ManageEngine
Log360

From attack to defense: Exploration of five cyber threats that can break organizations

Shehnaaz

shehnaaz.n@manageengine.com

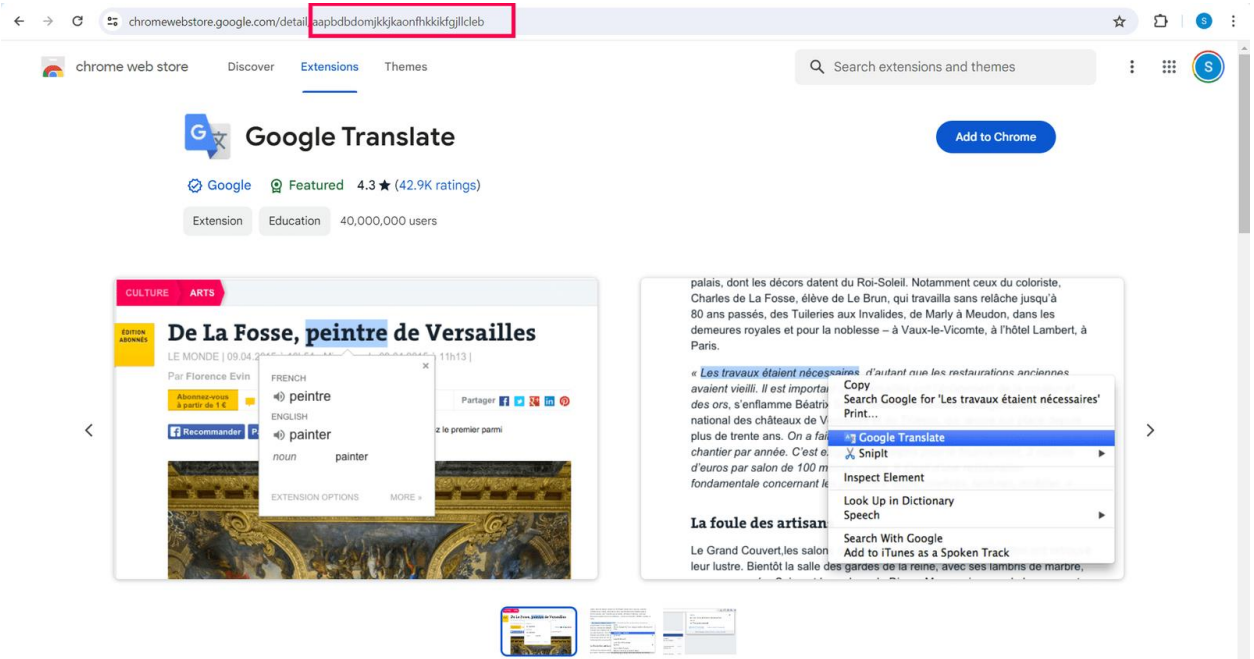
ManageEngine Log360

No fluff, just different attacks and their detection strategies

ManageEngine Log360

1. Attacking your organization using web browser based threats

Snapshot IDs, anyone?

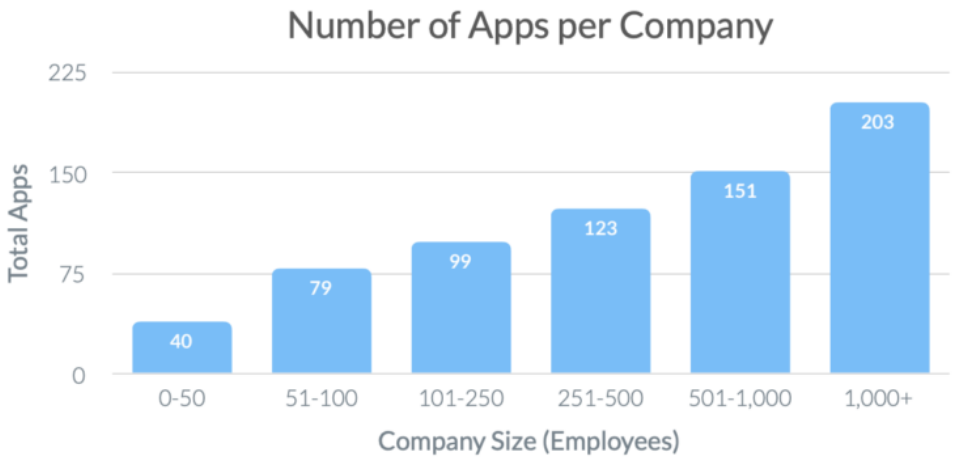


ManageEngine
Log360 Cloud

List of legitimate snapshot IDs

Extension Name	Extension Link
Adobe Acrobat	https://chrome.google.com/webstore/detail/adobe-acrobat/efaidnbmnnnibpcajpcglclefindmkaj?hl=en
Checker Plus for Google Calendar	https://chrome.google.com/webstore/detail/checker-plus-for-google-c/hkhggnncdpfibdhinjiegagmopldibha?hl=en
ChromeVox Classic Extension	https://chrome.google.com/webstore/detail/chromevox-classic-extensi/kgejglhpjiefppelpmljglcjbhoiplfn?hl=en
Cisco Webex Content Sharing	https://chrome.google.com/webstore/detail/cisco-webex-content-shari/ifbdadgbpalmagalaclfafifakmfkac
Cisco Webex Extension	https://chrome.google.com/webstore/detail/cisco-webex-extension/jlhmfgmfgeifomenelglieieghnjghma
Cisco Webex Scheduler	https://chrome.google.com/webstore/detail/cisco-webex-scheduler/ankblejcdieljecjfagjeoelaedaoaka
DeskPins (individual approval needed)	https://efotinis.neocities.org/deskpins/
DisplayNote Extension	https://chrome.google.com/webstore/detail/displaynote-extension/dbfegieohdkcnaomlgefdlpnbdhiiknh
Gmail App	https://chrome.google.com/webstore/detail/gmail/pjkljhegncpnkpknbcohdijeoejaedia?hl=en
Google Docs Offline Chrome Extension	https://chrome.google.com/webstore/detail/google-docs-offline/ghbmnnjooekpmoecnnnilnnbdlolhkhi
Google Keep Chrome Extension	https://chrome.google.com/webstore/detail/google-keep-chrome-extens/lpcaedmchfhocbbapmcbpinfpgnhiddi

Organizations on an average use 110 SaaS applications per day.



95% cloud usage happens outside your IT team's control.

