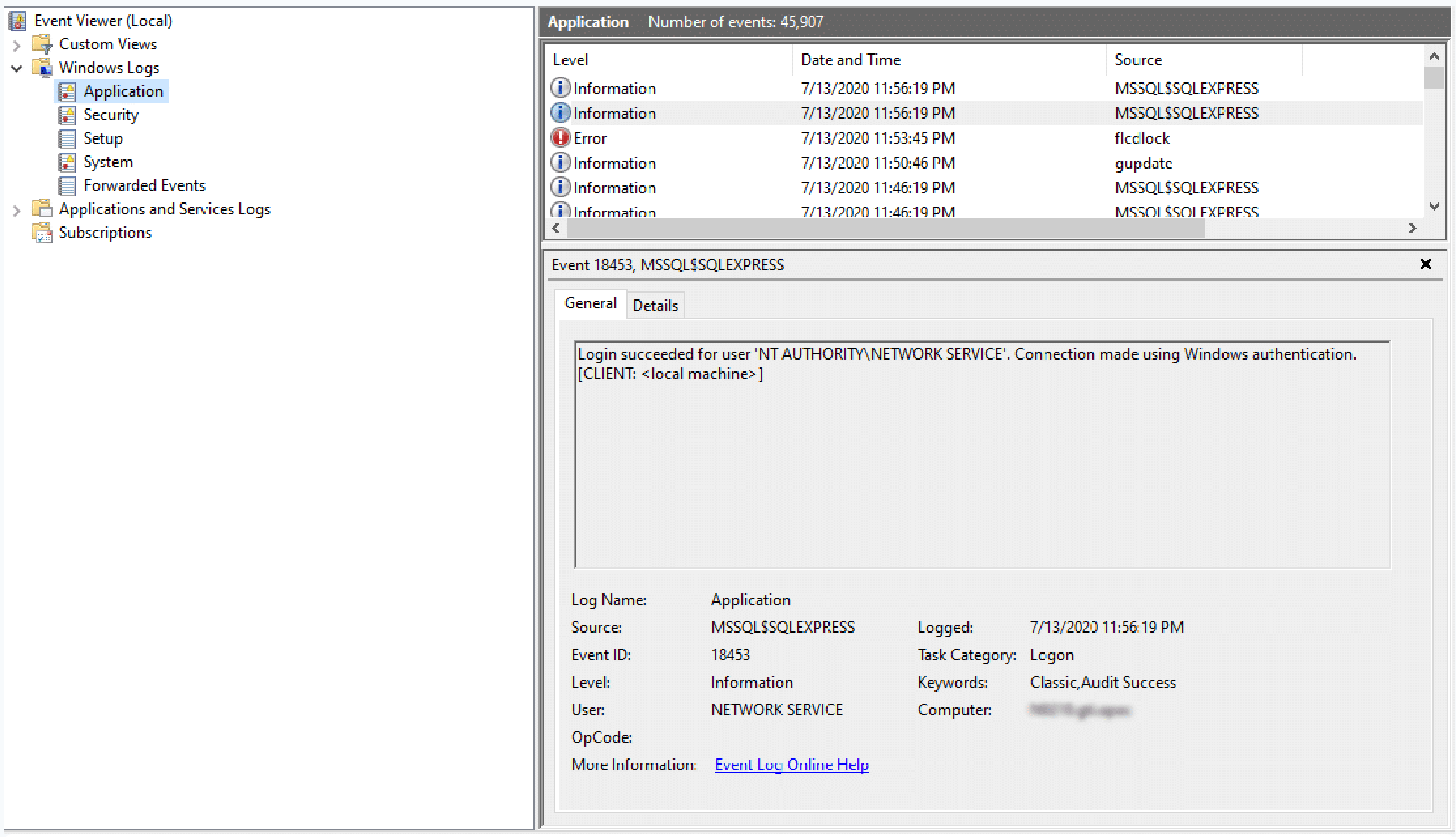


```
Level          Date and Time          Source
Information    7/13/2020 11:56:19 PM  MSSQL$SQLEXPRESS
Information    7/13/2020 11:56:19 PM  MSSQL$SQLEXPRESS
Error          7/13/2020 11:53:45 PM  flcdlock
Information    7/13/2020 11:50:46 PM  gupdate
Information    7/13/2020 11:46:19 PM  MSSQL$SQLEXPRESS
Information    7/13/2020 11:46:19 PM  MSSQL$SQLEXPRESS
```

