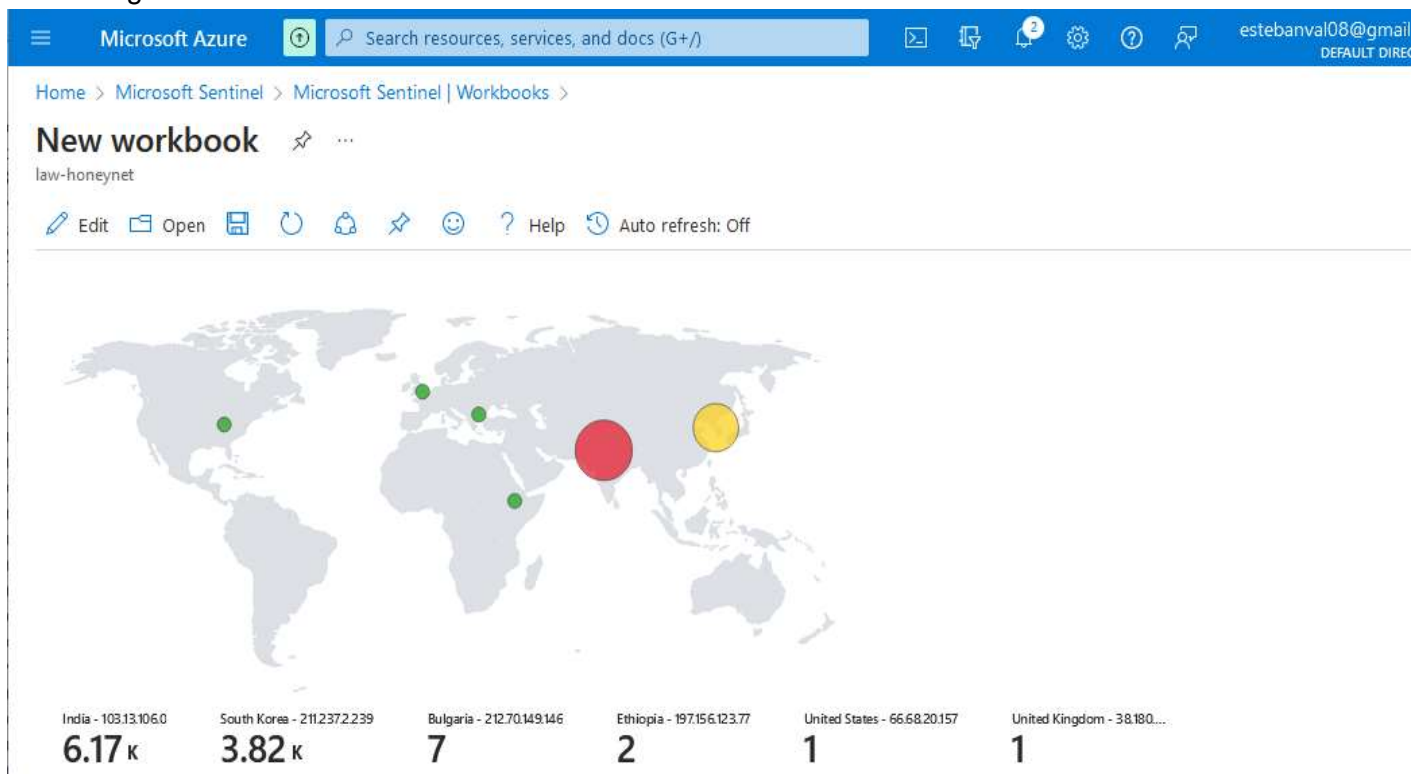


# Create and monitor a honeypot in Microsoft Azure

## Project description

The security team at my organization decided that we need to take proactive steps to understand attack techniques and detect threats as early as possible, especially since a new internet-facing host has just been deployed into the network and the team recently reported unusual, failed RDP (Remote Desktop Protocol) login attempts. To aid in this, I plan to create a honeypot machine with similar characteristics to the host, and open port 3389 (RDP), to lure attackers in and learn more about their hacking tactics, techniques, and procedures. By the end, we can expect a Microsoft Sentinel map illustration, showcasing the geolocation of potential attackers' IP addresses and the number of unsuccessful attempts to our honeypot. It should look something like this:



## Step 1: Build Azure Virtual Machine


The first step in creating our honeypot is to navigate to [portal.azure.com](https://portal.azure.com) and register for a free account, Microsoft Azure gives us a free \$200 credit to try their services so there shouldn't be any cost in case you wish to follow along.


After setting up our account, on our Azure dashboard, we will search for “**Virtual Machines**” in the search bar and click the first option. Then, we will click the “**Create**” button as shown below to start setting up our machine:


## Virtual machines


Default Directory

[+ Create](#) [Switch to classic](#) [Reservations](#) [Manage view](#) [Refresh](#) [Export to CSV](#) [Open query](#) [...](#)

 **Azure virtual machine**  
Create a virtual machine hosted by Azure

 **Azure virtual machine with preset configuration**  
Create a virtual machine with presets based on your workloads


 **Azure Arc virtual machine**  
Create a new Azure Arc virtual machine in one of your non-Azure environments

 **Azure VMware Solution virtual machine**  
Create a VMware virtual machine hosted by Azure

Type equals **all** [Add filter](#) [More \(2\)](#)

No grouping [List view](#)

Subscription [↑↓](#) Resource group [↑↓](#) Location [↑↓](#) Status [↑↓](#) Op



### No virtual machines to display

Create a virtual machine that runs Linux or Windows. Select an image from the marketplace or use your own customized image.

[Create](#) [Learn more about Windows virtual machines](#) [Learn more about Linux virtual machines](#)

Next, we must specify the VM's basic details, such as creating a new group to assign the VM to, name the VM, choose your region, and for this project I am assigning Windows 10 Pro as the OS for the honeypot. This is how the configuration should look like:

### Create a virtual machine

#### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ 

Azure subscription 1

Resource group \* ⓘ 

(New) Honeynet  
[Create new](#)

#### Instance details

Virtual machine name \* ⓘ 

honeynet-vm

Region \* ⓘ 

(US) West US 3

Availability options ⓘ 

Availability zone

Availability zone \* ⓘ 

Zones 1

☒ You can now select multiple zones. Selecting multiple zones will create one VM per zone. [Learn more](#)

Security type ⓘ 

Trusted launch virtual machines  
[Configure security features](#)

Image \* ⓘ 

Windows 10 Pro, version 22H2 - x64 Gen2 (free services eligible)  
[See all images](#) | [Configure VM generation](#)

VM architecture ⓘ 

☐ Arm64  
☒ x64  

Arm64 is not supported with the selected image.

Run with Azure Spot discount ⓘ 

☐

Esteban E. Valverde

Following , we will move to the next screen, where we will assign a username and password for the root account and assign the inbound port to “**RDP (3389)**” since we will be monitoring for failed RDP attempts.

