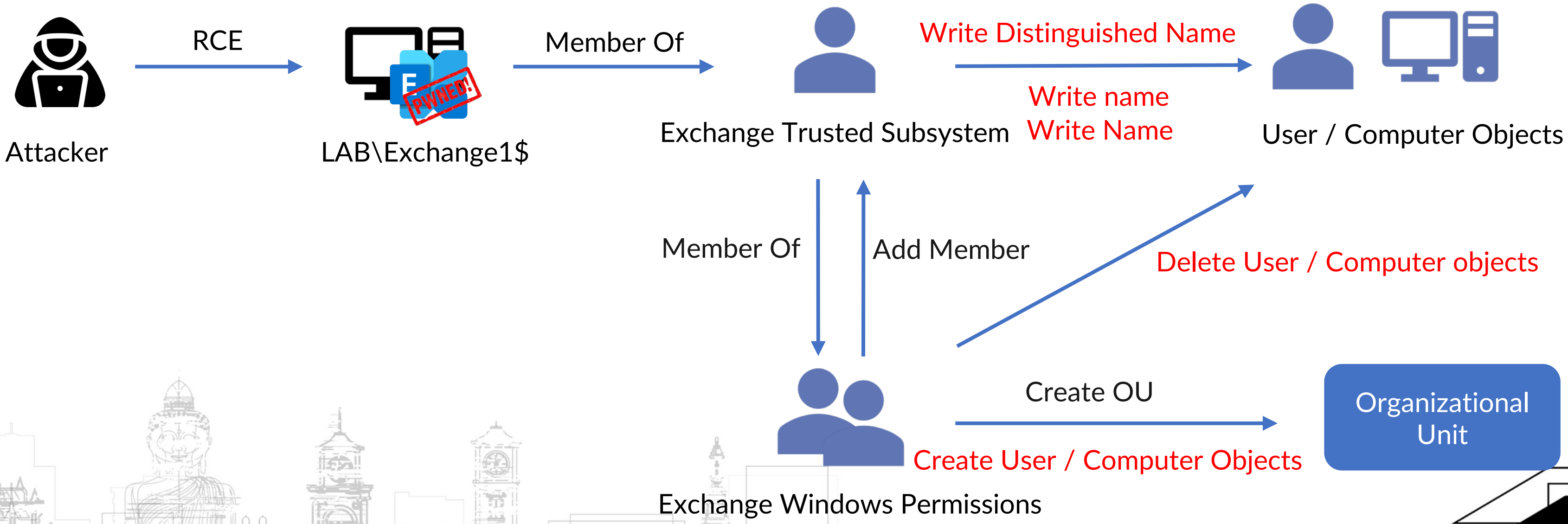
Exchange Windows Permission group has privileges to create new OUs in the Active Directory

The attacker can create a new OU and have full control on this OU



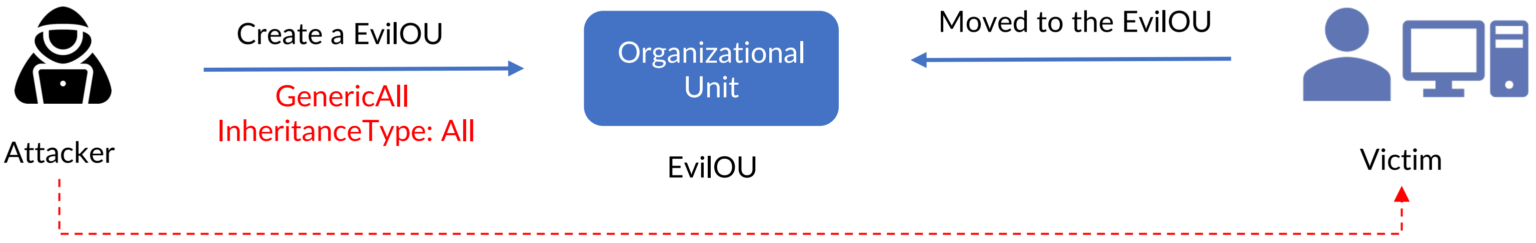
New Attack Path

The attacker can move arbitrary User / Computer object (except adminCount=1) to the newly created OU



New Attack Path

The attacker can set arbitrary ACEs with InheritanceType set to All on the newly created OU, these ACEs will inherit down to all descendent objects.