ManageEngine
Log360 Cloud

Stealing data from your SaaS applications

AI-driven data leaks

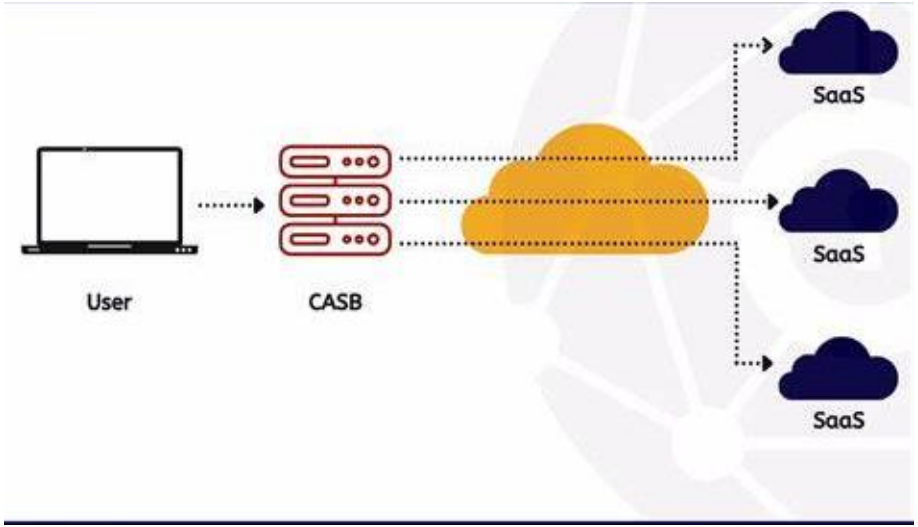
What is a CASB?

- **Gartner** defines the cloud access security broker (CASB) market as products and services that address security gaps in an organization’s use of cloud services.

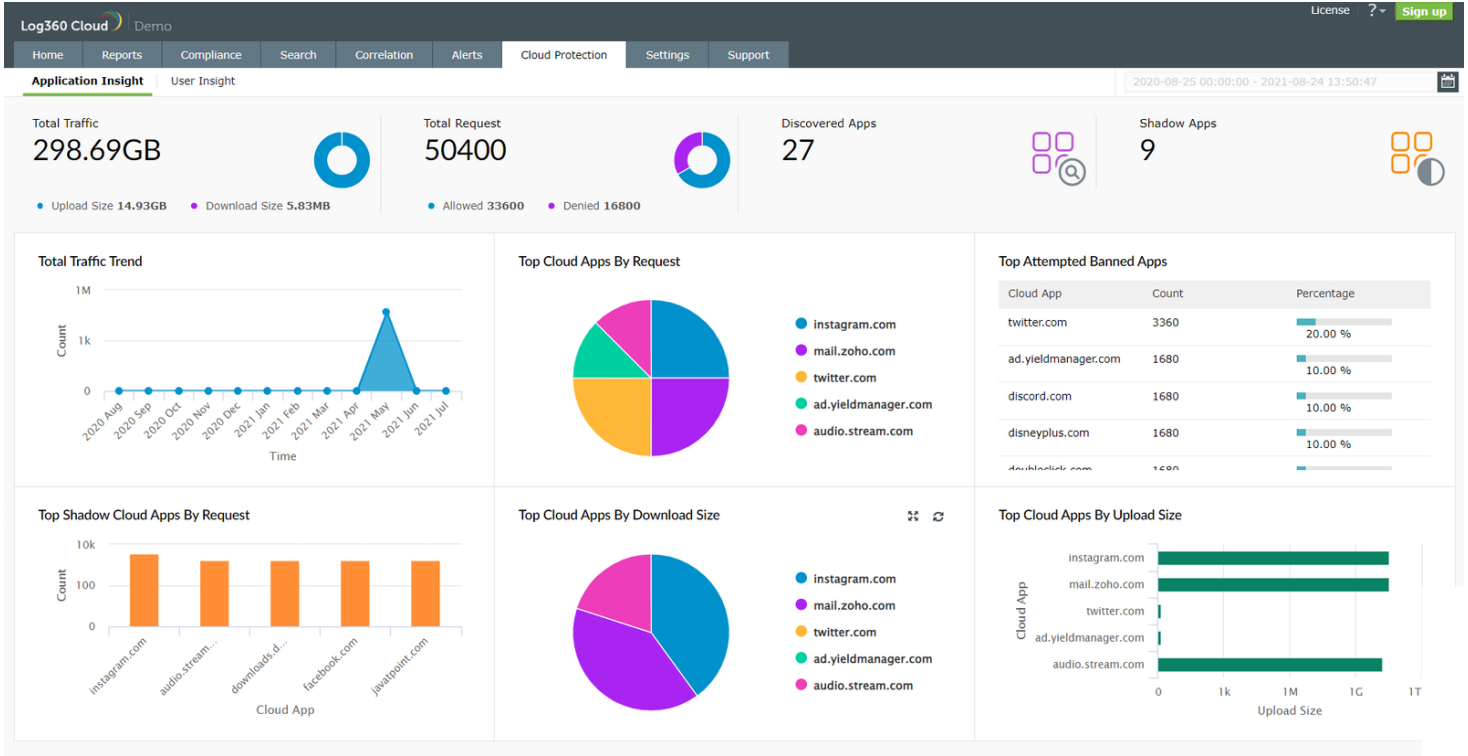
- They deliver differentiated, cloud-specific capabilities generally not available as features in other security controls such as web application firewalls (WAFs) and enterprise firewalls.

What exactly is a CASB?

A CASB is an intermediary that sits between a cloud service consumer and a cloud service provider.