**Administrator account**

Username \* ⓘ  ✓

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 \*  ✓

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

Next, since we are opening our computer to the whole internet, we have to create a new Network Interface Controller security group. For this, we'll choose “**Advanced**” under ‘**NIC network security group**’, and “**Create new**”:

[Home](#) > [Virtual machines](#) >

## Create a virtual machine ...



Basics Disks **Networking** Management Monitoring Advanced Tags Review + create

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution. [Learn more](#)

### Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network \* ⓘ  [Create new](#)

Subnet \* ⓘ

Public IP ⓘ  [Create new](#)

NIC network security group ⓘ ☐ None ☐ Basic ☒ Advanced

Configure network security group \*  [Create new](#)

Delete public IP and NIC when VM is deleted ⓘ ☐

Next, we will want to delete any default inbound/outbound rules, and then select “**Add an inbound rule**”. We're going to make sure the following settings are applied:

**Source:** Any

**Source Port Ranges:** \*

**Destination:** Any

Esteban E. Valverde

**Service:** Custom

**Destination Port Ranges:** \*

**Protocol:** Any

**Action:** Allow

**Priority:** 100

**Name:** (name your rule).

**Description:** (add for your reference)

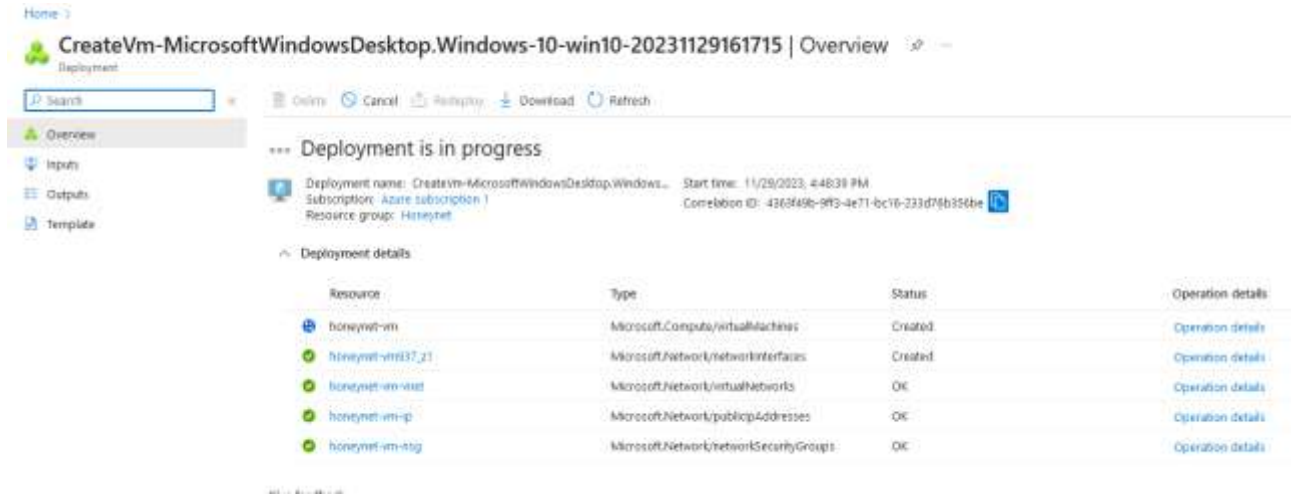
Then, we will press '**Review**', and '**Create**'.

The screenshot displays the Azure portal interface for creating a network security group (NSG) rule. On the left, the 'Create network security group' page is visible, showing the NSG name 'honeynet-vm-nsg' and buttons to add inbound or outbound rules. The main focus is the 'Add inbound security rule' dialog box, which is titled 'Add inbound security rule' and has a close button in the top right corner. The dialog contains the following fields and options:

- Source:** A dropdown menu set to 'Any'.
- Source port ranges:** A text input field containing an asterisk (\*).
- Destination:** A dropdown menu set to 'Any'.
- Service:** A dropdown menu set to 'Custom'.
- Destination port ranges:** A text input field containing an asterisk (\*), with a green checkmark indicating it is valid.
- Protocol:** A radio button selection with 'Any' selected.
- Action:** A radio button selection with 'Allow' selected.
- Priority:** A text input field containing '100', with a green checkmark indicating it is valid.
- Name:** A text input field containing 'DANGER\_ANY\_IN', with a green checkmark indicating it is valid.
- Description:** A text input field containing 'Will allow all traffic to enter virtual machine'.

At the bottom of the dialog, there are 'Add' and 'Cancel' buttons, and a 'Give feedback' link.

Our virtual machine is now being deployed, this may take a couple of minutes so we will move on to creating our Log Analytics Workspace (LAW) in the meantime.



## Step 2: Create Log Analytics Workspace

The next step is to create a log analytics workspace so we can ingest and assign the logs to and analyze later with Azure Sentinel. We will search for “**LAW**” on the search bar and click the first result, we will create a new LAW, assign it to the same region and resource group we previously created, and name it. Then we will click ‘**Review+Create**’ and create once it passes validation.

[Home](#) > [Log Analytics workspaces](#) >

### Create Log Analytics workspace



[Basics](#) [Tags](#) [Review + Create](#)

**i** A Log Analytics workspace is the basic management unit of Azure Monitor Logs. There are specific considerations you should take when creating a new Log Analytics workspace. [Learn more](#)

With Azure Monitor Logs you can easily store, retain, and query data collected from your monitored resources in Azure and other environments for valuable insights. A Log Analytics workspace is the logical storage unit where your log data is collected and stored.

#### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

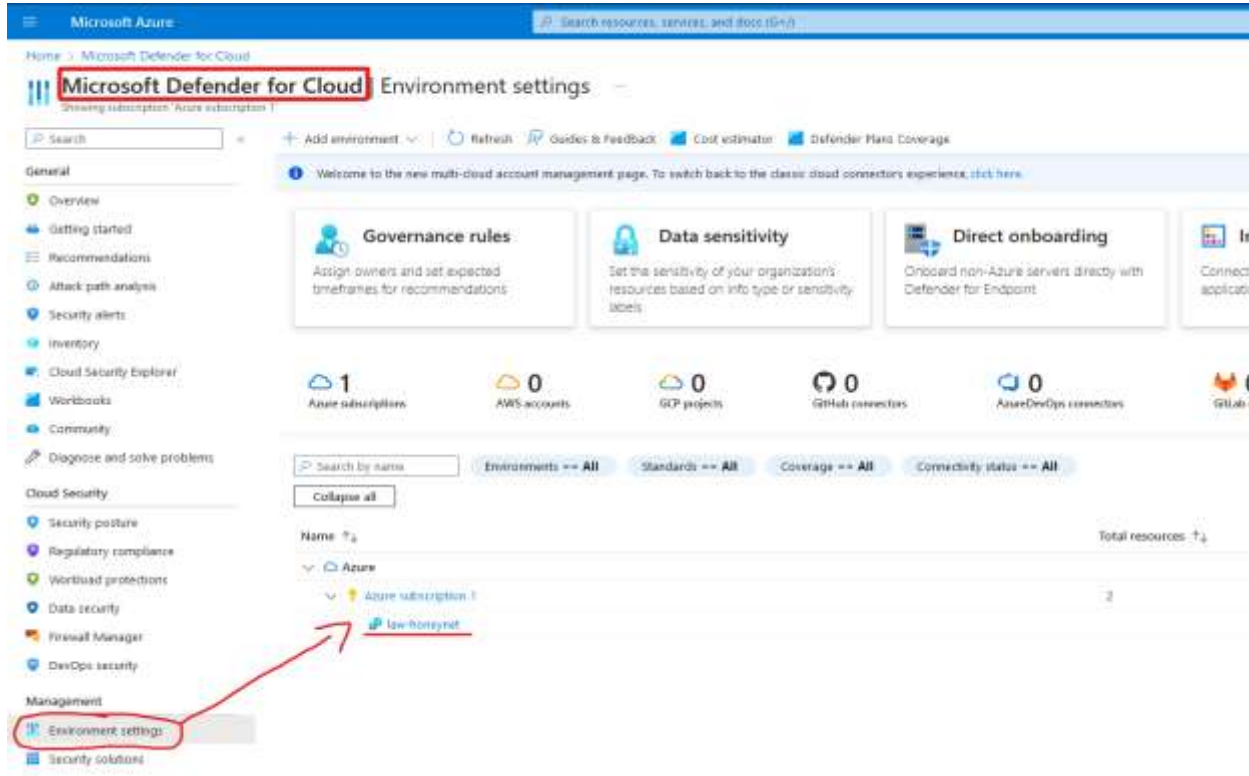
Resource group \* ⓘ  [Create new](#)

