SIEM + Honeypot Project

Full Documentation | 10-9-2023

Luke Tapanes

Purpose

Security Information and Event Management systems are a vital part of cyber security. With this in mind, it is important to get familiar with the basic functionalities of a SIEM as well as how to handle and interpret the data. This project aims to tackle that by configuring a vulnerable machine, monitoring it, and remediating the events.

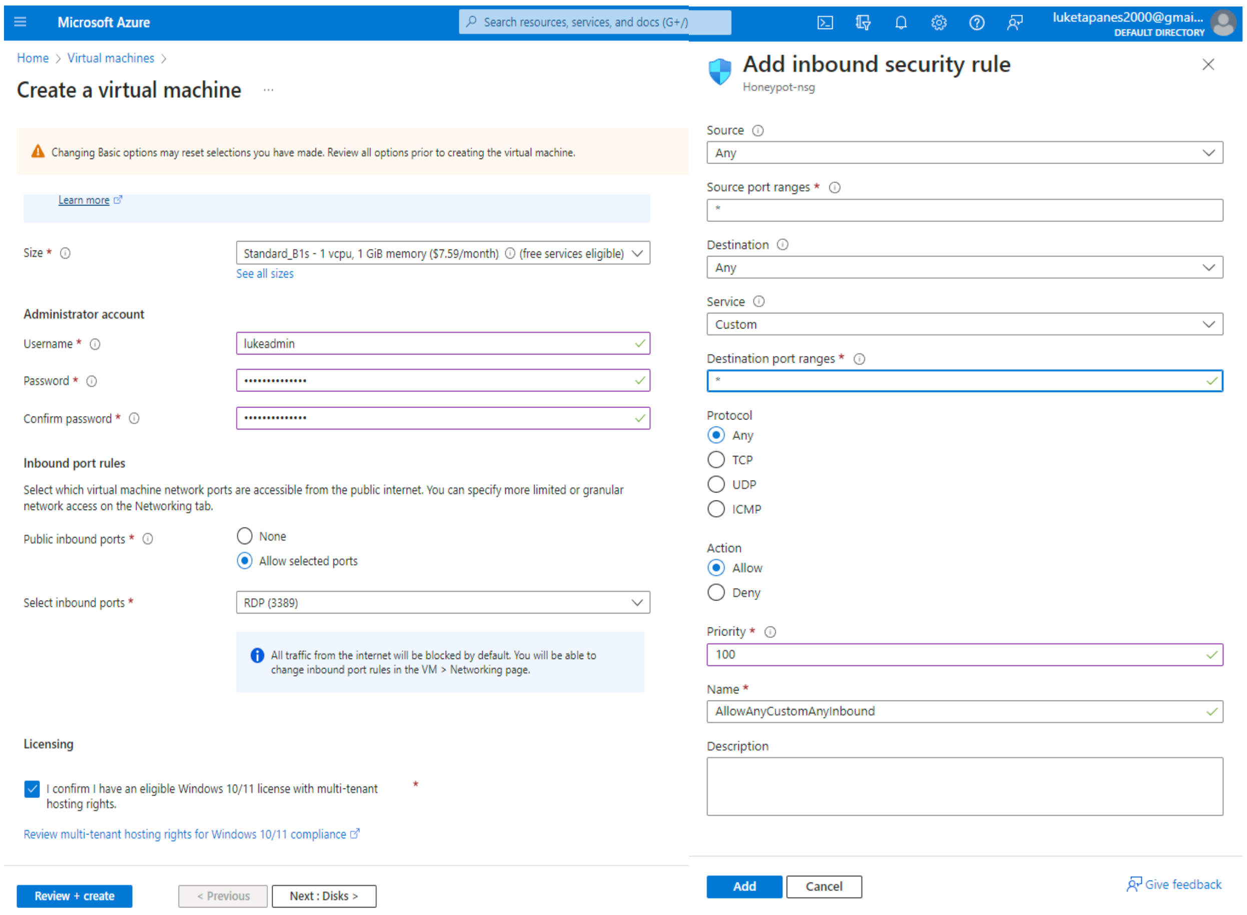
Scope

This project is the most complex that I have done so far. There are a few moving parts that will come together to make this project work. In a nutshell, I will first use Microsoft Azure as this will all be performed on the cloud. On Azure, I will then configure a virtual machine to be vulnerable to the internet and this will serve as the honeypot. All of the failed login attempt logs on the VM will be forwarded to Microsoft Sentinel which is a SIEM platform. Next, I will utilize a geolocation API to interpret where the attacks are coming from. The final part of the project will take the API data and visualize the attacks on a geographic map, in real time. Lastly, I will analyze the threat map and remediate the threats accordingly. The objectives to accomplish this are as follows:

- Configure the honeypot VM.
- Configure the log repository.
- Set up Microsoft Sentinel (SIEM).
- Configure log forwarding from the VM to Microsoft Sentinel.
- Configure geolocation API to translate Ip addresses to geographic locations.
- Configure Sentinel workbook to display the geographic data.
- Analyze the threat map.
- Remediate the threats.

<u>Project</u>

The first step in this project is to create the virtual machine that I will be monitoring. This virtual machine is intentionally going to be vulnerable so that I can monitor attacks on the machine in real time. To do this, I made sure the machine is open to the internet by disabling firewalls and creating a firewall rule to allow all inbound connections. This machine is going to be referred to as a honeypot for the remainder of the write-up. The configuration of the honeypot and the firewall rule can be seen in the screenshots below.


Microsoft Azure
Search resources, services, and docs (G+/)
luketapanes2000@gmai...
DEFAULT DIRECTORY
Home > Virtual machines >

Create a virtual machine

Changing Basic options may reset selections you have made. Review all options prior to creating the virtual machine.

Learn more

Size *	Standard_B1s - 1 vcpu, 1 GiB memory ($7.59/month) (free services eligible)

See all sizes

Administrator account

Username *	lukeadmin
Password *	••••••••••••••
Confirm password *	••••••••••••••

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports *	None [✓ Allow selected ports]
Select inbound ports *	RDP (3389)

All traffic from the internet will be blocked by default. You will be able to change inbound port rules in the VM > Networking page.

Licensing

[✓] I confirm I have an eligible Windows 10/11 license with multi-tenant hosting rights. *

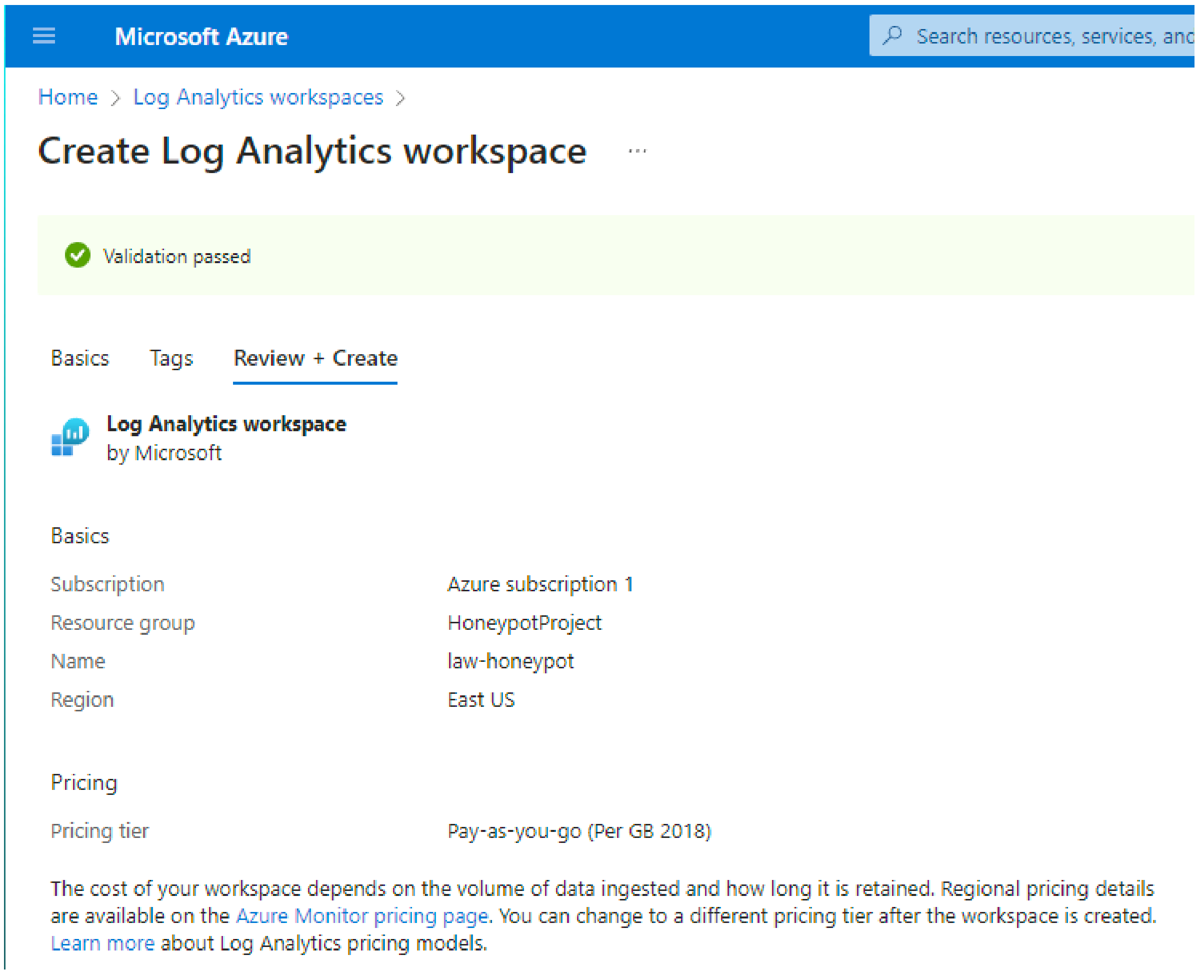
Review multi-tenant hosting rights for Windows 10/11 compliance

Add inbound security rule

Honeypot-nsg

Source	Any
Source port ranges *	*
Destination	Any
Service	Custom
Destination port ranges *	*
Protocol	[✓ Any] TCP UDP ICMP
Action	[✓ Allow] Deny
Priority *	100
Name *	AllowAnyCustomAnyInbound
Description	

The next step is to create a Log Analytics Workspace. This is essentially a repository for all of the logs to be ingested from the honeypot VM. First, create and name the repository and then I link the repository to the honeypot. The exact configuration settings can be seen in the screenshots below.