Event 18453, MSSQL$SQLEXPRESS

```
Login succeeded for user 'NT AUTHORITY\NETWORK SERVICE'. Connection made using Windows authentication.
[CLIENT: <local machine>]
```

```
Log Name:          Application
Source:            MSSQL$SQLEXPRESS     Logged:         7/13/2020 11:56:19 PM
Event ID:          18453                Task Category:  Logon
Level:             Information          Keywords:       Classic,Audit Success
User:              NETWORK SERVICE      Computer:       [redacted]
OpCode:
More Information:  Event Log Online Help
```

Monitor MSSQL authentication with Splunk

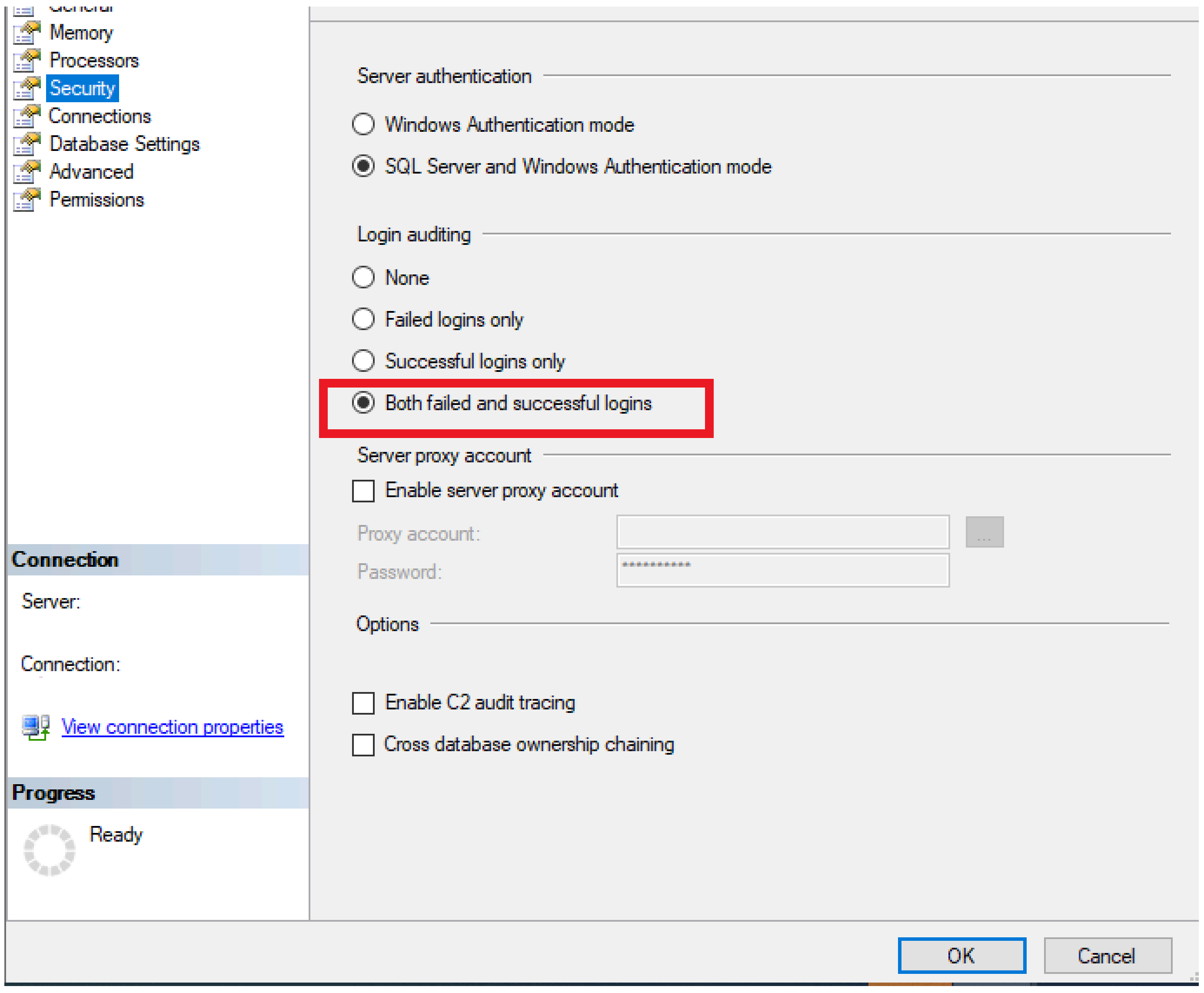
Home / Privileged Account Monitoring / Monitor MSSQL authentication with Splunk

PRIVILEGED ACCOUNT MONITORING | SPLUNK

By Kelvin Yip • July 8, 2020

Today, we are going to discuss how to monitor MSSQL authentication with Splunk.