#### Instance details

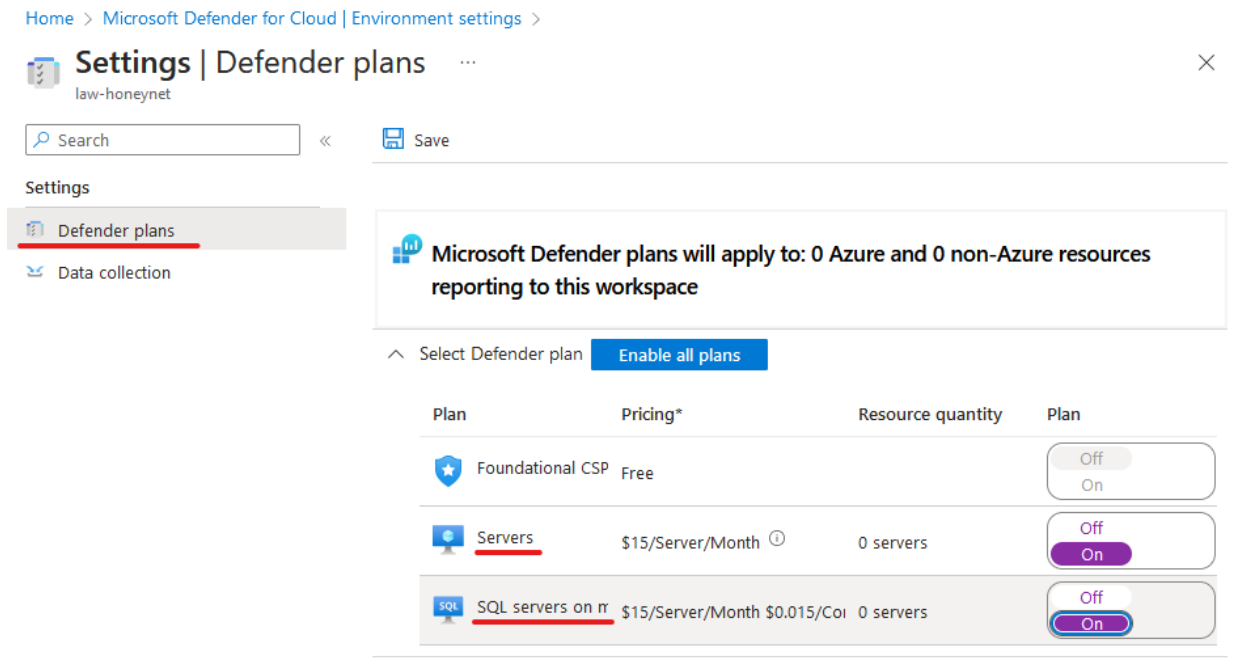
Name \* ⓘ  ✓

Region \* ⓘ

Next, we are going to search for “**Microsoft Defender For Cloud**” on the search bar and click the first result, once there, we will click on “**Environment Settings**” under ‘**Management**’, click the dropdown and select the LAW we previously created:



We want to make sure that our Servers and SQL Servers are both on, then we will click “**Data Collection**” on the left menu



Here, we'll want to click "All Events" and hit **SAVE**.

## Esteban E. Valverde

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.

Next, we will navigate back to Log Analytics Workspaces and choose the LAW we created earlier. We are going to connect it to our virtual machine. This should take a few minutes.

[Home](#) > [Log Analytics workspaces](#) > [law-honeynet | Virtual machines \(deprecated\)](#) >

### honeynet-vm

Virtual machine

**Connect** Disconnect Refresh

**i** Not connected

Status

Not connected

Workspace Name

None

Message

VM is not connected to Log Analytics.

Next, we will search for '**Microsoft Sentinel**' on the search bar, and our newly created LAW should be there, we'll click it, and connect our workspace to Microsoft Sentinel:



## Add Microsoft Sentinel to a workspace ...

[+ Create a new workspace](#) [Refresh](#)

Microsoft Sentinel offers a 31-day free trial. See [Microsoft Sentinel pricing](#) for more details.

Filter by name...				
Workspace ↑↓	Location ↑↓	ResourceGroup ↑↓	Subscription ↑↓	Directory ↑↓
law-honeynet	westus3	honeynet	Azure subscription 1	Default Directory

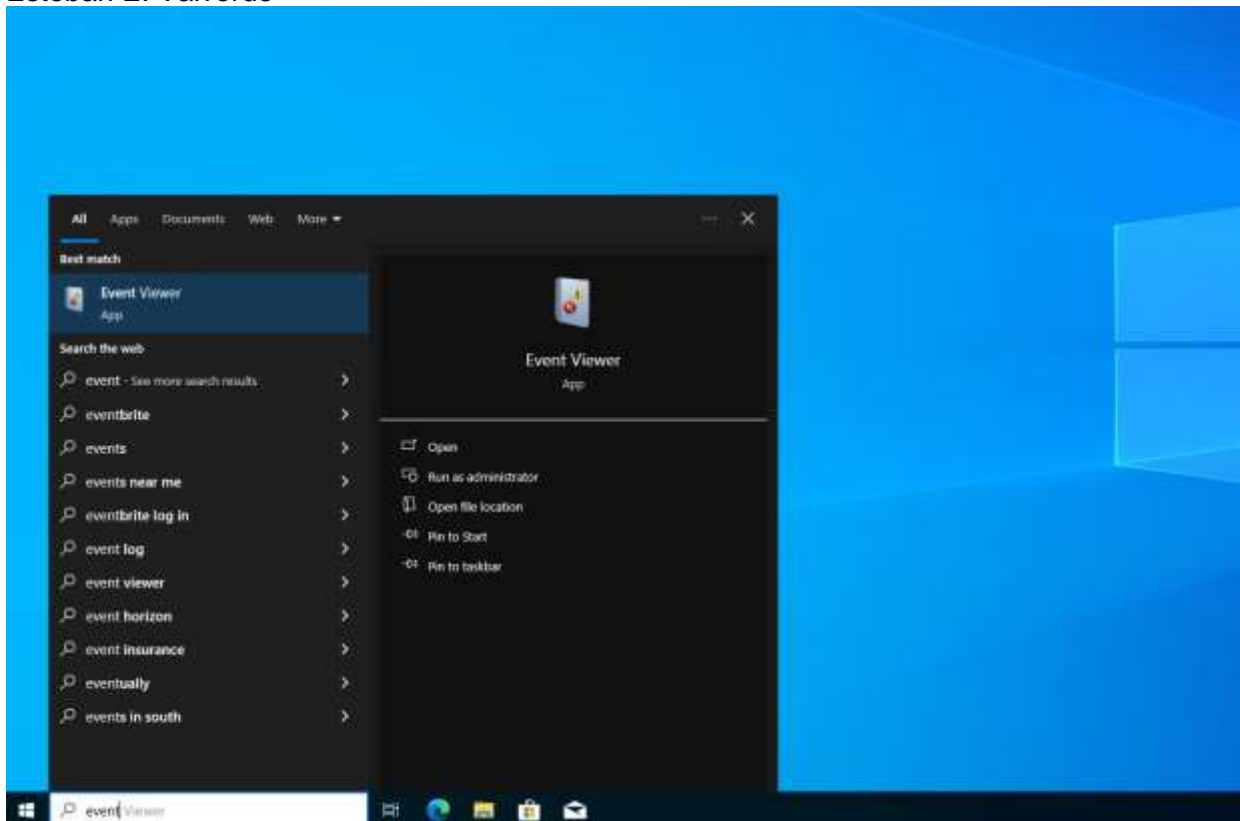
### Step 3: Connect and Configure Virtual Machine.