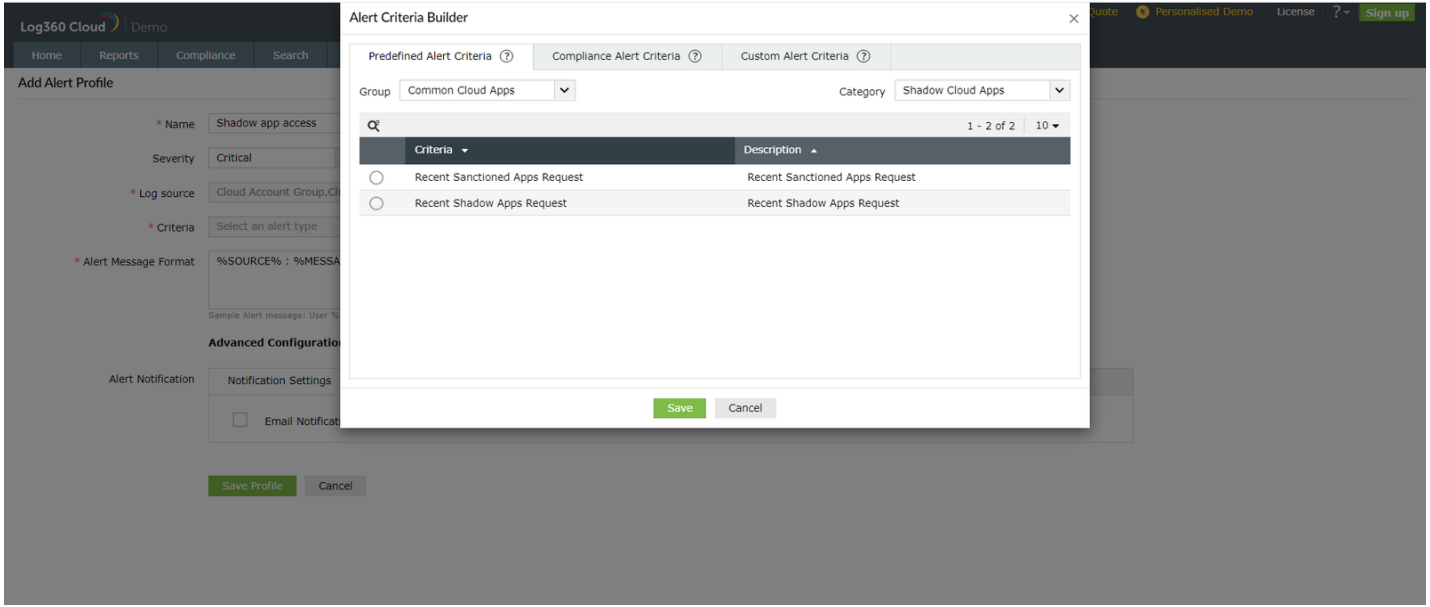


Tracking application access using CASB



ManageEngine
Log360 Cloud

Getting notified for access of shadow and banned applications

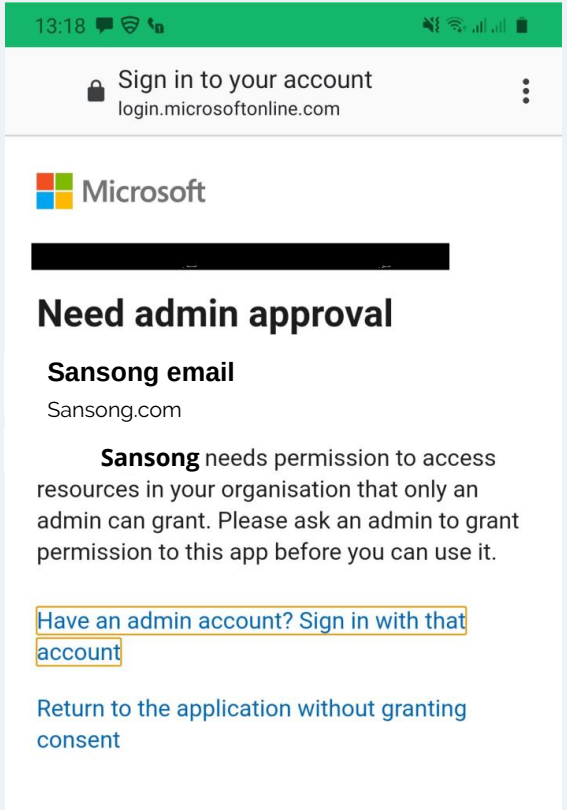


ManageEngine
Log360 Cloud

2. Azure consent grant attack

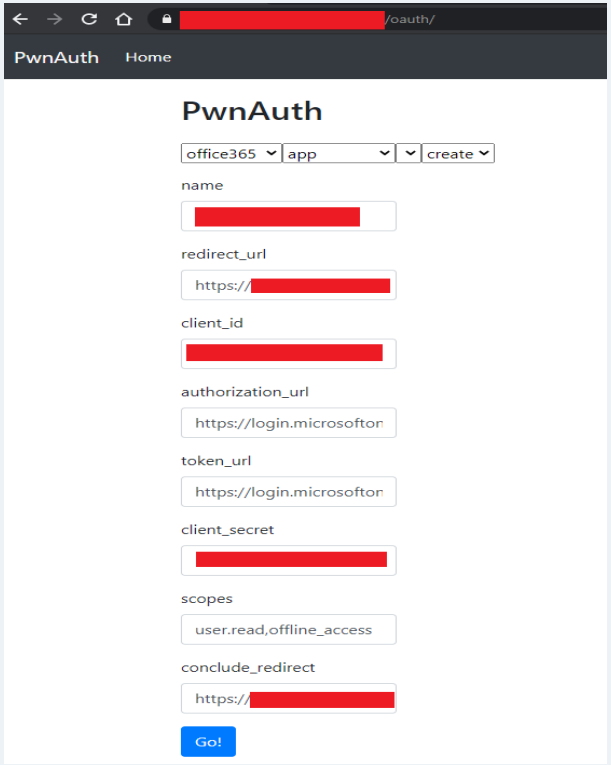
Nuanced Phishing Emails

- ✓ **Illicit consent grants:**
- ✓ Consent is the process of granting authorization to an Azure AD application to access protected resources on the users' behalf.
- ✓ Attackers can send illicit consent grant requests to users' inboxes and they get complete access if the user grants the unknown application a set of privileges.



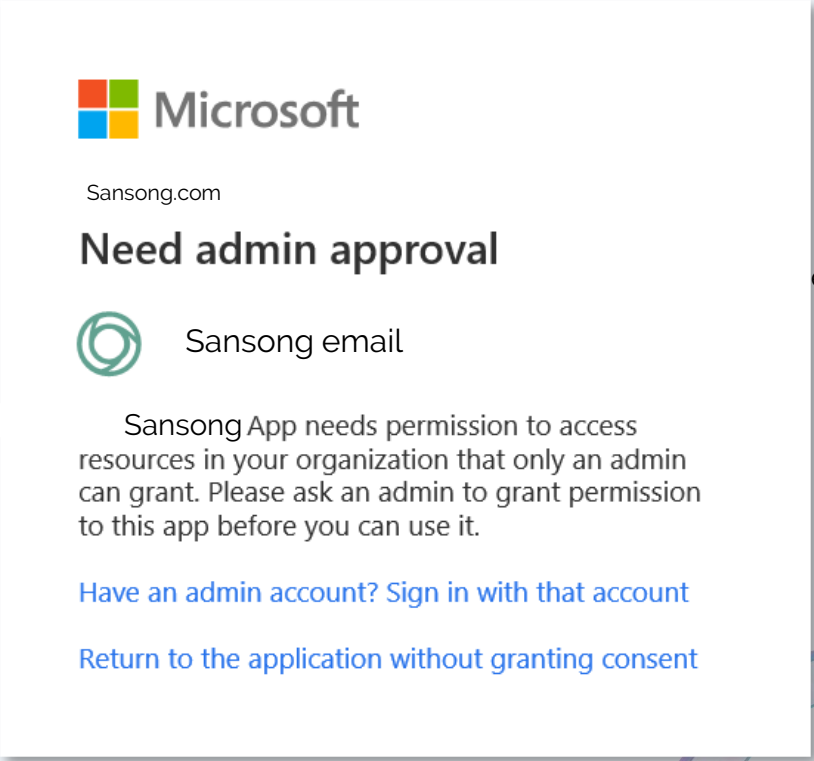
Let's go phishing!