First of all, we will need to enable MSSQL authentication log. To do so, we will log in to SQL Server Management Studio, simply right-click on your instance, go to “properties”, and then click on “security”. You will see the screen below, select “Both failed and successful logins”, and restart that particular MSSQL instance.



Server authentication
- ( ) Windows Authentication mode
- (●) SQL Server and Windows Authentication mode

Login auditing
- ( ) None
- ( ) Failed logins only
- ( ) Successful logins only
- (●) Both failed and successful logins

Server proxy account
- [ ] Enable server proxy account

Proxy account:
Password: ***********

Options
- [ ] Enable C2 audit tracing
- [ ] Cross database ownership chaining

Table of Contents

After turning on MSSQL authentication, both successful and failure authentication will be logged in **Application** Event Log no matter you are using Windows Authentication mode or Mixed mode(SQL Server and Windows Authentication mode). Below list out the Event Code/Event ID for both successful and failure authentication:

- Successful logon: 18453, 18454, 18455
- Failure logon: 18456

Analysis and Security Monitoring

Enable MSSQL authentication EventLog is only the first step, and the most important part is to monitor and reviews those audit logs. Some MSSQL instances by default use “Network Service” built-in account to start MSSQL service. It will automatically generate both successful and failure logon from this account even if you haven’t logon to MSSQL using “Network Service” account. Therefore, it is advise not to explicit grant database permission to this service account, and monitor the authentication attempts of other privilege account.

When you build the privilege account list, you must explicitly include **ALL default account and built-in account**. It is because those default account or built-in account may NOT follow your password lockout policy. The default built-in “administrator” with RID 500 is an example for this. You may find the “administrator” lockout with EventCode 4740, but this account will automatically unlocked when someone enter the correct password.

MSSQL Named pipe

In addition, MSSQL provides a method called “**named pipe**” which can connect to MSSQL via SMB (port 445). For instance, you can connect to a remote MSSQL server using the following syntax:

```
\\<Server IP>\pipe\MSSQL$SQLEXPRESS\sql\query
np:\\<Server IP>\pipe\MSSQL$SQLEXPRESS\sql\query
```

Named pipe connections support both "Windows Authentication" and "SQL Server Authentication". So, whenever there is any named pipe connection, you will see a log entry similar to the following:

```
Login succeeded for user 'sa'. Connection made using SQL Server authentication. [CLIENT: <named
pipe>]
```

In this case, we do not know whether it is a local or remote connections. Therefore, we need to correlate the Event Code 4624 log to identify whether it is coming from local or connected from remote computer. As "named pipe" relies on SMB to authenticate first, the domain users must authenticate to Window system first. So, what we should saw is Event Code 4624 (with Logon Type 3) and Event Code 18454 pair.

Monitoring MSSQL authentication with Splunk

In this section, we will discuss monitoring MSSQL authentication with Splunk. Splunk provided an app "Splunk Add-on for Microsoft SQL Server" to collect such audit logs and more. The mechanism is to configure server auditing on MSSQL DB then set up DB Connect app to retrieve audit and trace logs from Microsoft SQL DB. However, in our case, we only need to monitor the authentication log and do not want things getting too complicated. Therefore, we will stick with monitoring Event Log using Splunk.