Our next step towards building our honeypot involves connecting it into the actual VM we created to properly configure the endpoint. To do this, first we will find and copy the machine's public IP address in our Azure's '**Virtual Machine**' menu. Then, we will use this address to connect to the VM through RDP, using the admin credentials we previously created to gain access.

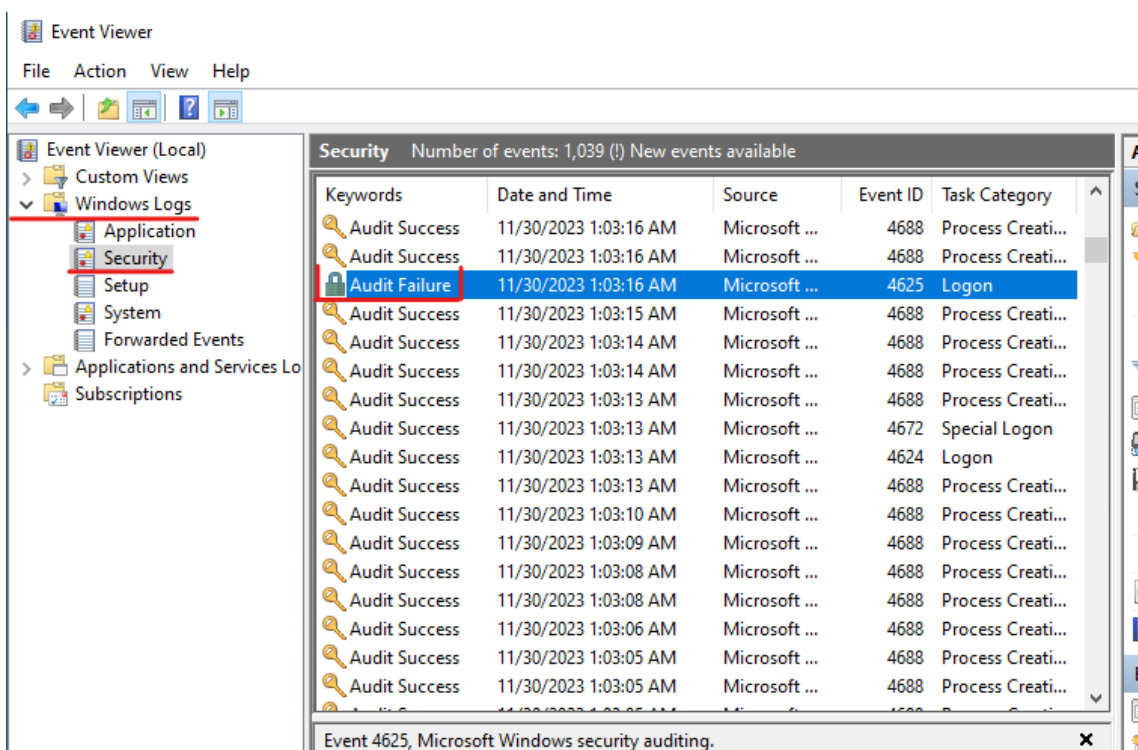


Once inside of our VM, we will search for the Event Viewer in Windows search bar, this is the storage for the logs of the metrics that we want to track (failed RDP attempts).





We are going to be analyzing any '**Audit Failure**' logs, under **Windows Logs > Security**



We can expand on these logs details and see important information such as the account name and failure reason, which was a bad username or password in this case. By monitoring and analyzing these failed RDP connection attempts, we gain insights into the persistence and determination of these attackers, ultimately strengthening our cybersecurity defenses.

Esteban E. Valverde

General Details

Logon Type: 3

Account For Which Logon Failed:

Security ID: NULL SID

Account Name: gusadmin

Account Domain: .

Failure Information:

**Failure Reason: Unknown user name or bad password.**

Status: 0xC000006D

Sub Status: 0xC000006A

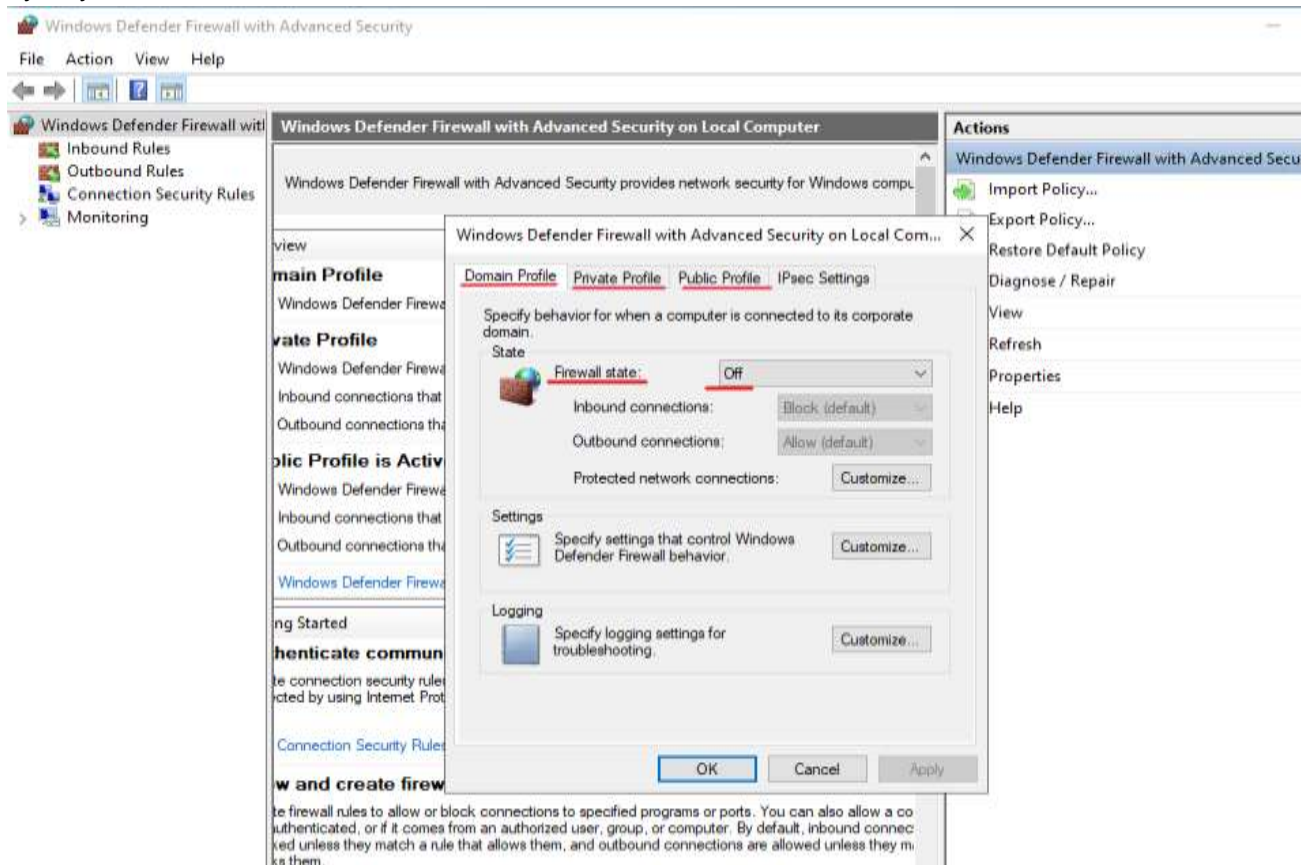
Process Information:

Caller Process ID: 0x0

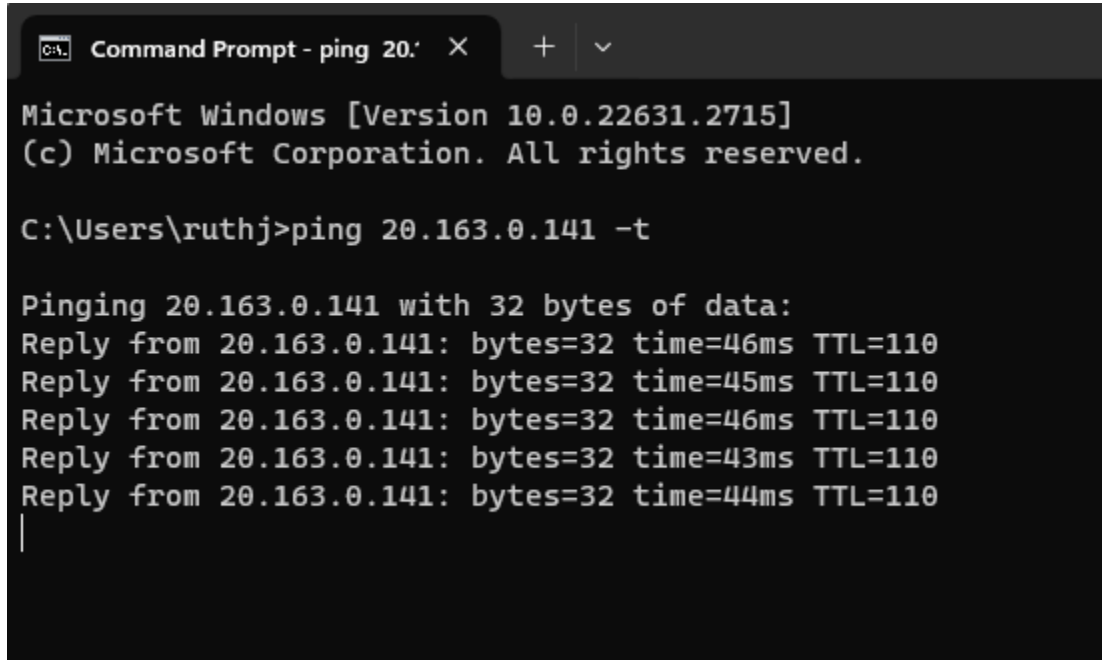
Caller Process Name: -

Network Information:

Next, I must make sure our domain, private, and public profile's firewalls are disabled so that it can be pinged by anyone on the internet.



We can verify our work by going to our host machine's terminal and using the ping command with the -t flag to make sure the VM can be pinged without issues.



```
Command Prompt - ping 20.163.0.141 X + v
Microsoft Windows [Version 10.0.22631.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ruthj>ping 20.163.0.141 -t

Pinging 20.163.0.141 with 32 bytes of data:
Reply from 20.163.0.141: bytes=32 time=46ms TTL=110
Reply from 20.163.0.141: bytes=32 time=45ms TTL=110
Reply from 20.163.0.141: bytes=32 time=46ms TTL=110
Reply from 20.163.0.141: bytes=32 time=43ms TTL=110
Reply from 20.163.0.141: bytes=32 time=44ms TTL=110
|
```

Then, I supported my project with a power shell script that will extract information from Windows Event Logs related to failed RDP login attempts, and create geographical data based on the source IP address. Here are the key components of the script:

### 1. Importing Necessary Modules:

The script begins by importing the required PowerShell modules, including , , and .

### 2. Setting Variables:

: This variable specifies the name of the Windows Event Log you want to target. In this case, it's set to 'Security' to access security-related events.

### 3. Querying Event Logs:

The script queries the 'Security' event log for specific event IDs related to failed RDP login attempts. It uses the cmdlet to filter the events and stores the results in the variable.

### 4. Iterating Through Events:

The script enters a loop to process each event in the array. It extracts relevant information from each event, such as the username, timestamp, source IP address, and more.

### 5. Extracting Data:

Regular expressions are employed to extract specific details from the event description, which is stored in the property of the event. Information like the username, IP address, and other relevant data is extracted using regex patterns.

### 6. Storing Data:

Esteban E. Valverde

The extracted data is stored in a custom log file named 'failed\_rdp.log' located in the directory. The script appends this information to the log file.

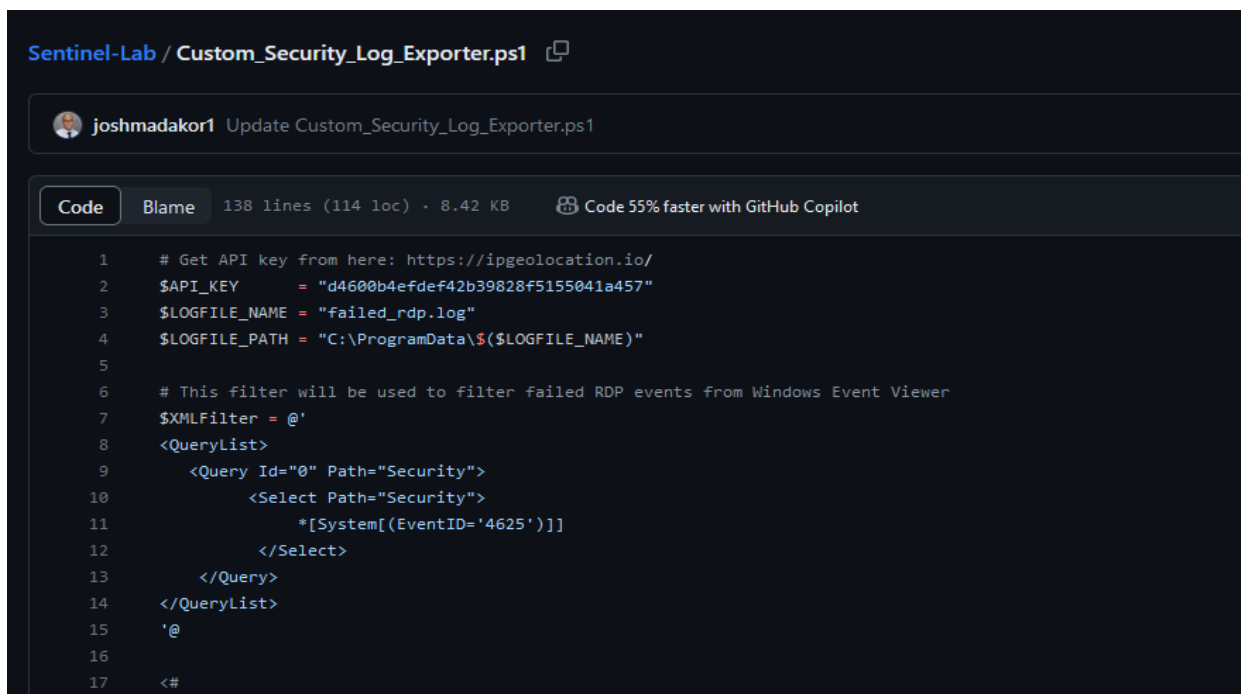
## 7. Geolocation Data:

The script also includes logic for obtaining geolocation data based on the extracted IP address. It appears that this feature isn't implemented directly in the script, but it instructs users to obtain an API key from '<https://ipgeolocation.io/>' and paste it into the script for future use. This API key would be used to determine the geographical location associated with each IP address.

## 8. Running the Script:

To execute the script, users can save it as a '.ps1' file and run it in a PowerShell environment. The script runs through the defined event logs, extracts the necessary information, and appends it to the 'failed\_rdp.log' file.