Microsoft Azure
Search resources, services, and
Home > Log Analytics workspaces >

Create Log Analytics workspace

Validation passed

Basics
Tags
Review + Create

Log Analytics workspace
by Microsoft

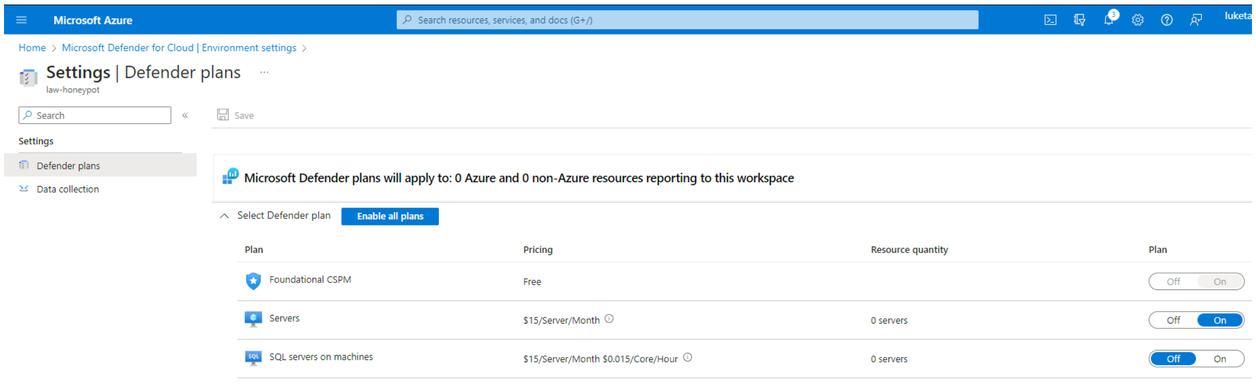
Basics

Subscription	Azure subscription 1
Resource group	HoneypotProject
Name	law-honeypot
Region	East US

Pricing

Pricing tier	Pay-as-you-go (Per GB 2018)

The cost of your workspace depends on the volume of data ingested and how long it is retained. Regional pricing details are available on the Azure Monitor pricing page. You can change to a different pricing tier after the workspace is created. Learn more about Log Analytics pricing models.

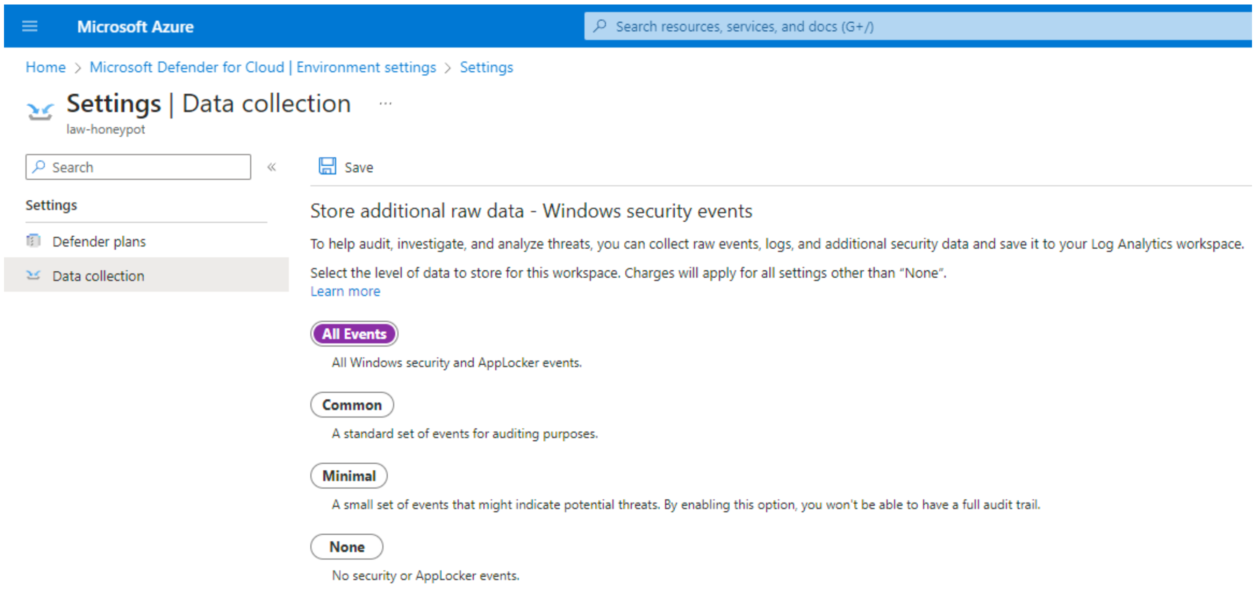

Microsoft Azure
Search resources, services, and docs (G+/)
luketa
Home > Microsoft Defender for Cloud | Environment settings >
Search
Save
Settings
Defender plans
Data collection

Settings | Defender plans
law-honeypot

Microsoft Defender plans will apply to: 0 Azure and 0 non-Azure resources reporting to this workspace

Select Defender plan **Enable all plans**

Plan	Pricing	Resource quantity	Plan
Foundational CSPM	Free		Off On
Servers	$15/Server/Month	0 servers	Off [✓ On]
SQL servers on machines	$15/Server/Month $0.015/Core/Hour	0 servers	[✓ Off] On


Microsoft Azure

Search resources, services, and docs (G+/)

Home > Microsoft Defender for Cloud | Environment settings > Settings

Settings | Data collection

law-honeypot

Search

Save

Settings

Defender plans

Data collection

Store additional raw data - Windows security events

To help audit, investigate, and analyze threats, you can collect raw events, logs, and additional security data and save it to your Log Analytics workspace.

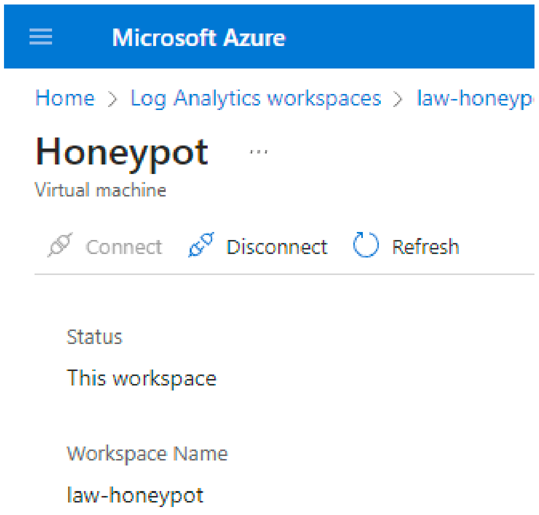
Select the level of data to store for this workspace. Charges will apply for all settings other than "None".
Learn more

[✓ All Events]
All Windows security and AppLocker events.

Common
A standard set of events for auditing purposes.

Minimal
A small set of events that might indicate potential threats. By enabling this option, you won't be able to have a full audit trail.

None
No security or AppLocker events.


Microsoft Azure

Home > Log Analytics workspaces > law-honeyp

Honeypot

Virtual machine

Connect Disconnect Refresh

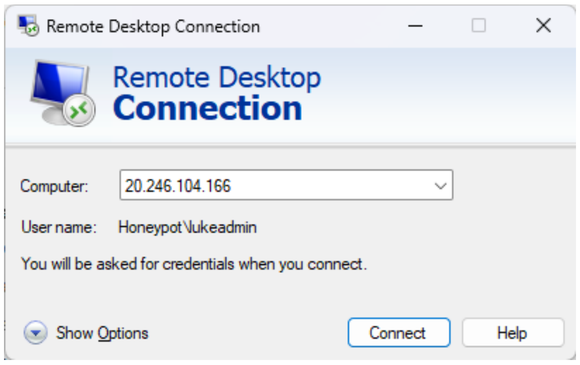
Status

This workspace

Workspace Name

law-honeypot

The next step is to RDP into the honeypot to configure a few more settings.