Now, we need to define some new knowledge objects inside the Splunk_TA_windows app under the local folder. For example, for the first one which is **eventtypes.conf**, it will be placed inside $SPLUNK_HOME/etc/apps/Splunk_TA_windows/local/ of your Splunk Search Head. In addition, below configuration works for both WinEventLog or XmlWinEventLog.

eventtypes.conf

```
[wineventlog_mssql_authentication]
search = index=wineventlog source="XmlWinEventLog:Application" OR
source="WinEventLog:Application" OR sourcetype="WMI:WinEventLog:Application" (EventCode=18453 OR
EventCode=18454 OR EventCode=18455) OR (EventCode=18456)
```

props.conf

```
[source::WinEventLog:Application]
REPORT-authenication_for_mssql =
instance_for_mssql_authentication,mssql_authentication_success,mssql_authentication_failure

[source::XmlWinEventLog:Application]
REPORT-xml_authenication_for_mssql =
xml_mssql_authentication_success,xml_mssql_authentication_failure
```

tags.conf

```
[eventtype=wineventlog_mssql_authentication]
authentication = enabled
mssql = enabled
```

transforms.conf

```
[xml_mssql_authentication_success]
CLEAN_KEYS = 0
FORMAT = user::$2 src::$4 action::success app::"mssql"
REGEX = <EventID Qualifiers='\d+'>(18453|18454|18455)<\/EventID>.*?<EventData><Data>(.*?)<\/Data>
<Data> \[CLIENT: (&lt;)?(.*?)(&gt;)?]<\/Data><Binary>
SOURCE_KEY = _raw

[xml_mssql_authentication_failure]
CLEAN_KEYS = 0
FORMAT = user::$2 signature::$3 src::$5 action::failure app::"mssql"
REGEX = <EventID Qualifiers='\d+'>(18456)<\/EventID>.*?<EventData><Data>(.*?)<\/Data><Data>(.*?)
<\/Data><Data> \[CLIENT: (&lt;)?(.*?)(&gt;)?]<\/Data><Binary>
SOURCE_KEY = _raw

[mssql_authentication_success]
CLEAN_KEYS = 0
FORMAT = user_domain::$user_domain user::$user client::$client action::"success" app::"mssql"
REGEX = Login succeeded for user '((?<user_domain>[^\\]*)\\)?(?<user>[^']*)'.*\[CLIENT:\s(?
<client>.*)\]

[mssql_authentication_failure]
CLEAN_KEYS = 0
FORMAT = user_domain::$user_domain user::$user client::$client action::"failure" app::"mssql"
REGEX = Login failed for user '((?<user_domain>[^\\]*)\\)?(?<user>[^']*)'.*\[CLIENT:\s(?
<client>.*)\]

[instance_for_mssql_authentication]
CLEAN_KEYS = 0
FORMAT = instance::$1
REGEX = (?m)SourceName=(.*)[\r\n]*EventCode=(18453|18454|18455|18456)
```

We have another post discussing MySQL authentication logging, feel free to read it here.

Reference material: https://www.eventtracker.com/EventTracker/media/EventTracker/Files/support-docs/Integration-Guide-Microsoft-SQL-Server.pdf

#Authentication #Event Log #MSSQL #Privileged Account Monitoring #Splunk

← PREVIOUS
LAPS logging and Splunk

NEXT →
Detect hidden inbox forward rule in On-Premise Exchange

Similar Posts

Event ID 4625 & 4740

By Kelvin Yip • September 28, 2020

Splunk local threat intel

By Kelvin Yip • October 4, 2020

Import EventLog into Splunk

By Kelvin Yip • July 8, 2020

LAPS logging and Splunk

By Kelvin Yip • July 8, 2020

Ingest logs into Splunk using TLS

By Kelvin Yip • April 5, 2023

Windows DNS logging

By Kelvin Yip • July 24, 2020

發佈留言

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Post Comment

CYBERSECTHREAT

ISO/IEC 27001:2013 (ARES/TW/I2208053I)

繁體中文

PRODUCTS

Intezer - AI Powered Autonomous SOC Platform

AUTOCRYPT

Appdome

Kount

AKO Labs

Patented Degauss Verification Magnetic Sticker

Swimlane Turbine - AI Enabled Automation Platform

Recorded Future Intelligence Cloud

SERVICES

Free Degaussing & Data Destruction of Magnetic-Only Hard Disk Drive Program

NextGen IT Asset Disposition (ITAD)

Data Destruction Service – APJ

Data Disposal Service – Taiwan

vCISO (Virtual CISO) Services

Data Recovery

OSCP

RESOURCES

Blog

Vulnerability Research

Free Splunk Apps

COMPANY

About CyberSecThreat

News

Contact Us

About Founder

ISO27001 ISMS Policy