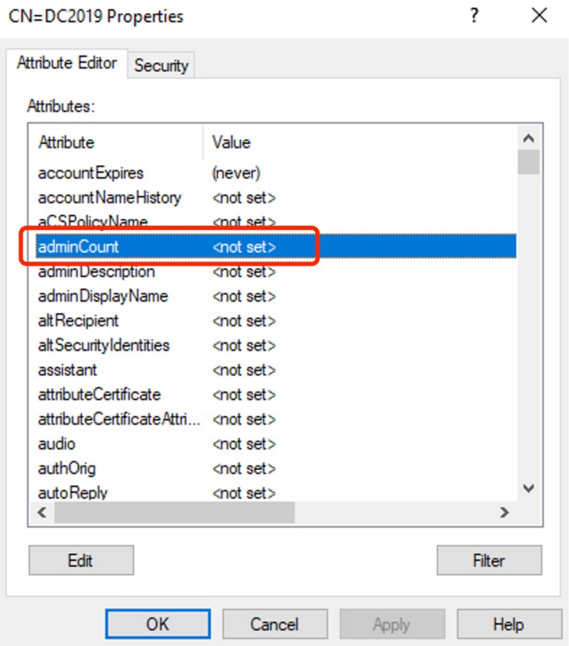
The victim will inherit ACEs from the EvilOU

GenericAll

Take over users / computers with ShadowCredentials / RBCD attack

RCE on Domain Controllers

Domain Controller computers don‘t set adminCount by default



CN=DC2019 Properties

Attribute	Value
accountExpires	(never)
accountNameHistory	<not set>
aCSPolicyName	<not set>
adminCount	<not set>
adminDescription	<not set>
adminDisplayName	<not set>
altRecipient	<not set>
altSecurityIdentities	<not set>
assistant	<not set>
attributeCertificate	<not set>
attributeCertificateAttri...	<not set>
audio	<not set>
authOrig	<not set>
autoReply	<not set>