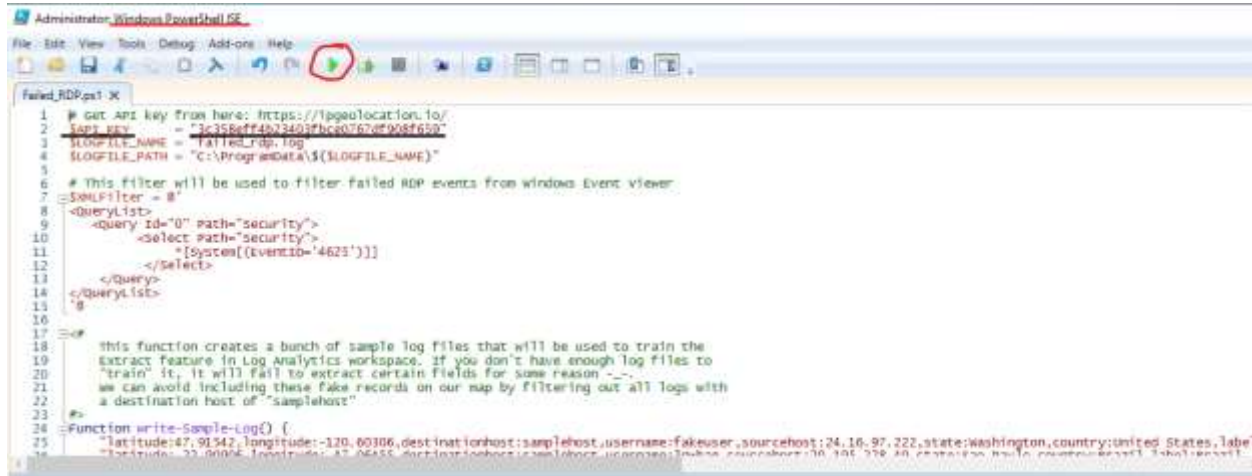
The script can be found here: [https://github.com/joshmadakor1/Sentinel-Lab/blob/main/Custom\\_Security\\_Log\\_Exporter.ps1](https://github.com/joshmadakor1/Sentinel-Lab/blob/main/Custom_Security_Log_Exporter.ps1)

A screenshot of a GitHub repository page for the file 'Sentinel-Lab / Custom\_Security\_Log\_Exporter.ps1'. The page shows the file's metadata, including the author 'joshmadakor1', the commit message 'Update Custom\_Security\_Log\_Exporter.ps1', and statistics: '138 lines (114 loc) · 8.42 KB'. A badge indicates 'Code 55% faster with GitHub Copilot'. The code is displayed in a dark-themed editor with line numbers 1 through 17. The script is a PowerShell file that sets an API key, a log file name, and a path, and defines an XML filter for Windows Event Viewer logs with EventID '4625'.

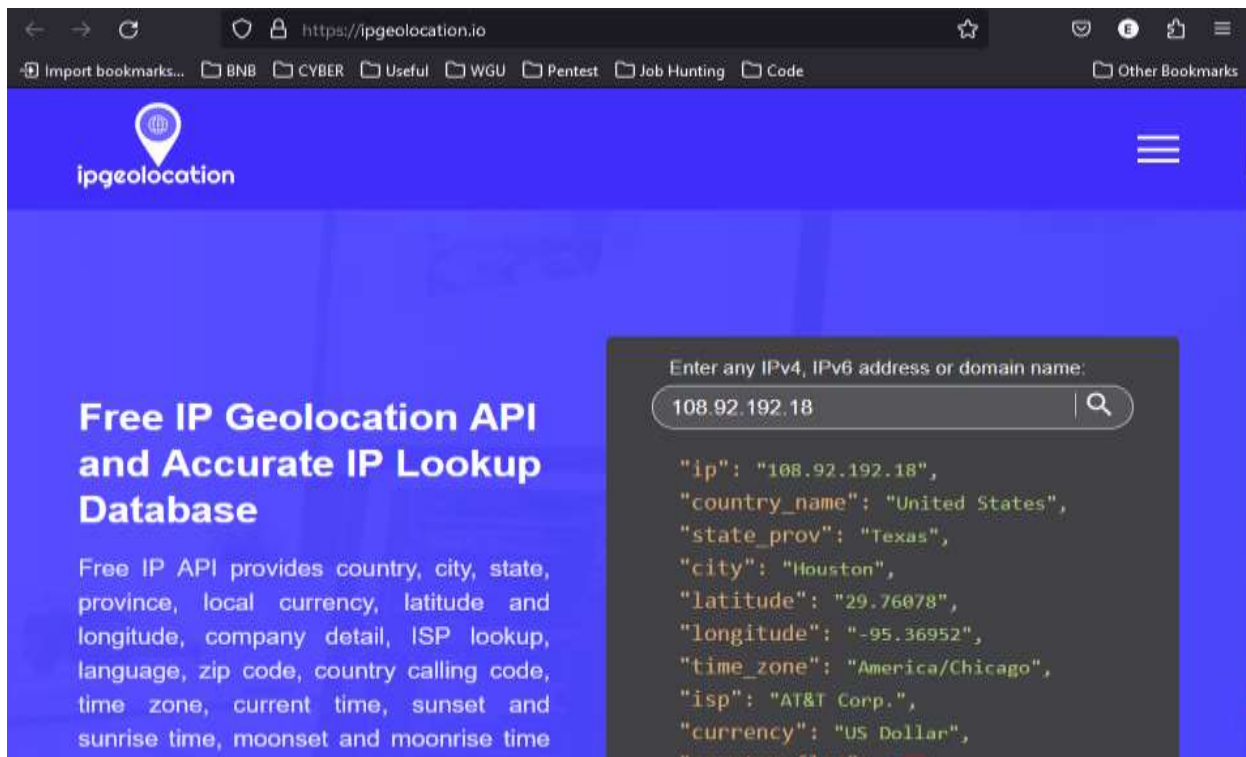
```
1 # Get API key from here: https://ipgeolocation.io/
2 $API_KEY = "d4600b4efdef42b39828f5155041a457"
3 $LOGFILE_NAME = "failed_rdp.log"
4 $LOGFILE_PATH = "C:\ProgramData\$($LOGFILE_NAME)"
5
6 # This filter will be used to filter failed RDP events from Windows Event Viewer
7 $XMLFilter = @"
8 <QueryList>
9   <Query Id="0" Path="Security">
10     <Select Path="Security">
11       *[System[(EventID='4625')]]
12     </Select>
13   </Query>
14 </QueryList>
15 '@
16
17 <#
```

I copied this code and pasted it into a new .ps1 file inside my VM's PowerShell ISE. Then, I went over to <https://ipgeolocation.io> and signed up for a free account to get my API key to insert into the code for proper IP geolocation.