- ✓ **PwnAuth to launch the phishing campaign:**
- ✓ Enter the token URL and authorization URL from the source tenant and in the Scopes field, enter the names of all API permissions you (the attacker) wants to request.



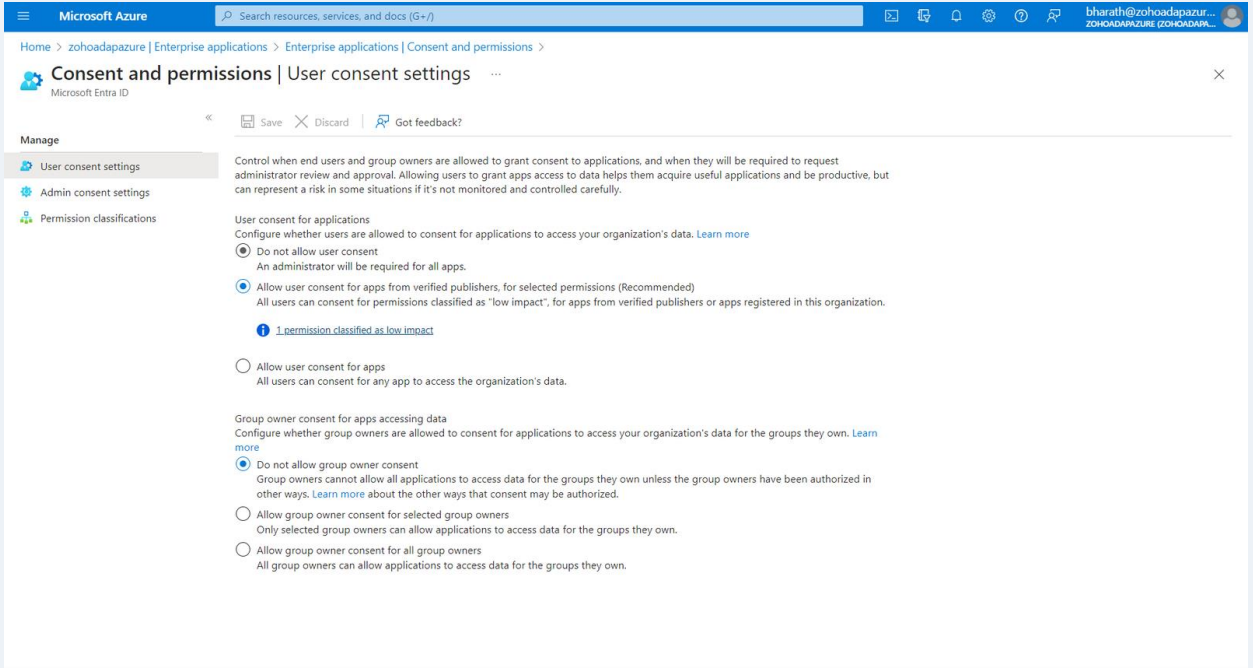
Illicit consent-grant attack

- ✓ **PwnAuth to launch the phishing campaign:**
- ✓ The victim will receive a consent-grant pop-up on which will prompt them to grant access to the application. All of this can happen without the user connecting to the corporate network.



ManageEngine Log360

Curbing illicit access grant attempts from Azure AD: Fixing mistakes at the native tool.



Enterprise applications> Consent and permissions

ManageEngine Log360

Detecting consent-grant attacks using PowerShell

```
PS C:\WINDOWS\system32> Get-AzureADOAuth2PermissionGrant | Where-Object { $_.ClientId -eq 'bb313523-9bec-[redacted]' } | Select-Object -Property *


ClientId    : bb313523-9bec-4b78-[redacted]
ConsentType : AllPrincipals
ExpiryTime  : 21-12-2022 07:54:17
ObjectId    : IzUxu-ybeEuX3h_uEJmJH2j_[redacted]
PrincipalId :
ResourceId  : 74e5ff68-284a-4d54-[redacted]
Scope       : user_impersonation
StartTime   : 01-01-0001 00:00:00

ClientId    : bb313523-9bec-4b78-[redacted]
ConsentType : AllPrincipals
ExpiryTime  : 21-12-2022 07:47:04
ObjectId    : IzUxu-ybeEuX3h_uEJmJHwD_[redacted]
PrincipalId :
ResourceId  : 66caff00-b3f3-469a-[redacted]
Scope       : offline_access User.Read openid Application.Read.All
StartTime   : 01-01-0001 00:00:00

ClientId    : bb313523-9bec-4b78-[redacted]
ConsentType : AllPrincipals
ExpiryTime  : 21-12-2022 07:54:17
ObjectId    : IzUxu-ybeEuX3h_uEJmJH574[redacted]
PrincipalId :
ResourceId  : 0e8df89e-7e1a-4328-[redacted]
Scope       : user_impersonation
StartTime   : 01-01-0001 00:00:00
```

PowerShell command to identify all apps and the consent grants a user has issued:

Get-AzureADPSPermissions.ps1 | Export-csv -Path "Permissions.csv" -NoTypeInformation

ManageEngine Log360

Revoking consent-grant using PowerShell

```
Account                        Environment TenantId                              TenantDomain           AccountType
-------                        ----------- --------                              ------------           -----------
Z[redacted].com AzureCloud  e[redacted]757fb ha[redacted].com User


PS C:\WINDOWS\system32> # Get Service Principal using objectId
$sp = Get-AzureADServicePrincipal -ObjectId "d8c94fda-ea95[redacted]821b3cc"

# Get all delegated permissions for the service principal
$spOAuth2PermissionsGrants = Get-AzureADOAuth2PermissionGrant -All $true| Where-Object { $_.clientId -eq $sp.ObjectId }

# Remove all delegated permissions
$spOAuth2PermissionsGrants | ForEach-Object {
    Remove-AzureADOAuth2PermissionGrant -ObjectId $_.ObjectId
}


# Get all application permissions for the service principal
$spApplicationPermissions = Get-AzureADServiceAppRoleAssignedTo -ObjectId $sp.ObjectId -All $true | Where-Object { $_.Pr

# Remove all delegated permissions
$spApplicationPermissions | ForEach-Object {
    Remove-AzureADServiceAppRoleAssignment -ObjectId $_.PrincipalId -AppRoleAssignmentId $_.objectId
```

Revoking consent grants for applications using PowerShell commands:

Remove-AzureADOAuth2PermissionGrant

ManageEngine Log360

3. Domain user escalating their privilege to domain admin

Home | Kali | Windows 10 and later x64 | Windows Server 2019

Applications Places Terminal Aug 14 12:12

thmvpn.sh

IDA Freeware ...