Move domain controller computers to the EvilOU

Set GenericAll on domain controller computers

RBCD / ShadowCredentials attack

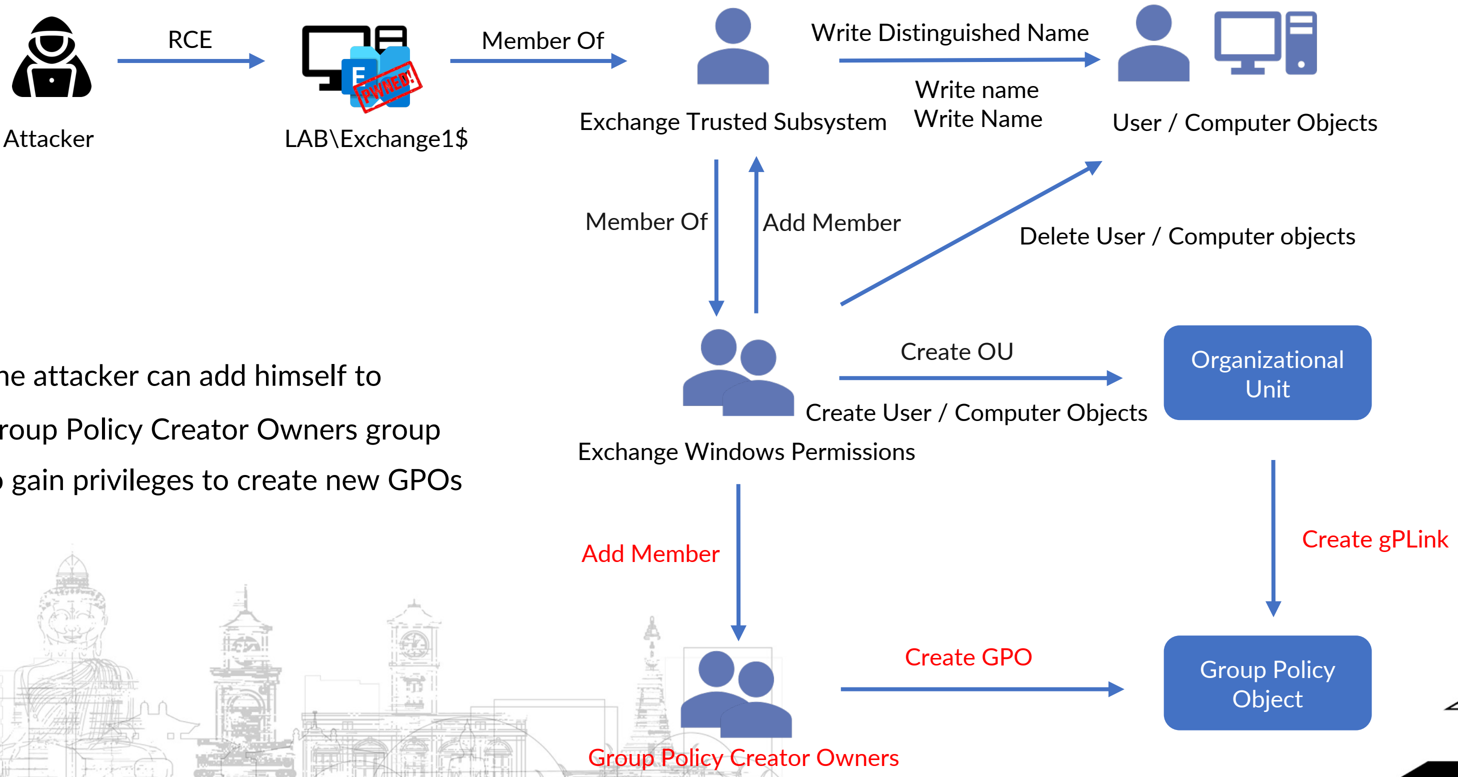
RCE on domain controllers

Domain must have at least one DC running at least Windows 2012

RCE on Domain Controllers

DEMO: https://youtu.be/GsTfIAw5WFY

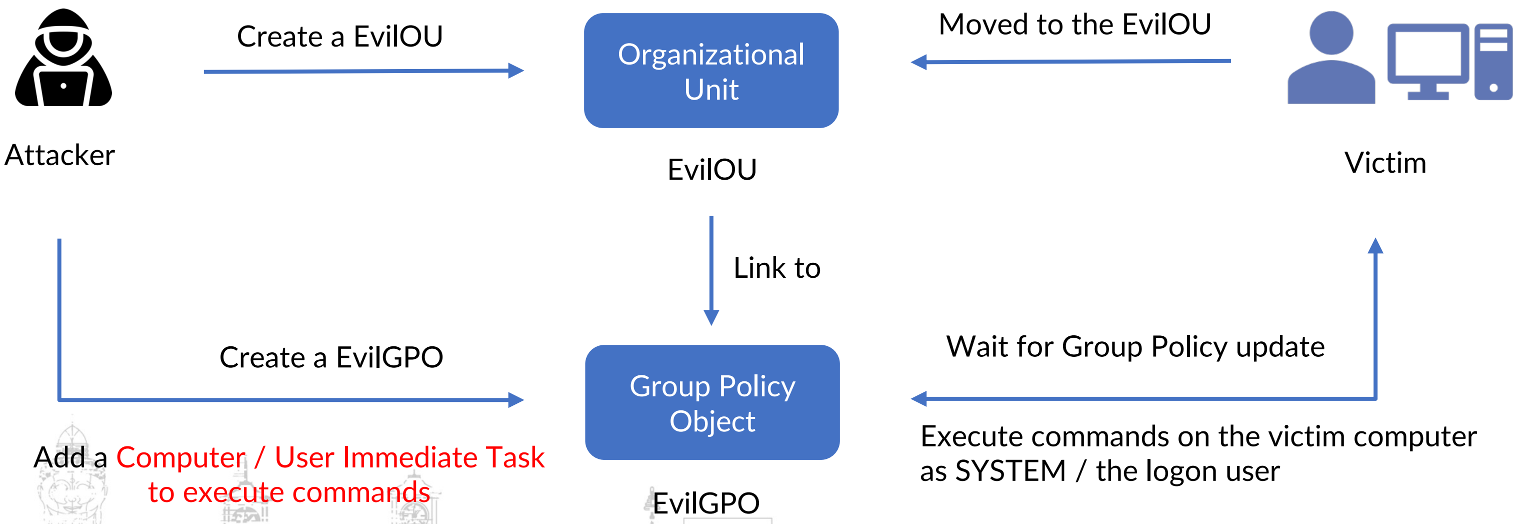
New Attack Path



The attacker can add himself to Group Policy Creator Owners group to gain privileges to create new GPOs

New Attack Path

The attacker can create a new GPO and link it to the newly create OU, the GPO will take effect on the objects in the OU.