Remote Desktop Connection

Remote Desktop
Connection

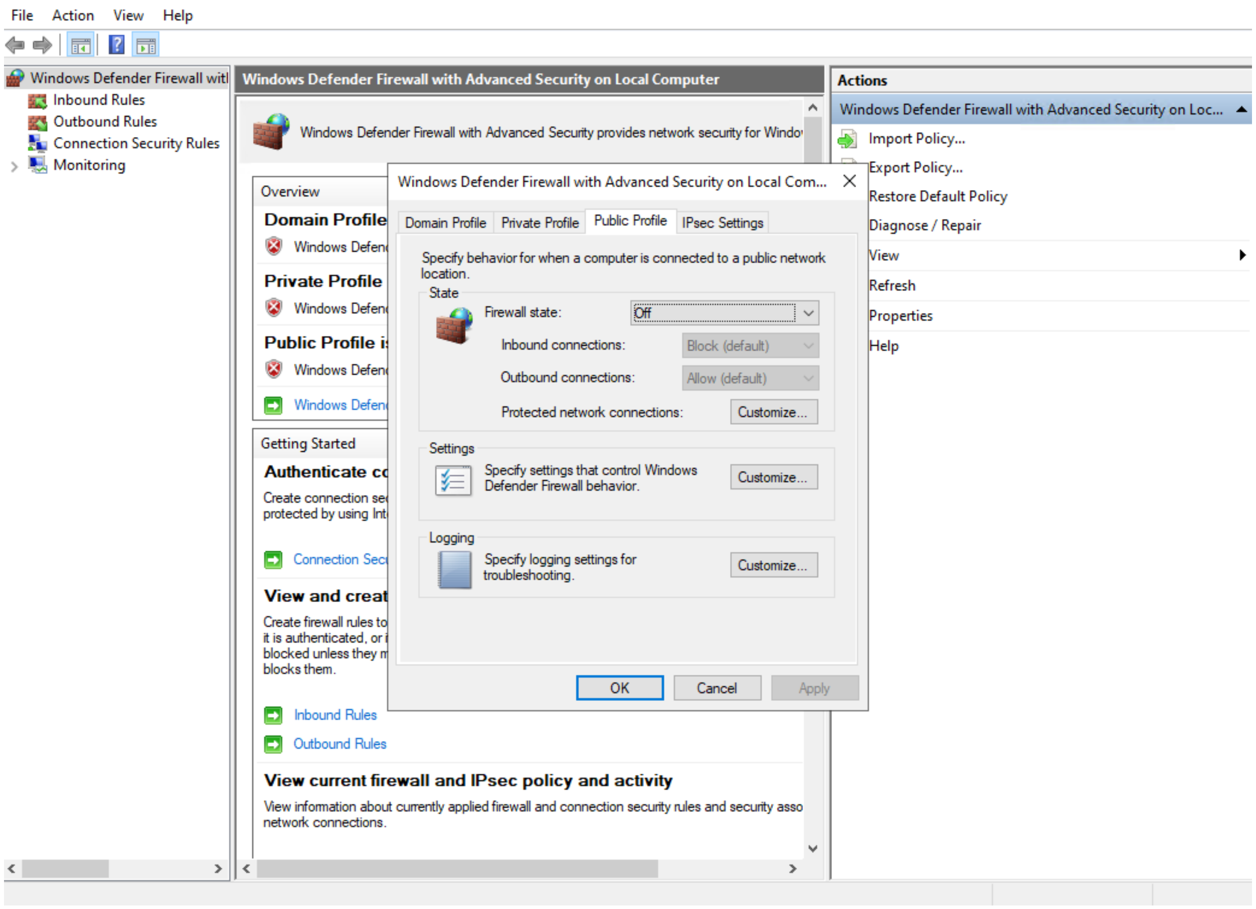
Computer: 20.246.104.166

User name: Honeypot\lukeadmin

You will be asked for credentials when you connect.

Show Options Connect Help

After I RDP into the VM, I went to Windows defender to disable a few more firewall rules in an effort to make this machine as discoverable as possible.



Windows Defender Firewall with Advanced Security provides network security for Windo

View information about currently applied firewall and connection security rules and security asso network connections.

Windows Defender Firewall with Advanced Security on Local Com...

Specify behavior for when a computer is connected to a public network location.

State

Firewall state: Off

Inbound connections: Block (default)

Outbound connections: Allow (default)

Protected network connections: Customize...

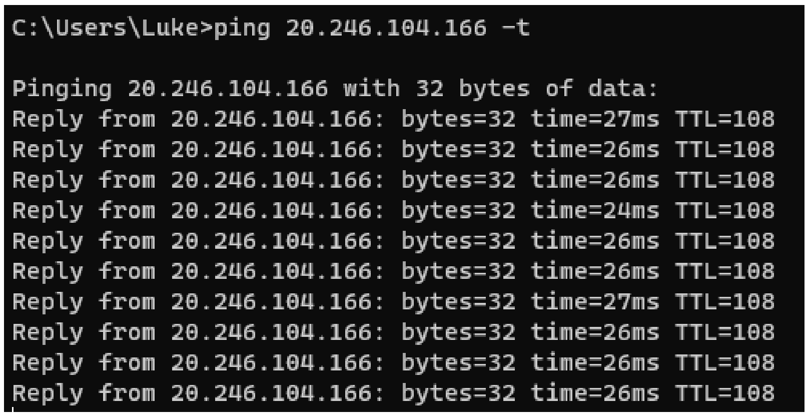
Settings

Specify settings that control Windows Defender Firewall behavior. Customize...

Logging

Specify logging settings for troubleshooting. Customize...

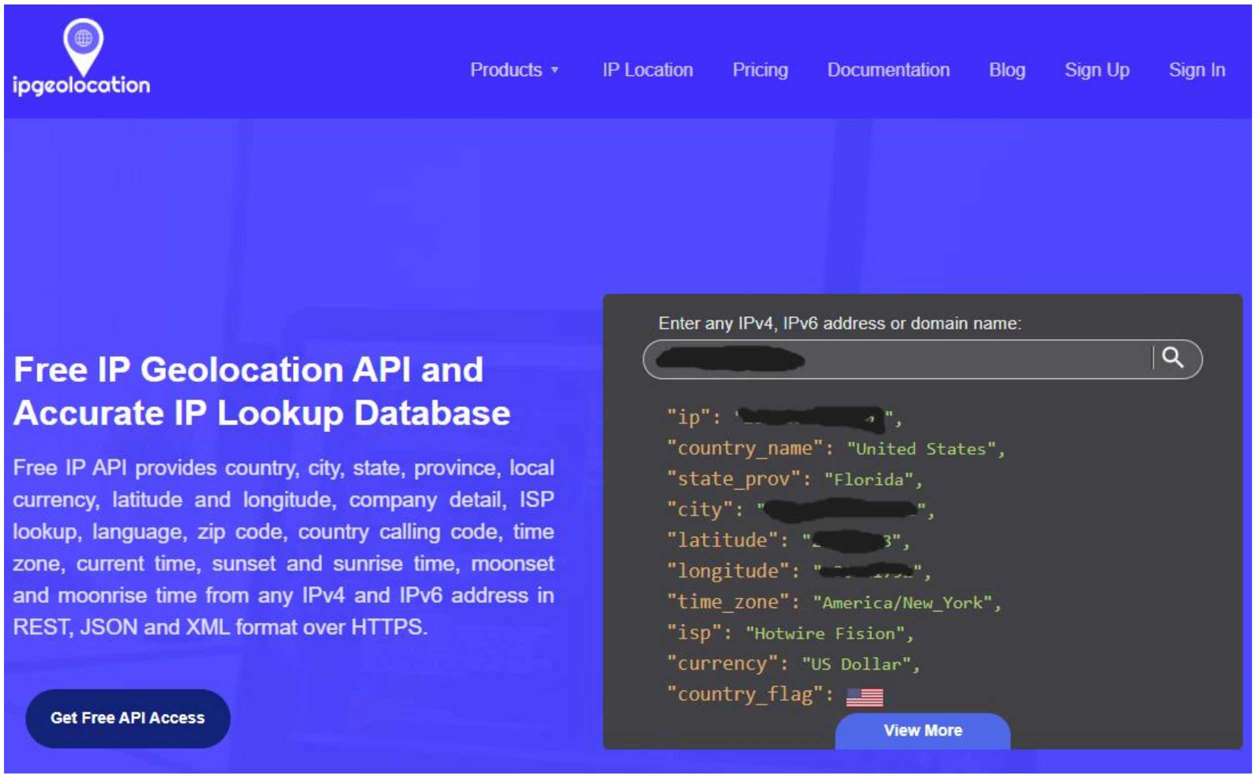
I then ping the honeypot from my personal machine to make sure the honeypot is not rejecting ICMP requests. This is important because this is primarily how attackers will discover the honeypot.



```
C:\Users\Luke>ping 20.246.104.166 -t

Pinging 20.246.104.166 with 32 bytes of data:
Reply from 20.246.104.166: bytes=32 time=27ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=24ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=27ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
Reply from 20.246.104.166: bytes=32 time=26ms TTL=108
```

At this point, the honeypot and the log forwarding is complete. The machine is now vulnerable to discovery by anyone on the internet. In normal circumstances, this would be very bad but, in this situation, this is exactly what I want.