Esteban E. Valverde



```
1 # Get API key from here: https://ipgeolocation.io/
2 $API_KEY = "3c358eff4922407f1c40767ef909df659"
3 $LOGFILE_NAME = "failed_rdp_log"
4 $LOGFILE_PATH = "c:\ProgramData\${LOGFILE_NAME}"
5
6 # This filter will be used to filter failed RDP events from windows Event viewer
7 $XMLFilter = @"
8 <QueryList>
9   <Query Id="0" path="security">
10     <Select path="security">
11       *[System[(EventID='4623')]]
12     </Select>
13   </Query>
14 </QueryList>
15 "@
16
17 #
18 # this function creates a bunch of sample log files that will be used to train the
19 # extract feature in Log Analytics workspace. If you don't have enough log files to
20 # "train" it, it will fail to extract certain fields for some reason ---
21 # we can avoid including these fake records on our map by filtering out all logs with
22 # a destination host of "samplehost"
23 #
24 Function Write-Sample-Log {
25   "latitude:47.91542,longitude:-120.60306,destinationhost:samplehost,username:fakeuser,sourcehost:24.16.97.222,state:Washington,country:United States,label:"
26   "latitude:22.90094,longitude:87.85455,destinationhost:samplehost,username:fakeuser,sourcehost:10.101.238.40,destinationhost:samplehost,label:samplest"
27 }
```



Free IP Geolocation API and Accurate IP Lookup Database

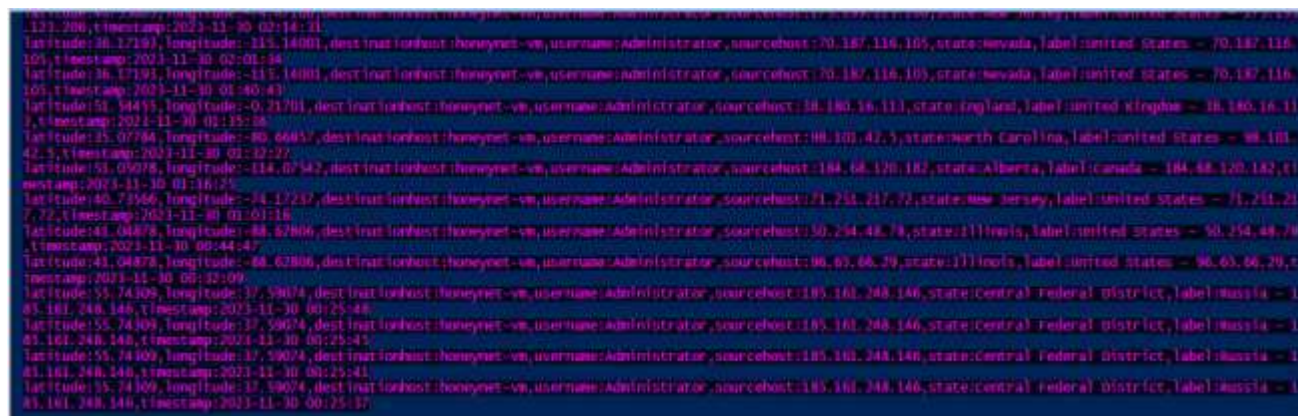
Free IP API provides country, city, state, province, local currency, latitude and longitude, company detail, ISP lookup, language, zip code, country calling code, time zone, current time, sunset and sunrise time, moonset and moonrise time

Enter any IPv4, IPv6 address or domain name:

108.92.192.18

"ip": "108.92.192.18",  
"country\_name": "United States",  
"state\_prov": "Texas",  
"city": "Houston",  
"latitude": "29.76078",  
"longitude": "-95.36952",  
"time\_zone": "America/Chicago",  
"isp": "AT&T Corp.",  
"currency": "US Dollar",  
"country\_flag": "🇺🇸"

The next step will be to simply run my code, and after only a few minutes I could already start to see the script in action inside of the VM's PowerShell CLI:

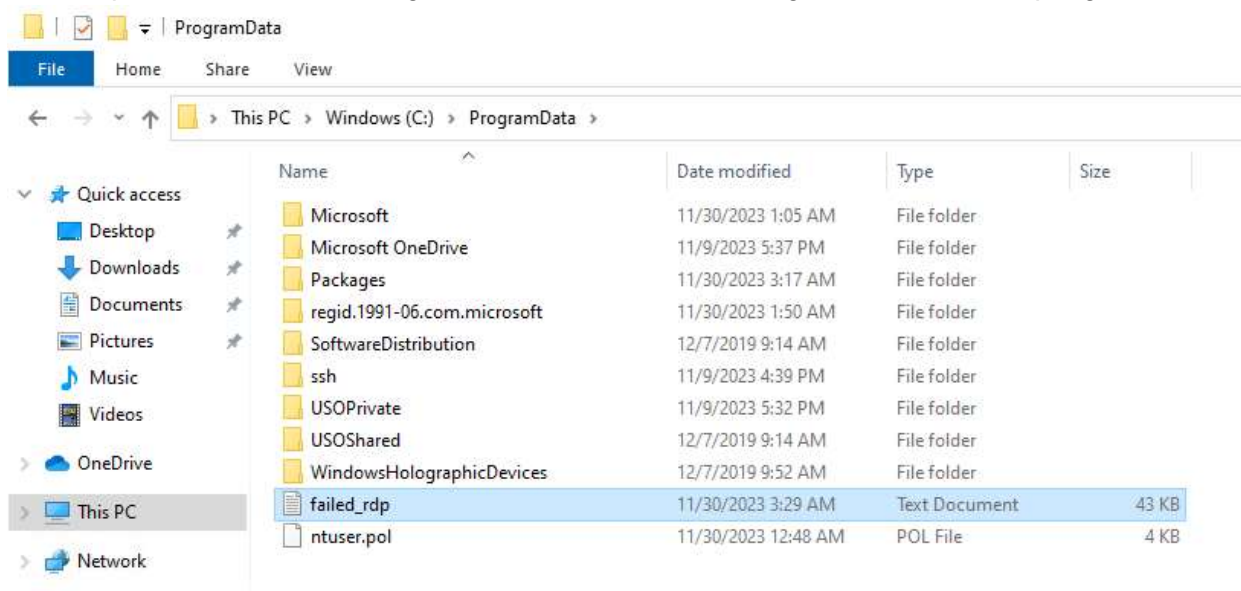


```
123.206,timestamp:2023-11-30 02:14:21
latitude:38.17193,longitude:-115.14001,destinationhost:honeywet-vm,username:Administrator,sourcehost:70.187.116.105,state:Nevada,label:United States - 70.187.116.105,timestamp:2023-11-30 02:01:34
latitude:38.17193,longitude:-115.14001,destinationhost:honeywet-vm,username:Administrator,sourcehost:70.187.116.105,state:Nevada,label:United States - 70.187.116.105,timestamp:2023-11-30 02:40:43
latitude:58.34419,longitude:-0.23701,destinationhost:honeywet-vm,username:Administrator,sourcehost:18.180.16.111,state:England,label:United Kingdom - 18.180.16.111,timestamp:2023-11-30 02:15:28
latitude:35.07744,longitude:-85.66853,destinationhost:honeywet-vm,username:Administrator,sourcehost:98.102.42.3,state:North Carolina,label:United States - 98.102.42.3,timestamp:2023-11-30 02:12:27
latitude:55.09276,longitude:-114.07542,destinationhost:honeywet-vm,username:Administrator,sourcehost:184.68.120.182,state:Alberta,label:Canada - 184.68.120.182,timestamp:2023-11-30 01:16:25
latitude:40.72546,longitude:-74.17237,destinationhost:honeywet-vm,username:Administrator,sourcehost:71.251.217.72,state:New Jersey,label:United States - 71.251.217.72,timestamp:2023-11-30 01:03:16
latitude:41.04876,longitude:-88.62806,destinationhost:honeywet-vm,username:Administrator,sourcehost:50.254.48.78,state:Illinois,label:United States - 50.254.48.78,timestamp:2023-11-30 00:44:47
latitude:41.04876,longitude:-88.62806,destinationhost:honeywet-vm,username:Administrator,sourcehost:96.65.66.39,state:Illinois,label:United States - 96.65.66.39,timestamp:2023-11-30 00:12:09
latitude:55.74309,longitude:37.59074,destinationhost:honeywet-vm,username:Administrator,sourcehost:185.161.248.146,state:Central Federal District,label:Russia - 185.161.248.146,timestamp:2023-11-30 00:25:48
latitude:55.74309,longitude:37.59074,destinationhost:honeywet-vm,username:Administrator,sourcehost:185.161.248.146,state:Central Federal District,label:Russia - 185.161.248.146,timestamp:2023-11-30 00:25:45
latitude:55.74309,longitude:37.59074,destinationhost:honeywet-vm,username:Administrator,sourcehost:185.161.248.146,state:Central Federal District,label:Russia - 185.161.248.146,timestamp:2023-11-30 00:25:41
latitude:55.74309,longitude:37.59074,destinationhost:honeywet-vm,username:Administrator,sourcehost:185.161.248.146,state:Central Federal District,label:Russia - 185.161.248.146,timestamp:2023-11-30 00:25:37
```