Attacker
Create a EvilOU
Organizational Unit
EvilOU
Moved to the EvilOU
Victim
Link to
Create a EvilGPO
Group Policy Object
EvilGPO
Wait for Group Policy update
Add a Computer / User Immediate Task to execute commands
Execute commands on the victim computer as SYSTEM / the logon user

This attack path can also lead to remote commands execution as SYSTEM on domain controllers.
No requirement for the domain controller version.

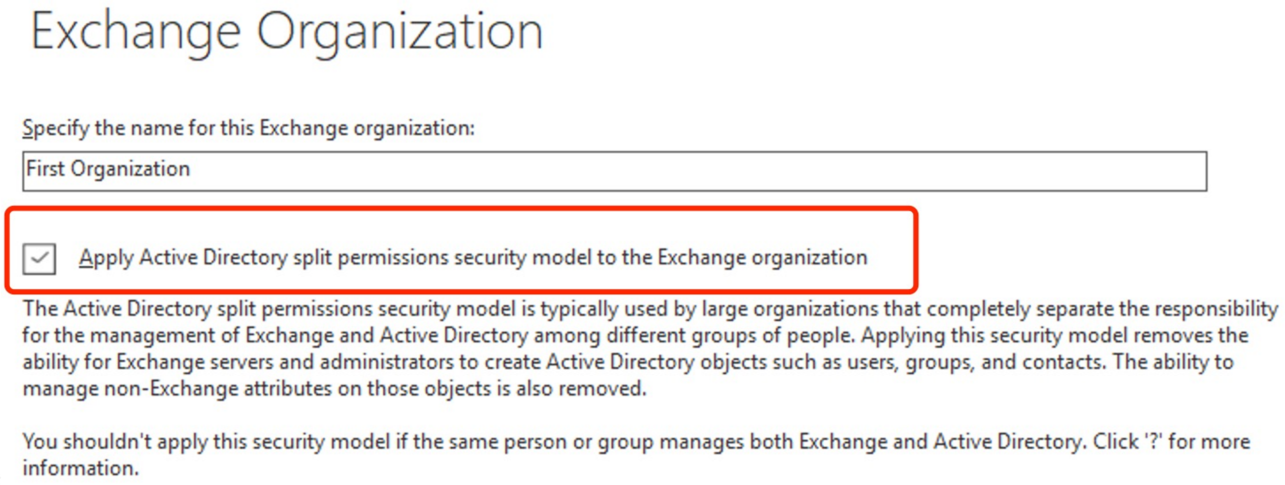
NTLM Relay

Combined with NTLM Relay ?

- Need another vulnerability to trigger NTLM authentication of Exchange machine account over HTTP(s)
  - You can also perform RBCD attack on Exchange Servers, but this attack path can help you escalate to Domain Admin
- NTLM relay from HTTP to LDAP (just like what PrivExchange did) to add the attacker to the following high-privileged groups
  - Exchange Trusted Subsystem
  - Group Policy Creator Owners

Won't Fix

- Microsoft won't fix this privilege escalation method
- Apply Active Directory split permissions model (not enabled by default) to Exchange can protect your Active Directory



Exchange Organization

Specify the name for this Exchange organization:

First Organization

☑ Apply Active Directory split permissions security model to the Exchange organization

The Active Directory split permissions security model is typically used by large organizations that completely separate the responsibility for the management of Exchange and Active Directory among different groups of people. Applying this security model removes the ability for Exchange servers and administrators to create Active Directory objects such as users, groups, and contacts. The ability to manage non-Exchange attributes on those objects is also removed.

You shouldn't apply this security model if the same person or group manages both Exchange and Active Directory. Click '?' for more information.

Conclusion & Takeaways

Vulnerabilities

- NTLM relay attack surface in Exchange Server cluster, attackers can achieve arbitrary mailbox takeover / remote code execution on your Exchange Servers with only a normal domain user / machine account.
- Privilege escalation methods from Exchange to Domain Admin that still works up to now and won't be fixed.

Mitigations

- Enable Extended Protection to mitigate NTLM relay attack surface on Exchange Server.
- Keep your Exchange Servers and Windows Servers they are running on up-to-date.
- Apply Active Directory split permissions model to mitigate privilege escalation methods.

THANK YOU!

Tianze Ding (@D1iv3)