Community-LinksEdito...

Community-LinksEdito...

sanjum.ovpn

IDA Freeware ...

rev.exe

SharpUp.exe

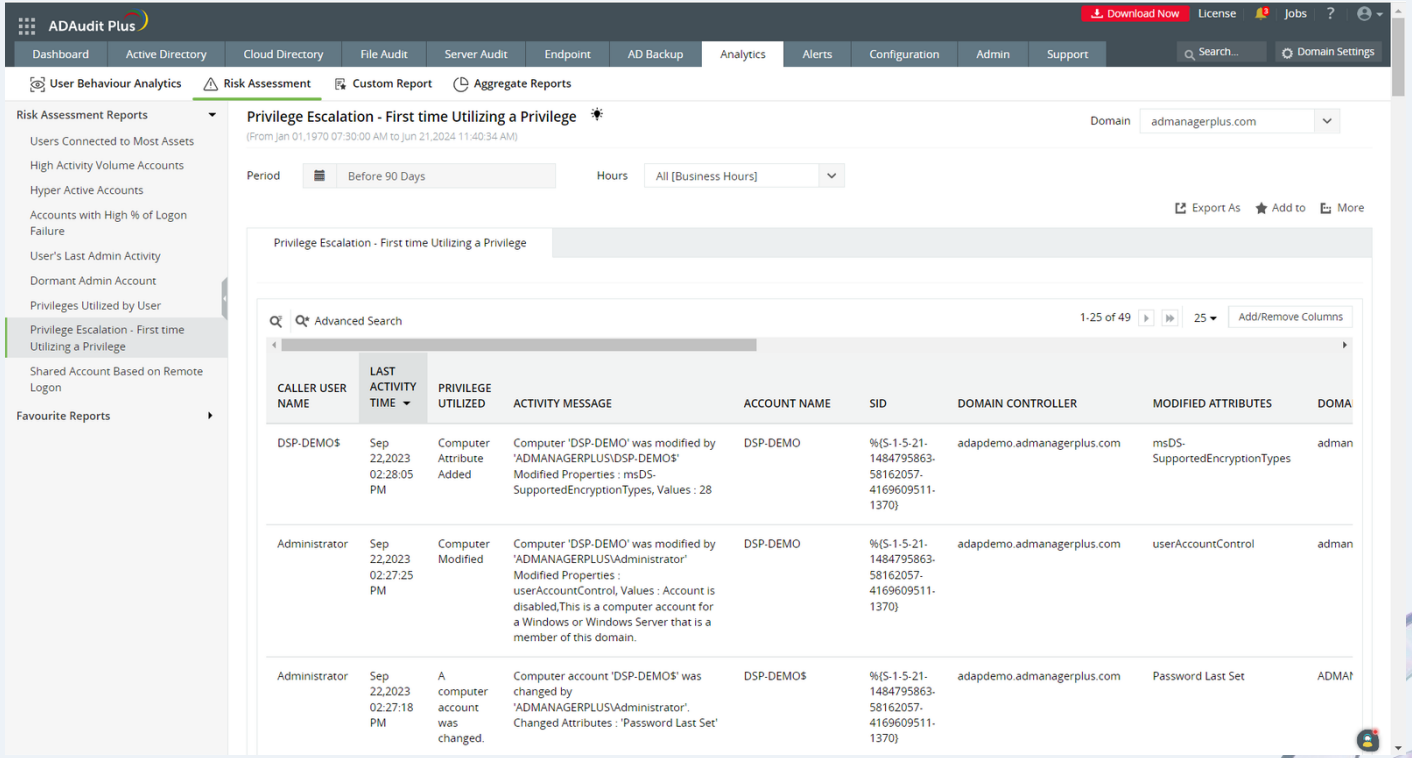
kali@kali: ~

kali@kali: ~ | kali@kali: ~/Desktop

```
msf6 exploit(windows/http/rejetto_hfs_exec) >
```