The next phase of the project is to retrieve the geographic location of every Ip address that attempts to log into the honeypot. Because this is not a feature that Windows or Microsoft Sentinel has built in, I am going to use a third-party API to retrieve the geographic details and forward them to the log repository. This data will be ingested with all of the other logs. The API I used is IPGeoLocation.



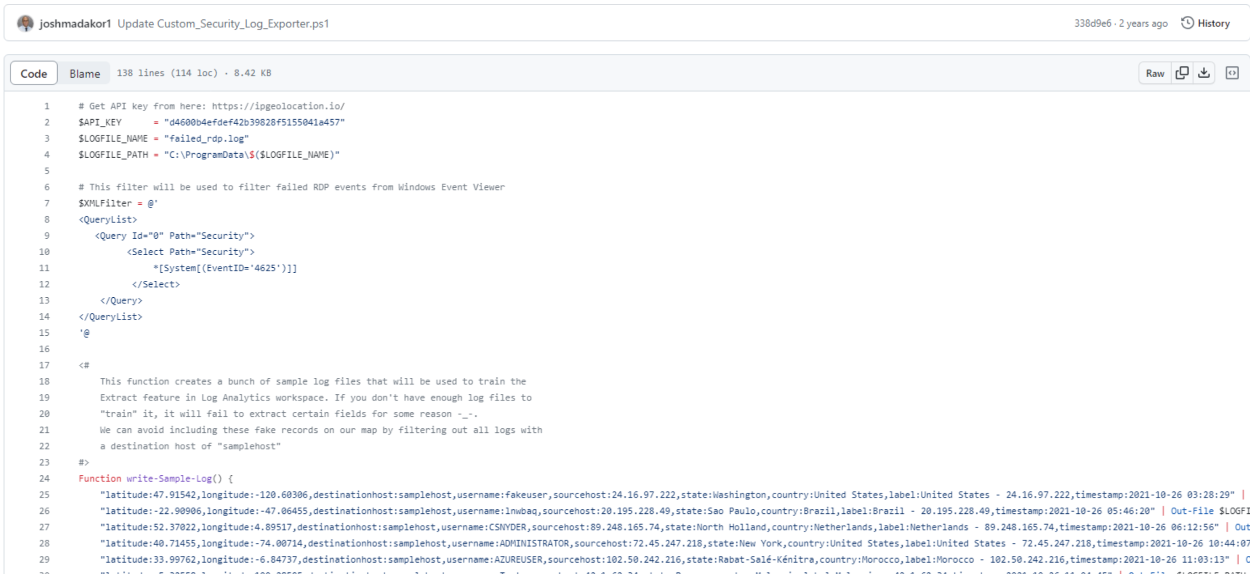
Free IP Geolocation API and Accurate IP Lookup Database

Free IP API provides country, city, state, province, local currency, latitude and longitude, company detail, ISP lookup, language, zip code, country calling code, time zone, current time, sunset and sunrise time, moonset and moonrise time from any IPv4 and IPv6 address in REST, JSON and XML format over HTTPS.

Enter any IPv4, IPv6 address or domain name: [redacted]

```
"ip": "[redacted]",
"country_name": "United States",
"state_prov": "Florida",
"city": "[redacted]",
"latitude": "[redacted]",
"longitude": "[redacted]",
"time_zone": "America/New_York",
"isp": "Hotwire Fision",
"currency": "US Dollar",
"country_flag":
```

As you can see in the screenshot, the API translates the IP address and gives a lot of details about it. Now it is time to configure a PowerShell script to extract this API data and attach the data to the log. To do this, I used a script on GitHub made by Josh Madakor. This is the guy that inspired this project, and I will link his YouTube channel and video at the end of this write-up.

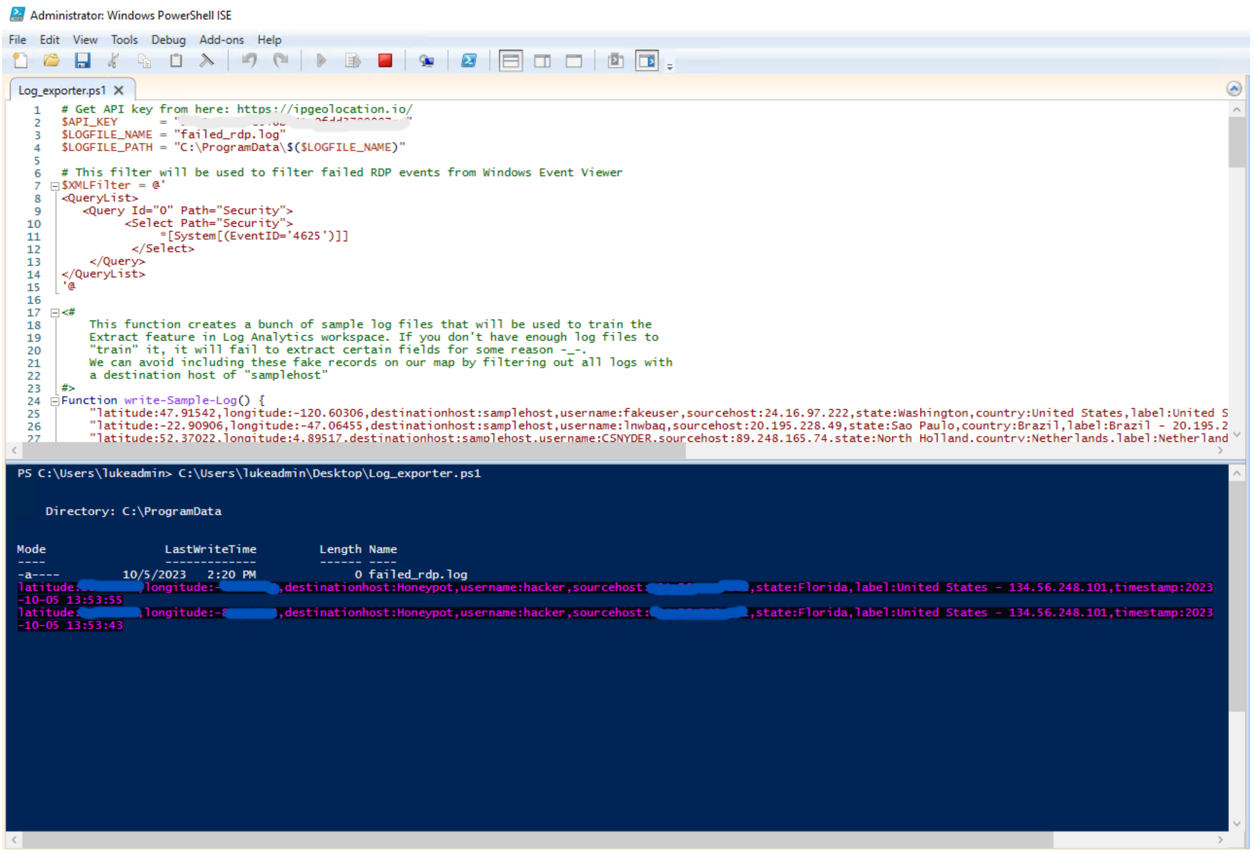


joshmadakor1 Update Custom_Security_Log_Exporter.ps1 338d9e6 · 2 years ago History

Code Blame 138 lines (114 loc) · 8.42 KB Raw

```
# Get API key from here: https://ipgeolocation.io/
$API_KEY      = "d4600b4efdef42b39828f5155041a457"
$LOGFILE_NAME = "failed_rdp.log"
$LOGFILE_PATH = "C:\ProgramData\$($LOGFILE_NAME)"

# This filter will be used to filter failed RDP events from Windows Event Viewer
$XMLFilter = @'
<QueryList>
   <Query Id="0" Path="Security">
         <Select Path="Security">
              *[System[(EventID='4625')]]
          </Select>
    </Query>
</QueryList>
'@

<#
    This function creates a bunch of sample log files that will be used to train the
    Extract feature in Log Analytics workspace. If you don't have enough log files to
    "train" it, it will fail to extract certain fields for some reason -_-.
    We can avoid including these fake records on our map by filtering out all logs with
    a destination host of "samplehost"
#>
Function write-Sample-Log() {
    "latitude:47.91542,longitude:-120.60306,destinationhost:samplehost,username:fakeuser,sourcehost:24.16.97.222,state:Washington,country:United States,label:United States - 24.16.97.222,timestamp:2021-10-26 03:28:29" |
    "latitude:-22.90906,longitude:-47.06455,destinationhost:samplehost,username:lnwbaq,sourcehost:20.195.228.49,state:Sao Paulo,country:Brazil,label:Brazil - 20.195.228.49,timestamp:2021-10-26 05:46:20" | Out-File $LOGFI
    "latitude:52.37022,longitude:4.89517,destinationhost:samplehost,username:CSNYDER,sourcehost:89.248.165.74,state:North Holland,country:Netherlands,label:Netherlands - 89.248.165.74,timestamp:2021-10-26 06:12:56" | Out
    "latitude:40.71455,longitude:-74.00714,destinationhost:samplehost,username:ADMINISTRATOR,sourcehost:72.45.247.218,state:New York,country:United States,label:United States - 72.45.247.218,timestamp:2021-10-26 10:44:07
    "latitude:33.99762,longitude:-6.84737,destinationhost:samplehost,username:AZUREUSER,sourcehost:102.50.242.216,state:Rabat-Salé-Kénitra,country:Morocco,label:Morocco - 102.50.242.216,timestamp:2021-10-26 11:03:13" | C
```