ManageEngine Log360

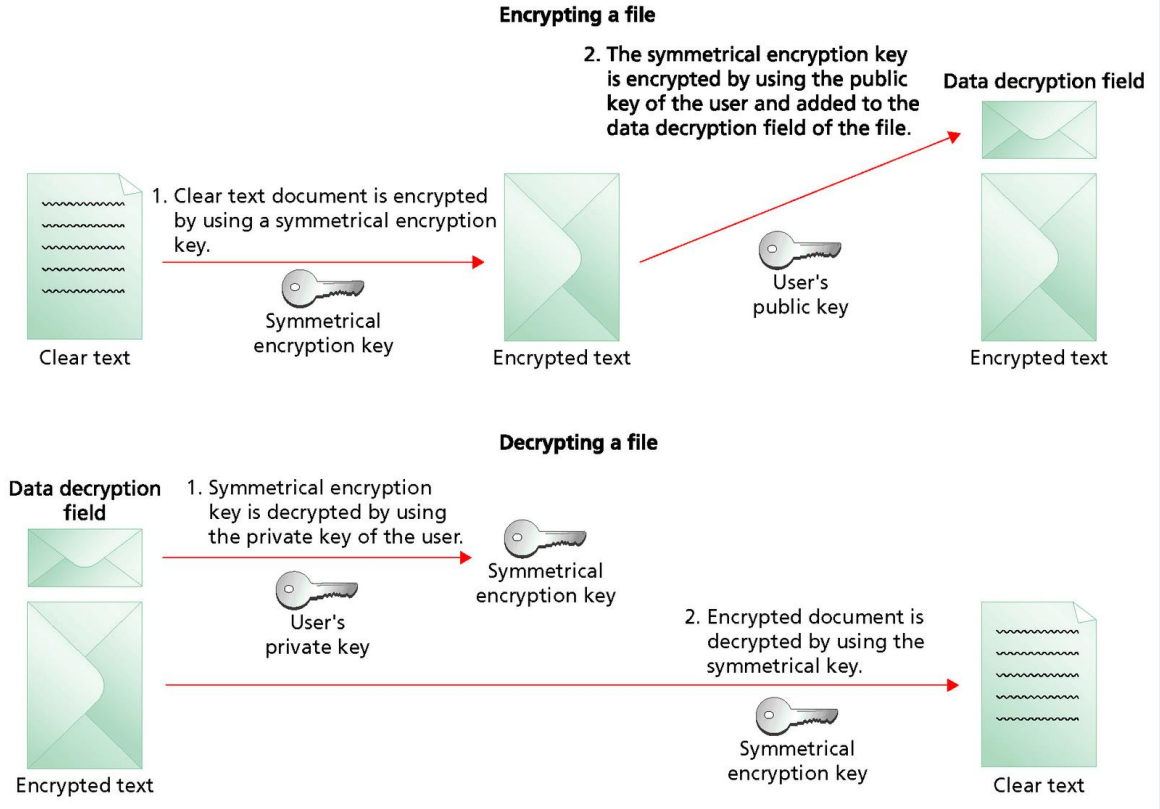
Tracking privilege escalation attempts



CALLER USER NAME	LAST ACTIVITY TIME	PRIVILEGE UTILIZED	ACTIVITY MESSAGE	ACCOUNT NAME	SID	DOMAIN CONTROLLER	MODIFIED ATTRIBUTES	DOMA
DSP-DEMO$	Sep 22,2023 02:28:05 PM	Computer Attribute Added	Computer 'DSP-DEMO' was modified by 'ADMANAGERPLUS\DSP-DEMO$' Modified Properties : msDS-SupportedEncryptionTypes, Values : 28	DSP-DEMO	%{S-1-5-21-1484795863-58162057-4169609511-1370}	adapdemo.admanagerplus.com	msDS-SupportedEncryptionTypes	adman
Administrator	Sep 22,2023 02:27:25 PM	Computer Modified	Computer 'DSP-DEMO' was modified by 'ADMANAGERPLUS\Administrator' Modified Properties : userAccountControl, Values : Account is disabled,This is a computer account for a Windows or Windows Server that is a member of this domain.	DSP-DEMO	%{S-1-5-21-1484795863-58162057-4169609511-1370}	adapdemo.admanagerplus.com	userAccountControl	adman
Administrator	Sep 22,2023 02:27:18 PM	A computer account was changed.	Computer account 'DSP-DEMO$' was changed by 'ADMANAGERPLUS\Administrator'. Changed Attributes : 'Password Last Set'	DSP-DEMO$	%{S-1-5-21-1484795863-58162057-4169609511-1370}	adapdemo.admanagerplus.com	Password Last Set	ADMAI

4. Abusing Windows Encrypting File System (EFS)

ManageEngine
Log360



ManageEngine Log360

An insider remotely accesses a server and encrypts data

Firstly, the insider must have RDP access.

For RDP access, the user must escalate their privilege.

ManageEngine Log360

The attack chain of EFS ransomware attack

1. The ransomware generates a key (using AdvApi32!CryptGenKey) to be used by EFS and records the file name used by CAPI for this key.

2. The ransomware generates a certificate for this key and adds it to the personal certificate store using Crypt32!CertAddCertificateContextToStore.

3. The ransomware sets the current EFS key to this certificate using AdvApi32!SetUserFileEncryptionKey.

4. Now the ransomware can invoke AdvApi32!EncryptFile (using the generated certificate) on every file/folder to be encrypted.

5. The ransomware saves the key file to memory and deletes it from the following two folders:

%APPDATA% \Microsoft\Crypto\RSA\sid\ (where sid is the user SID)
%ProgramData% \Microsoft\Crypto\RSA\MachineKeys

ManageEngine Log360

The code snippet to generate a AES-key using PowerShell

```
Administrator: Windows Powe

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\shehnaaz-10133> function LoadCryptographyHelper {
>>     Add-Type @"
>>         using System;
>>         using System.Runtime.InteropServices;
>>
>>         public class CryptographyHelper {
>>             [DllImport("advapi32.dll", SetLastError=true, CharSet=CharSet.Auto)]
>>             public static extern bool CryptAcquireContext(out IntPtr hProv, string pszContainer, string pszProvider, int dwProvType, int dwFlags);
>>
>>             [DllImport("advapi32.dll", SetLastError=true)]
>>             public static extern bool CryptReleaseContext(IntPtr hProv, int dwFlags);
>>
>>             [DllImport("advapi32.dll", SetLastError=true)]
>>             public static extern bool CryptGenKey(IntPtr hProv, int Algid, int dwFlags, out IntPtr phKey);
>>
>>             public static int CALG_AES_256 = 0x6602; // AES 256-bit algorithm identifier
>>
>>             public static IntPtr CreateNewKey() {
>>                 IntPtr hProv = IntPtr.Zero;
>>                 IntPtr phKey = IntPtr.Zero;
>>
>>                 // Acquire a cryptographic context
>>                 bool result = CryptAcquireContext(out hProv, "MyKeyContainer", null, 1 /* PROV_RSA_AES */, 0x08 /* CRYPT_VERIFYCONTEXT | CRYPT_NEWKEYSET
*/);
>>                 if (!result) {
>>                     int error = Marshal.GetLastWin32Error();
>>                     if (error == -2146893809) {
>>                         // If NTE_BAD_KEYSET error, try again with CRYPT_NEWKEYSET flag
>>                         result = CryptAcquireContext(out hProv, "MyKeyContainer", null, 1 /* PROV_RSA_AES */, 0x08 | 0x10 /* CRYPT_VERIFYCONTEXT | CRYPT_
NEWKEYSET */);
>>                         if (!result) {
>>                             throw new Exception("CryptAcquireContext failed. Error code: " + Marshal.GetLastWin32Error());
>>                         }
>>                     } else {
>>                         throw new Exception("CryptAcquireContext failed. Error code: " + error);
>>                     }
>>                 }
>>
>>                 // Generate a new AES 256-bit key
>>                 if (!CryptGenKey(hProv, CALG_AES_256, 0, out phKey)) {
>>                     CryptReleaseContext(hProv, 0);
>>                     throw new Exception("CryptGenKey failed. Error code: " + Marshal.GetLastWin32Error());
```

ManageEngine Log360

Administrator: Windows Powe

```
>>                      }
>>                  }
>>
>>                  // Generate a new AES 256-bit key
>>                  if (!CryptGenKey(hProv, CALG_AES_256, 0, out phKey)) {
>>                      CryptReleaseContext(hProv, 0);
>>                      throw new Exception("CryptGenKey failed. Error code: " + Marshal.GetLastWin32Error());
>>                  }
>>
>>                  // Release the cryptographic context
>>                  CryptReleaseContext(hProv, 0);
>>
>>                  return phKey;
>>              }
>>          }
>> "@
>> }
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Load the CryptographyHelper class
PS C:\Users\shehnaaz-10133> LoadCryptographyHelper
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Create a new AES 256-bit key using the helper class
PS C:\Users\shehnaaz-10133> try {
>>     $key = [CryptographyHelper]::CreateNewKey()
>>     Write-Output "AES 256-bit key created successfully."
>>     Write-Output "Key handle: $key"
>> }
>> catch {
>>     Write-Error "Error creating AES 256-bit key: $_"
>> }
AES 256-bit key created successfully.
Key handle: 2175093063984
PS C:\Users\shehnaaz-10133>
```

Rain showers
At night

Search

ENG
IN

16:51
03-07-2024

ManageEngine Log360

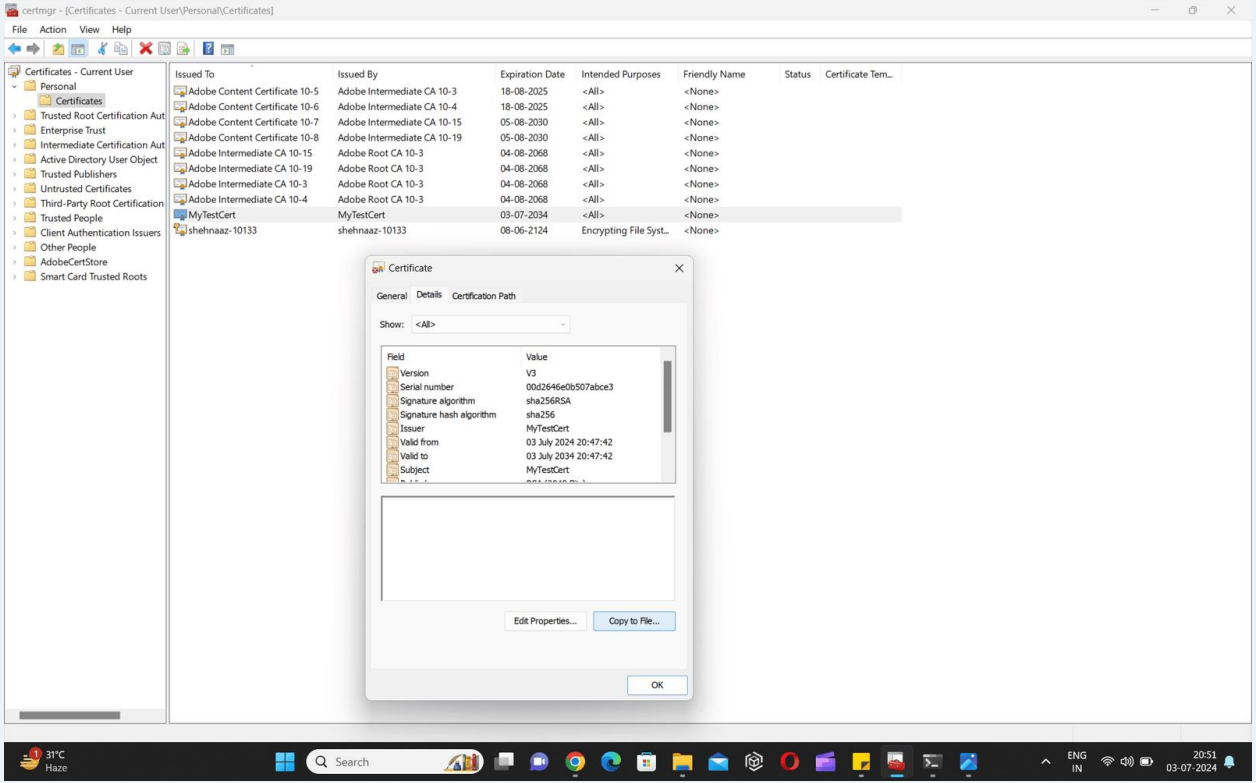
Code snippet to generate a certificate and plant it in certmgr.exe using the generated key

Administrator: Windows Powe

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\shehnaaz-10133> # Define the AES key handle
PS C:\Users\shehnaaz-10133> $keyHandle = 2440629188752
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Create a random RSA key pair for certificate signing (RSA is used for certificate operations, not AES)
PS C:\Users\shehnaaz-10133> $rsaKey = New-Object System.Security.Cryptography.RSACryptoServiceProvider 2048
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Create a Certificate Request
PS C:\Users\shehnaaz-10133> $subjectName = "CN=MyTestCert"
PS C:\Users\shehnaaz-10133> $certRequest = New-Object System.Security.Cryptography.X509Certificates.CertificateRequest($subjectName, $rsaKey, [System.Securi
ty.Cryptography.HashAlgorithmName]::SHA256, [System.Security.Cryptography.RSASignaturePadding]::Pkcs1)
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Add basic constraints extension (usually included in self-signed certificates)
PS C:\Users\shehnaaz-10133> $certRequest.CertificateExtensions.Add((New-Object System.Security.Cryptography.X509Certificates.X509BasicConstraintsExtension($
true, $false, 0, $true)))
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Self-sign the certificate
PS C:\Users\shehnaaz-10133> $cert = $certRequest.CreateSelfSigned([System.DateTime]::Now, [System.DateTime]::Now.AddYears(10))
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Convert the AES key handle to a byte array for storage in the certificate's extensions
PS C:\Users\shehnaaz-10133> $keyHandleBytes = [BitConverter]::GetBytes($keyHandle)
PS C:\Users\shehnaaz-10133> $keyHandleExtension = New-Object System.Security.Cryptography.X509Certificates.X509Extension("1.2.3.4", $keyHandleBytes, $false)
PS C:\Users\shehnaaz-10133> $cert.Extensions.Add($keyHandleExtension)
1
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> # Add the certificate to the Personal store
PS C:\Users\shehnaaz-10133> $store = New-Object System.Security.Cryptography.X509Certificates.X509Store("My", "CurrentUser")
PS C:\Users\shehnaaz-10133> $store.Open([System.Security.Cryptography.X509Certificates.OpenFlags]::ReadWrite)
PS C:\Users\shehnaaz-10133> $store.Add($cert)
PS C:\Users\shehnaaz-10133> $store.Close()
PS C:\Users\shehnaaz-10133>
PS C:\Users\shehnaaz-10133> Write-Output "Successfully created and added certificate to store."
Successfully created and added certificate to store.
PS C:\Users\shehnaaz-10133> Write-Output "Certificate Details:"
Certificate Details:
PS C:\Users\shehnaaz-10133> Write-Output $cert

Thumbprint                                Subject
----------                                -------
B8C3978AFF40C3A2C2B3E17B5D3A1C9B7829659B  CN=MyTestCert