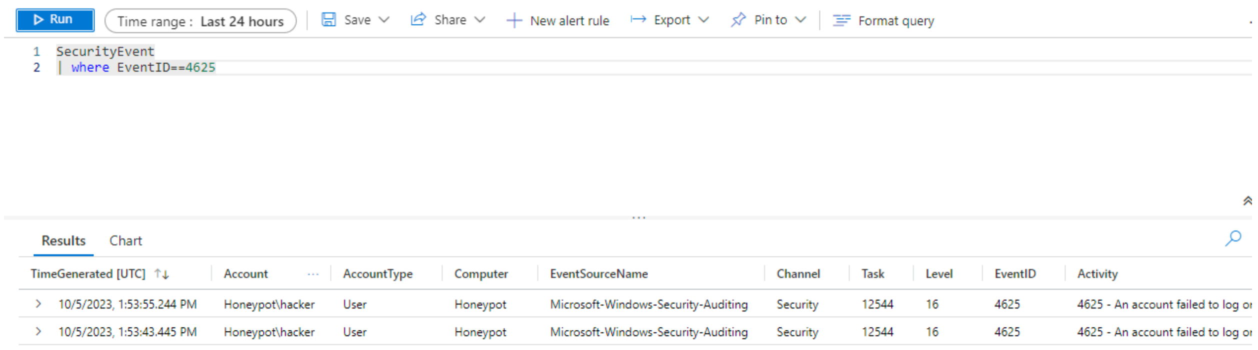
Next, I ran the script and tested it out by intentionally failing to log into the machine. Sure enough, it worked and you can see the failed login attempts in the purple.


Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
Log_exporter.ps1

```
# Get API key from here: https://ipgeolocation.io/
$API_KEY      = [redacted]
$LOGFILE_NAME = "failed_rdp.log"
$LOGFILE_PATH = "C:\ProgramData\$($LOGFILE_NAME)"

# This filter will be used to filter failed RDP events from Windows Event Viewer
$XMLFilter = @'
<QueryList>
   <Query Id="0" Path="Security">
         <Select Path="Security">
              *[System[(EventID='4625')]]
          </Select>
    </Query>
</QueryList>
'@

<#
    This function creates a bunch of sample log files that will be used to train the
    Extract feature in Log Analytics workspace. If you don't have enough log files to
    "train" it, it will fail to extract certain fields for some reason -_-.
    We can avoid including these fake records on our map by filtering out all logs with
    a destination host of "samplehost"
#>
Function write-Sample-Log() {
    "latitude:47.91542,longitude:-120.60306,destinationhost:samplehost,username:fakeuser,sourcehost:24.16.97.222,state:Washington,country:United States,label:United S
    "latitude:-22.90906,longitude:-47.06455,destinationhost:samplehost,username:lnwbaq,sourcehost:20.195.228.49,state:Sao Paulo,country:Brazil,label:Brazil - 20.195.2
    "latitude:52.37022,longitude:4.89517,destinationhost:samplehost,username:CSNYDER,sourcehost:89.248.165.74,state:North Holland,country:Netherlands,label:Netherland
```

```
PS C:\Users\lukeadmin> C:\Users\lukeadmin\Desktop\Log_exporter.ps1


    Directory: C:\ProgramData


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/5/2023   2:20 PM              0 failed_rdp.log
latitude:[redacted],longitude:-[redacted],destinationhost:Honeypot,username:hacker,sourcehost:[redacted],state:Florida,label:United States - 134.56.248.101,timestamp:2023
-10-05 13:53:55
latitude:[redacted],longitude:-[redacted],destinationhost:Honeypot,username:hacker,sourcehost:[redacted],state:Florida,label:United States - 134.56.248.101,timestamp:2023
-10-05 13:53:43
```

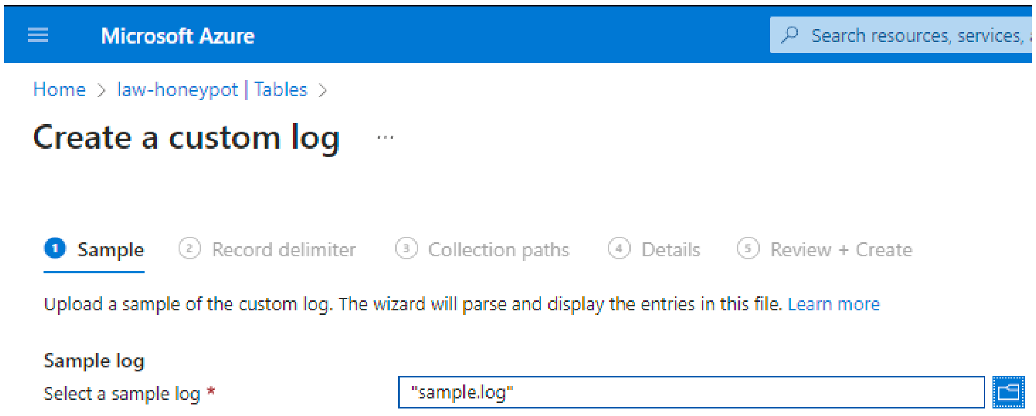
The next step is to ensure the logs are being fowared properly to the repository. To verify this, I ran a query on the logs to make sure that everything is properly linked. This can be seen in the screenshot below.



```
SecurityEvent
| where EventID==4625
```

TimeGenerated [UTC]	Account	AccountType	Computer	EventSourceName	Channel	Task	Level	EventID	Activity
10/5/2023, 1:53:55.244 PM	Honeypot\hacker	User	Honeypot	Microsoft-Windows-Security-Auditing	Security	12544	16	4625	4625 - An account failed to log on
10/5/2023, 1:53:43.445 PM	Honeypot\hacker	User	Honeypot	Microsoft-Windows-Security-Auditing	Security	12544	16	4625	4625 - An account failed to log on

The event ID seen in the screenshot is the ID of a failed login attempt. Now that I know it works, it is time to create a log table. This is to get the specific information that I am looking for in these logs which are the failed login attempts. I do this by specifying the logfile on the honeypot to retrieve.



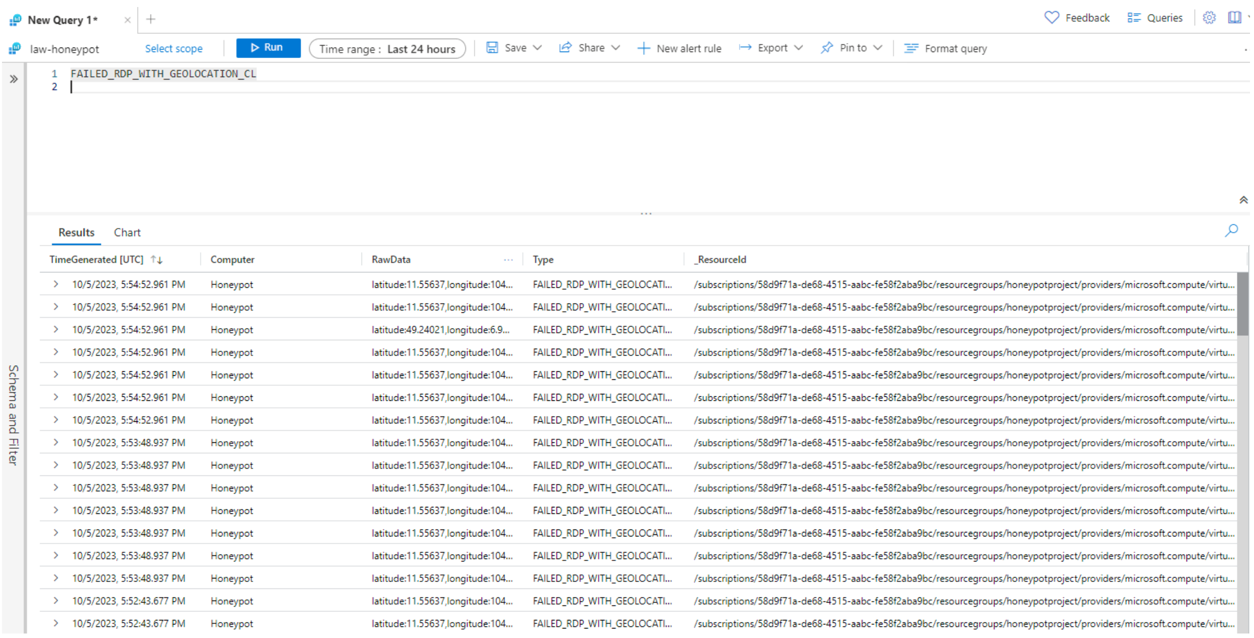
Create a custom log

Upload a sample of the custom log. The wizard will parse and display the entries in this file. Learn more

Sample log

Select a sample log * "sample.log"

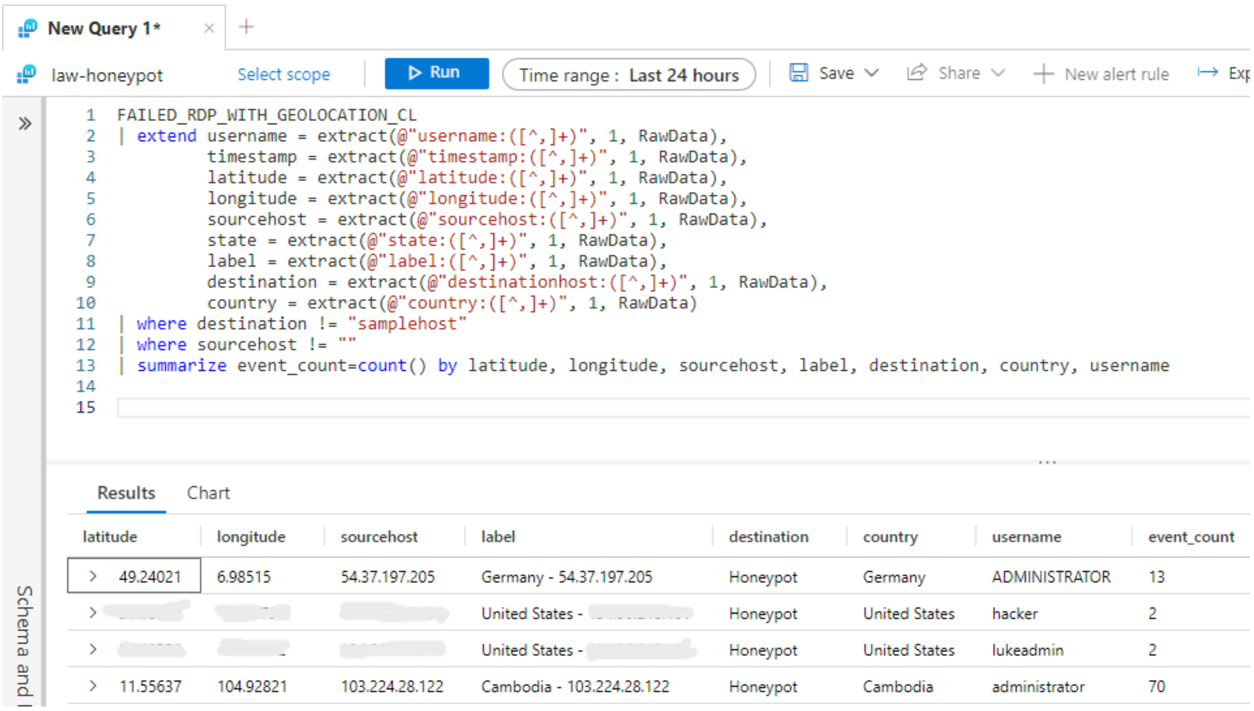
After the custom log was configured, I ran a query on that custom log. I named the custom log FAILED_RDP_WITH_GEOLOCATION. This can be seen in the screenshot below.



```
FAILED_RDP_WITH_GEOLOCATION_CL
```

TimeGenerated [UTC]	Computer	RawData	Type	_ResourceId
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:49.24021,longitude:6.9...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:54:52.961 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:53:48.937 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:52:43.677 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...
10/5/2023, 5:52:43.677 PM	Honeypot	latitude:11.55637,longitude:104...	FAILED_RDP_WITH_GEOLOCATI...	/subscriptions/58d9f71a-de68-4515-aabc-fe58f2aba9bc/resourcegroups/honeypotproject/providers/microsoft.compute/virtu...

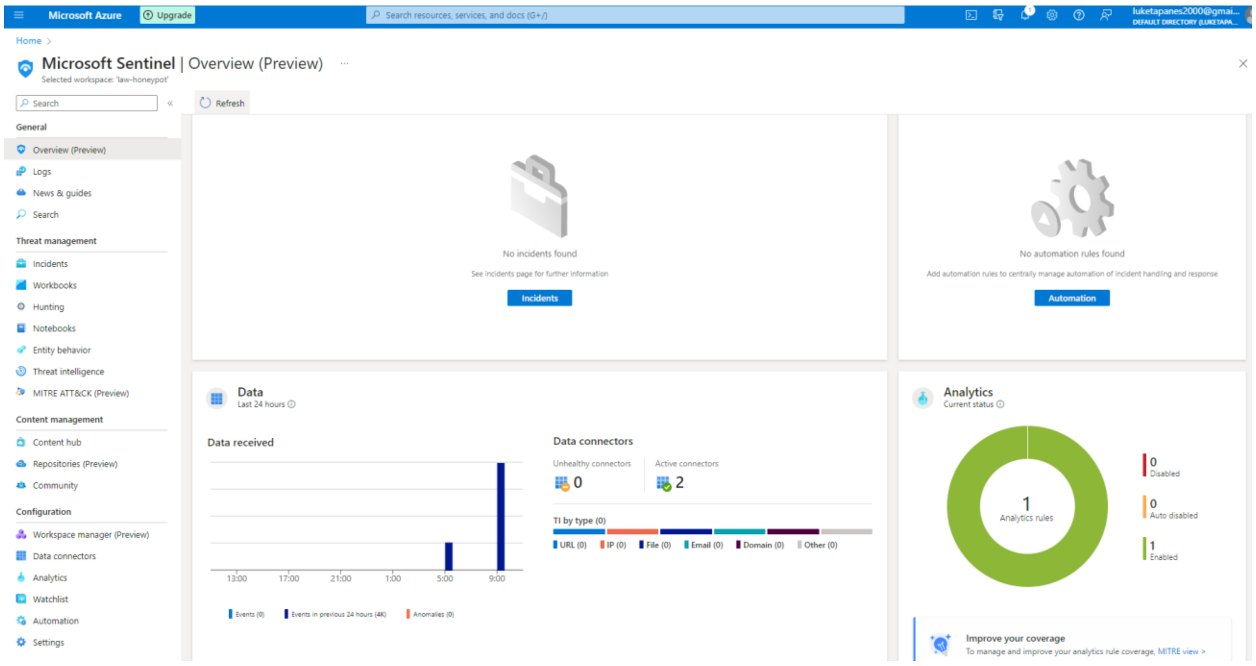
Now, it is time to extract the exact date I am looking for. To do this I need to be specific in the query and this can be seen in the screenshot below.



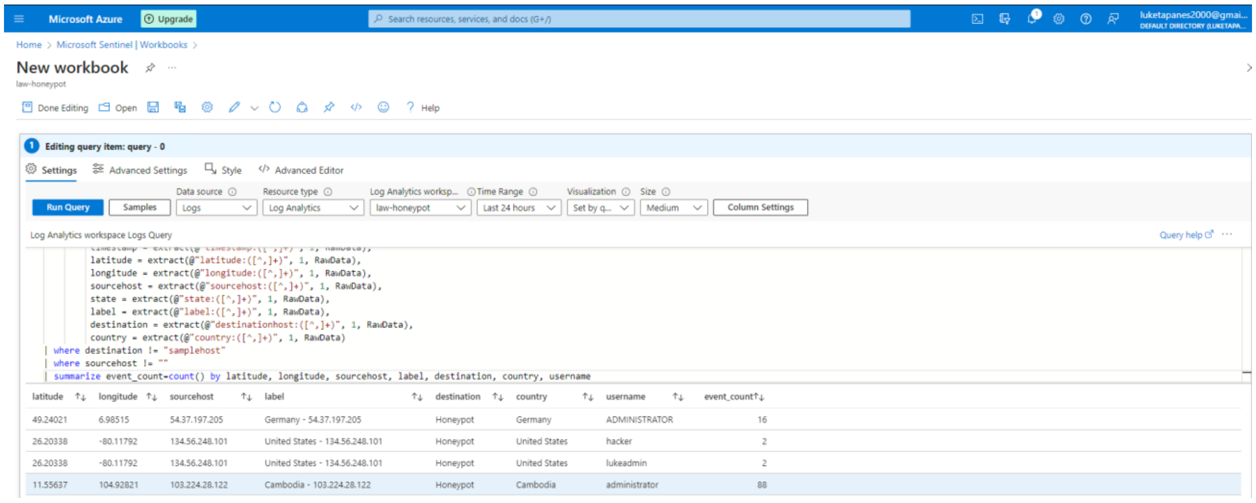
```
FAILED_RDP_WITH_GEOLOCATION_CL
| extend username = extract(@"username:([^,]+)", 1, RawData),
         timestamp = extract(@"timestamp:([^,]+)", 1, RawData),
         latitude = extract(@"latitude:([^,]+)", 1, RawData),
         longitude = extract(@"longitude:([^,]+)", 1, RawData),
         sourcehost = extract(@"sourcehost:([^,]+)", 1, RawData),
         state = extract(@"state:([^,]+)", 1, RawData),
         label = extract(@"label:([^,]+)", 1, RawData),
         destination = extract(@"destinationhost:([^,]+)", 1, RawData),
         country = extract(@"country:([^,]+)", 1, RawData)
| where destination != "samplehost"
| where sourcehost != ""
| summarize event_count=count() by latitude, longitude, sourcehost, label, destination, country, username
```

latitude	longitude	sourcehost	label	destination	country	username	event_count
49.24021	6.98515	54.37.197.205	Germany - 54.37.197.205	Honeypot	Germany	ADMINISTRATOR	13
[redacted]	[redacted]	[redacted]	United States - [redacted]	Honeypot	United States	hacker	2
[redacted]	[redacted]	[redacted]	United States - [redacted]	Honeypot	United States	lukeadmin	2
11.55637	104.92821	103.224.28.122	Cambodia - 103.224.28.122	Honeypot	Cambodia	administrator	70

The next phase is to configure the SIEM itself. This is very straightforward and all I did was link the log repository to Microsoft Sentinel.