PS C:\Users\shehnaaz-10133>
```

31°C Haze | Search | ENG IN | 20:48 03-07-2024

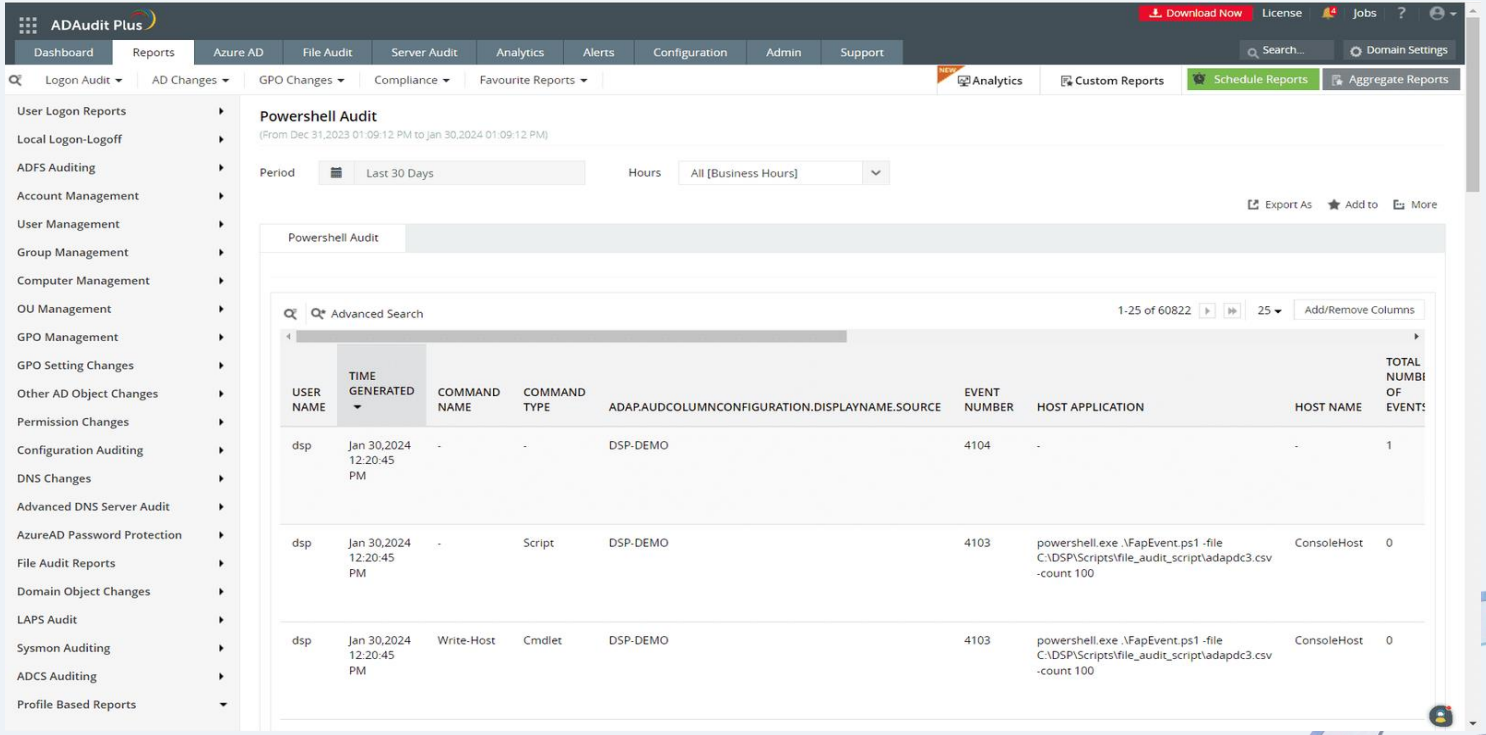
ManageEngine Log360



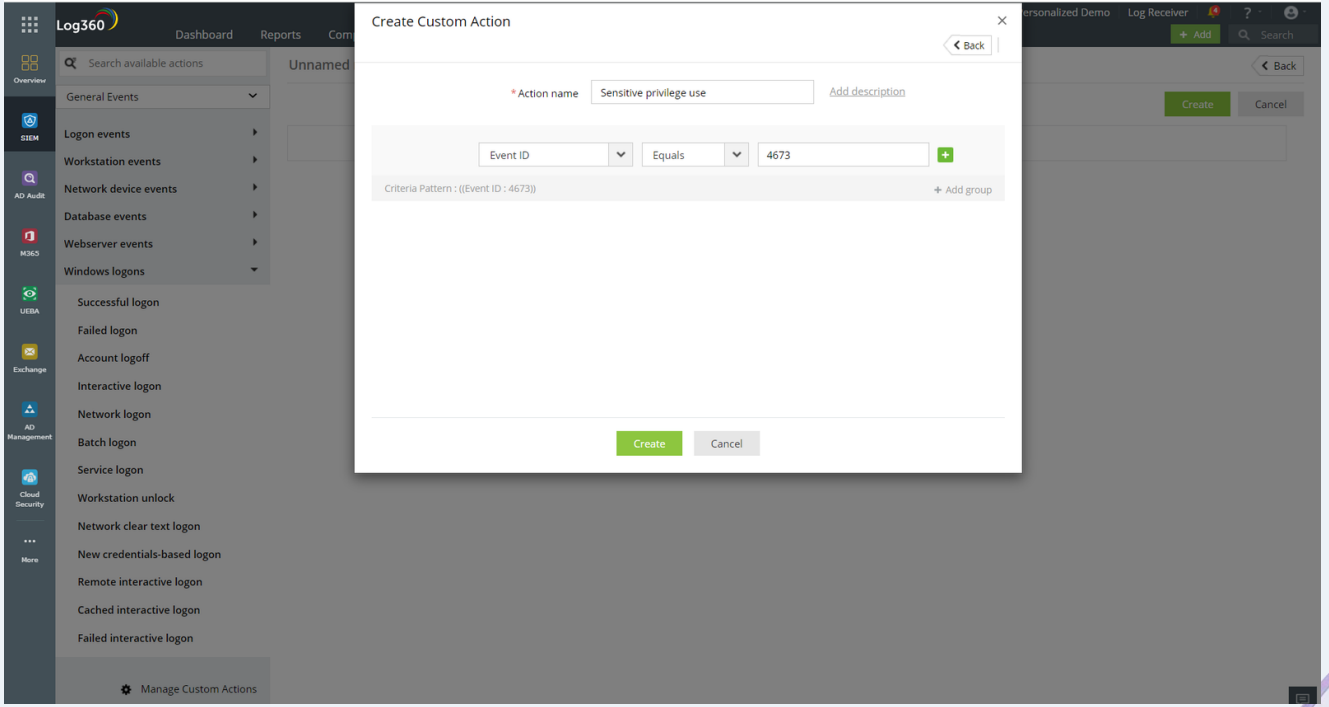
PowerShell script to delete all the data:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\shehnaaz-10133> $policy = Get-ExecutionPolicy; Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process; # Run your script here; Set-Execution
Policy -ExecutionPolicy $policy -Scope Process

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described
in the about_Execution_Policies help topic at https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): A
PS C:\Users\shehnaaz-10133>
```

Tracking PowerShell activities using Log360



Tracking encryption using the correlation module



5. One more way to take down your organization

Questions ?

ManageEngine
Log360

Bingo Code: 1503

ManageEngine
Log360

Thank you

Shehnaaz

shehnaaz.n@manageengine.com