Now it is time to take the log data and visualize it on a geographic map. To do this, I created a workbook and copy and pasted the query I used before.

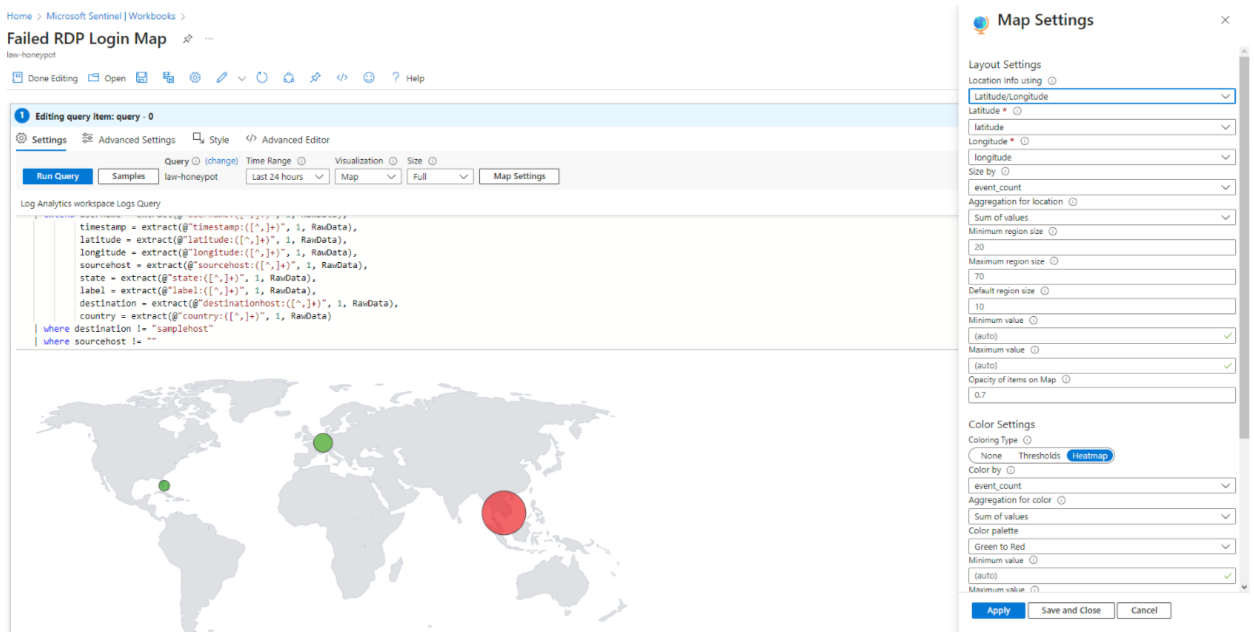


Data source: Logs; Resource type: Log Analytics; Log Analytics worksp...: law-honeypot; Time Range: Last 24 hours; Visualization: Set by q...; Size: Medium

```
        latitude = extract(@"latitude:([^,]+)", 1, RawData),
        longitude = extract(@"longitude:([^,]+)", 1, RawData),
        sourcehost = extract(@"sourcehost:([^,]+)", 1, RawData),
        state = extract(@"state:([^,]+)", 1, RawData),
        label = extract(@"label:([^,]+)", 1, RawData),
        destination = extract(@"destinationhost:([^,]+)", 1, RawData),
        country = extract(@"country:([^,]+)", 1, RawData)
| where destination != "samplehost"
| where sourcehost != ""
| summarize event_count=count() by latitude, longitude, sourcehost, label, destination, country, username
```

latitude	longitude	sourcehost	label	destination	country	username	event_count
49.24021	6.98515	54.37.197.205	Germany - 54.37.197.205	Honeypot	Germany	ADMINISTRATOR	16
26.20338	-80.11792	134.56.248.101	United States - 134.56.248.101	Honeypot	United States	hacker	2
26.20338	-80.11792	134.56.248.101	United States - 134.56.248.101	Honeypot	United States	lukeadmin	2
11.55637	104.92821	103.224.28.122	Cambodia - 103.224.28.122	Honeypot	Cambodia	administrator	88

From here, I clicked on the “Visualization” option and selected “Map.” All that needed to be done here was just to configure the map settings. This can be seen in the screenshot below.



Query: law-honeypot
Time Range: Last 24 hours
Visualization: Map
Size: Full

Log Analytics workspace Logs Query

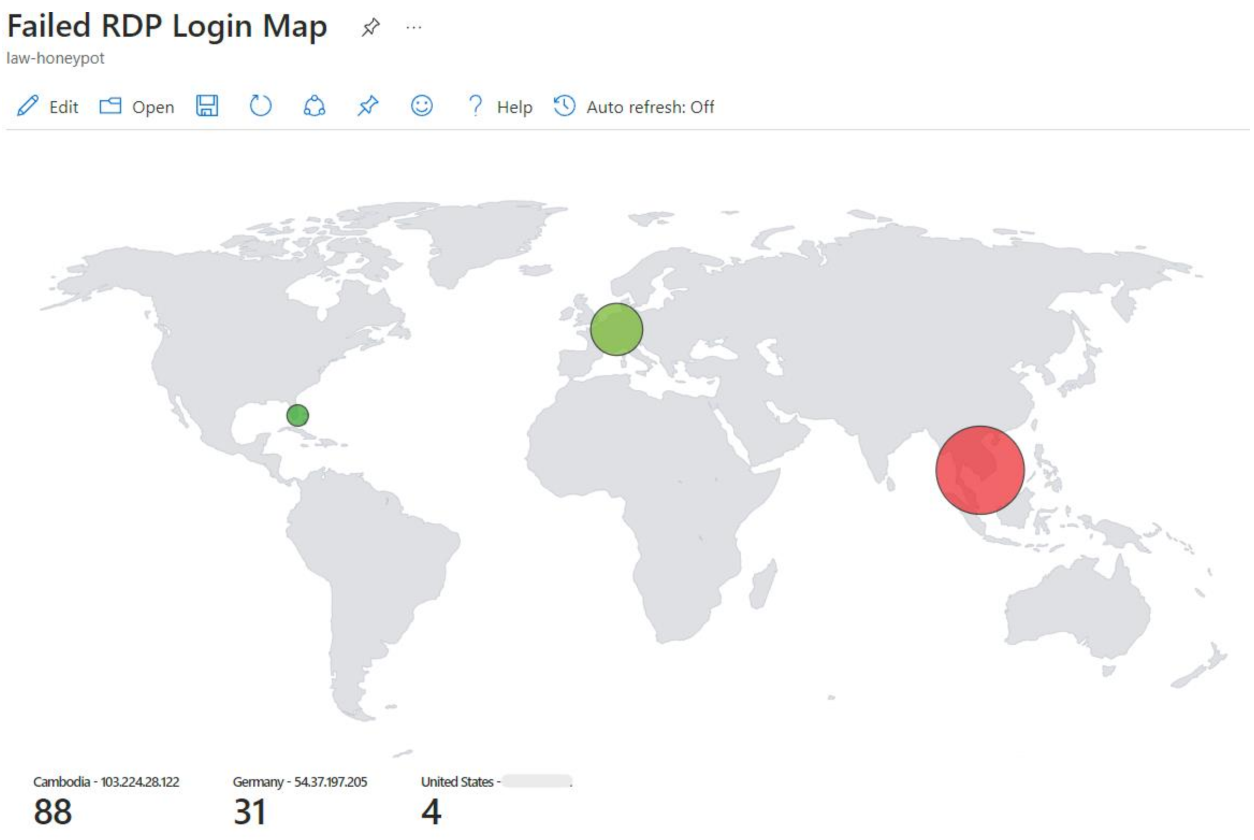
```
            timestamp = extract(@"timestamp:([^,]+)", 1, RawData),
            latitude = extract(@"latitude:([^,]+)", 1, RawData),
            longitude = extract(@"longitude:([^,]+)", 1, RawData),
            sourcehost = extract(@"sourcehost:([^,]+)", 1, RawData),
            state = extract(@"state:([^,]+)", 1, RawData),
            label = extract(@"label:([^,]+)", 1, RawData),
            destination = extract(@"destinationhost:([^,]+)", 1, RawData),
            country = extract(@"country:([^,]+)", 1, RawData)
| where destination != "samplehost"
| where sourcehost != ""
```

Map Settings

Layout Settings
Location Info using: Latitude/Longitude
Latitude *: latitude
Longitude *: longitude
Size by: event_count
Aggregation for location: Sum of values
Minimum region size: 20
Maximum region size: 70
Default region size: 10
Minimum value: (auto)
Maximum value: (auto)
Opacity of items on Map: 0.7

Color Settings
Coloring Type: None Thresholds [✓ Heatmap]
Color by: event_count
Aggregation for color: Sum of values
Color palette: Green to Red
Minimum value: (auto)
Maximum value

After I configured the settings, the map was finally finished. Interestingly enough, there were already attackers attempting to log into the honeypot. I also set the map to refresh every 5 minutes. Below is a picture of the results.

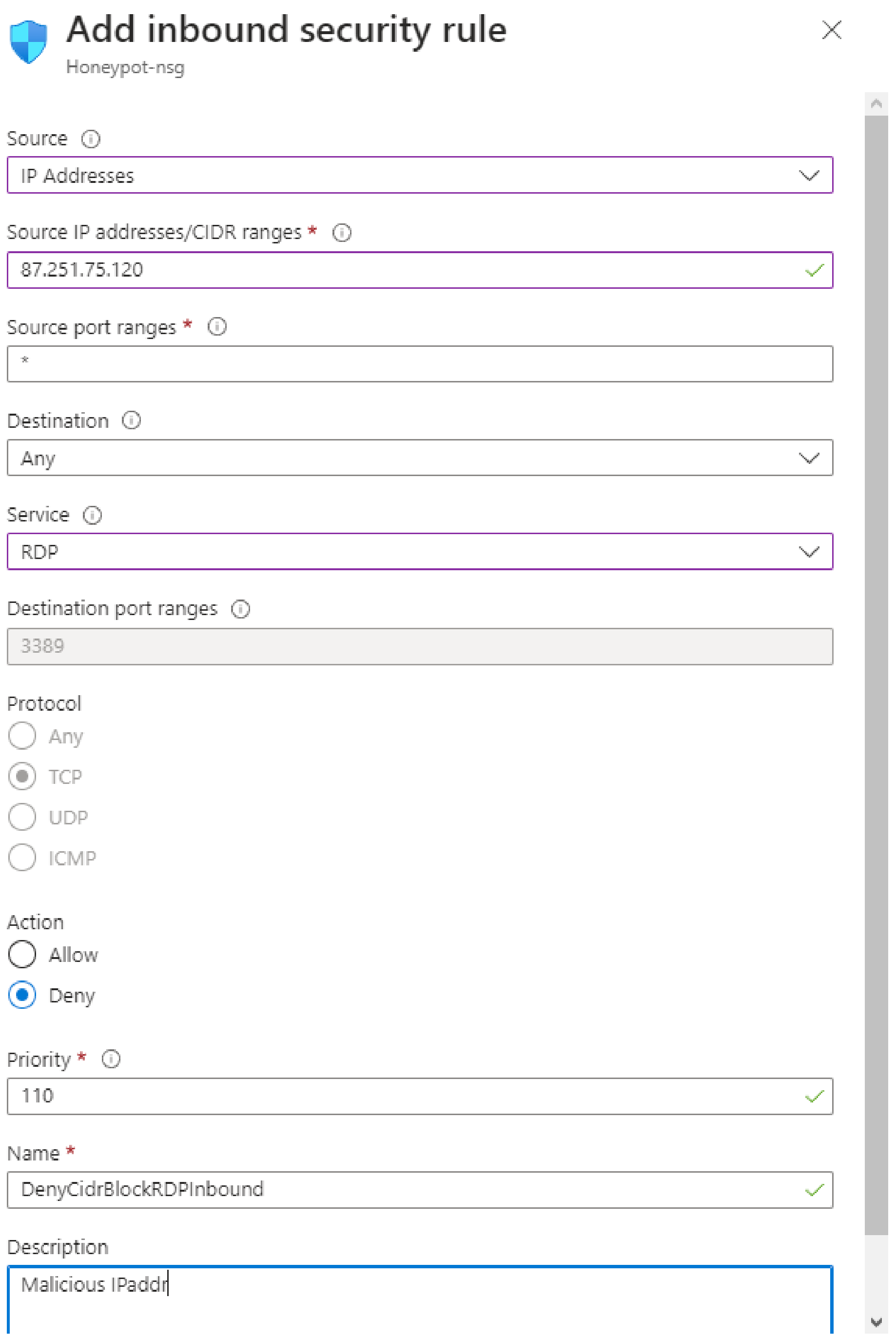


Cambodia - 103.224.28.122
88

Germany - 54.37.197.205
31

United States - [redacted]
4

After analyzing this data, it is time to take action and remediate the threats. To do this, I blocked the inbound IP addresses attempting to connect to the honeypot by creating firewall rules. This can be seen in the screenshot below.



Add inbound security rule

Honeypot-nsg

Source: IP Addresses

Source IP addresses/CIDR ranges *: 87.251.75.120

Source port ranges *: *

Destination: Any

Service: RDP

Destination port ranges: 3389

Protocol: Any [✓ TCP] UDP ICMP

Action: Allow [✓ Deny]

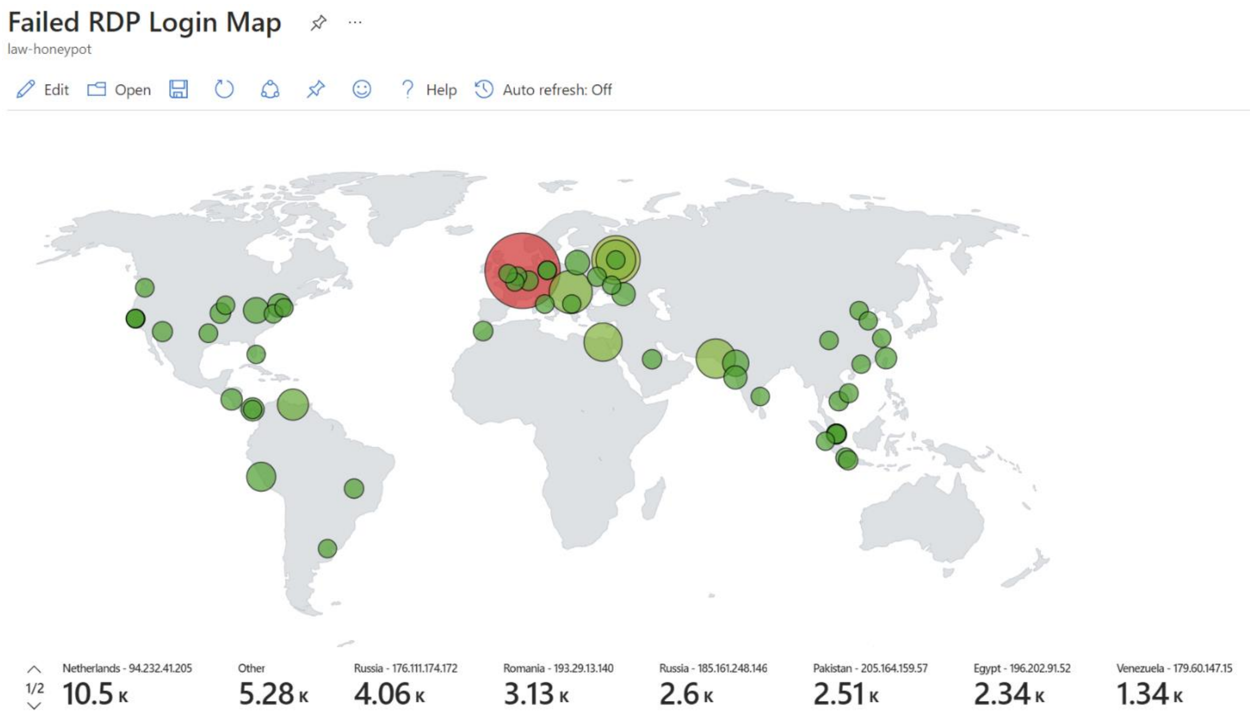
Priority *: 110

Name *: DenyCidrBlockRDPInbound

Description: Malicious IPaddr

The best way to remediate in a real-world scenario would be to deny all inbound connections and only allow verified IP addresses. But for the purpose of this project, I want the machine to get attacked.

I let the machine continue to get attacked for about 3 days and here are the results of that.



Netherlands - 94.232.41.205	Other	Russia - 176.111.174.172	Romania - 193.29.13.140	Russia - 185.161.248.146	Pakistan - 205.164.159.57	Egypt - 196.202.91.52	Venezuela - 179.60.147.15
10.5 K	5.28 K	4.06 K	3.13 K	2.6 K	2.51 K	2.34 K	1.34 K

As you can see in the screenshot, there are quite a few countries joining the party here. Based on this sample data, it appears that Russia and the Netherlands are the two loudest countries which is not very surprising.

<u>Lessons Learned</u>

This project was incredibly helpful in gaining a much better understanding of SIEMs. It is no surprise that a SIEM is an analyst's best friend due to the abundant amount of security information it displays. I am now much more confident operating a SIEM and running queries after this project. To recap what was accomplished in this project, I successfully:

- Configured the honeypot VM.
- Configured the log repository.
- Set up Microsoft Sentinel (SIEM).
- Configured log forwarding from the VM to Microsoft Sentinel.

- Configured geolocation API to translate Ip addresses to geographic locations.
- Queried the security logs.
- Configured Sentinel workbook to display geographic data.
- Analyzed the threat map.
- Remediated the threats.

This project was heavily inspired by the YouTuber Josh Madakor. He posted a video outlining how to do this project step by step and I highly recommend checking out the video.

https://www.youtube.com/watch?v=RoZeVbbZ0o0&t=306s