Esteban E. Valverde

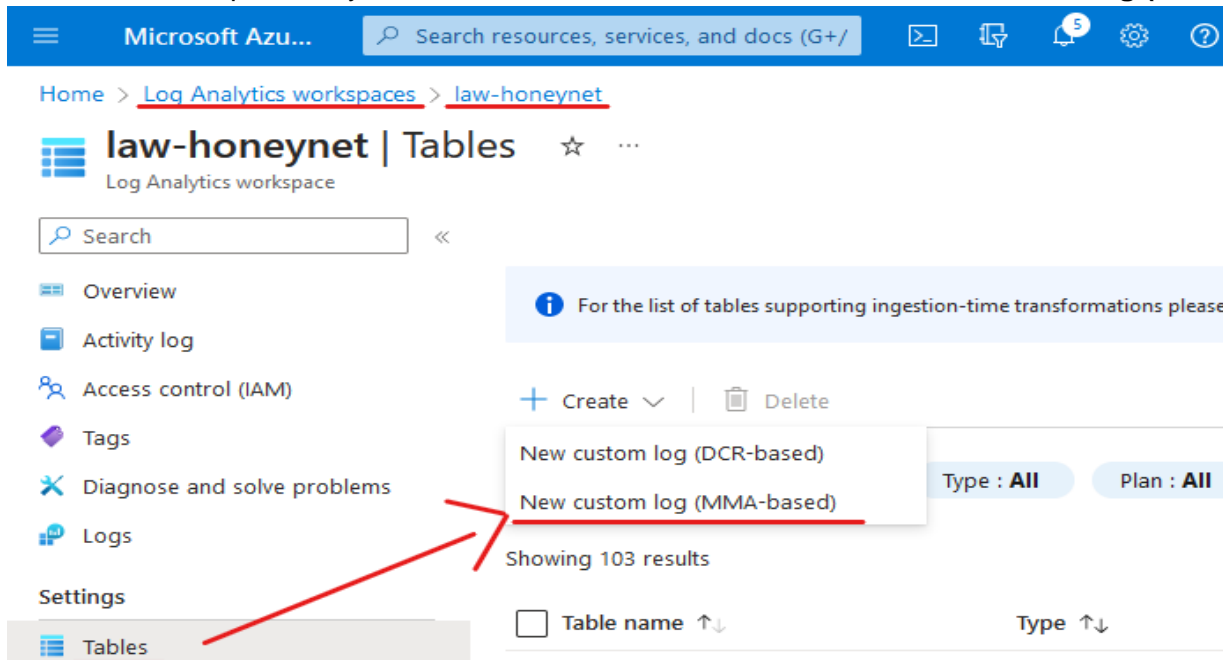
The script saves the desired logs into a hidden folder C:\ProgramData\failed\_rdp.log



## Step 4: Set up Custom Log Files and Query

Up next, before creating our log query, I must create a custom log file from the failed\_rdp.log file in our VM for Azure to collect our data in an organized way.

To do this, I first need to go back to **Home > Log Analytics Workspaces** in my Azure account and click the LAW we created previously. Select **Tables** on the left, the **Create > New Custom Log (MMA-based)**



It's going to ask us for a custom log file on our local host. Since we are looking to use the data from our Virtual Machine's failed RDP Logs, I am going to have to go back into my VM, copy the contents of the failed RDP Log, create a file on my local host, and then upload that file to Azure.

## Create a custom log ...

1 Sample 2 Record delimiter 3 Collection paths 4 Details 5 Review + Create

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.txt"



Just like in the script (unless it was modified), the file will be saved under C:\ProgramData\failed\_rdp.log:

## Create a custom log ...

✓ Sample ✓ Record delimiter 3 Collection paths 4 Details 5 Review + Create

Define one or more paths on the agent where it can locate the custom log. [Learn more](#)

### Collection paths

Type	Path
Windows	C:\programdata\failed_rdp.log ✓
Select type	

I then proceeded to name the custom log, and then reviewed and submitted the changes.

## Create a custom log ...

✓ Sample ✓ Record delimiter ✓ Collection paths 4 Details 5 Review + Create

Add a name and description to the custom log.

This name will be used for the log type, and will always end with \_CL to distinguish it as a custom log. [Learn more](#)

### Details

Custom log name \*

FAILED\_RDP\_WITH\_GEO

\_CL

Description

Description

After creating the log, we want to go back to the main log page, select logs from the left menu, and we'll be met with the query field.

We're going to build a query that will parse our log files to output the appropriate data to be able to plot to the map in sentinel.

The query we are going to run is:



Microsoft ... Search resources, services, and docs (G+)

Home > law-honeynet

law-honeynet | Logs Log Analytics workspace

Search

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Logs

Settings

Tables

Agents

Usage and estimated costs

Data export

Network isolation

New Query 1\*

Run Time range: Last 24 hours Save Share

```
1 FAILED_RDP_WITH_GEO_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 timestamp, label, country, state, sourcehost, username, destination, longitude, latitude
```

Queries History

In detail, we start the query by choosing the custom log table we created, in this case

“**FAILED\_RDP\_WITH\_GEO\_CL**”, then we will use “extend” to extend the data by extracting the username field from the RawData field (“**extract(@"username:([^\,]+)", 1, RawData)**”). We will repeat the section we used to extract the username field, and we are also going to extract the timestamp field, latitude, longitude, sourcehost, state, label, destination, and country.

After extracting all these fields, we will instruct the query to exclude any results where the ‘destination’ field is equal to ‘samplehost’, since the PowerShell script I used earlier contains some sample logs that we do not want to illustrate. We also want to exclude any results where ‘sourcehost’ is an empty field to avoid any false positives. Finally, we will summarize the event count for better illustration.

The final output provides structured data that can be used for visualization and analysis, particularly for plotting geographical information related to failed RDP connection attempts on a map in Microsoft Sentinel.

## Step 5: Construct Sentinel Map

It’s time to start putting all our work together in Sentinel! I will start by going to Microsoft Sentinel in my Azure, clicking on ‘**Workbooks**’ on the left menu, and then I’ll hit the **Add Workbook** button.

Microsoft ... Search resources, services, and docs (G+) estebanval08@gmail.com DEFAULT DIRECTORY

Home > Microsoft Sentinel > Microsoft Sentinel

# Microsoft Sentinel | Workbooks

Selected workspace: 'law-honeynet'

Search Refresh Add Workbook Guides & Feedback

General

- Overview (Preview)
- Logs
- News & guides
- Search

Threat management

- Incidents
- Workbooks**
- Hunting
- Notebooks
- Entity behavior
- Threat intelligence
- MITRE ATT&CK (Preview)

0 My workbooks 0 Templates 0 Updates More content at Content hub

## Microsoft Sentinel Workbooks

### What is it?

Workbooks enable you to get instant visualization and analysis of data so that you can know what's happening across all your connected data sources. Workbooks provide you with the full power of tools with tables and charts that are built in to provide you with analytics for your logs and queries. You can either use out-of-the-box (OOTB) workbooks from content hub or customize an installed OOTB workbook by saving them or create a new workbook easily, from scratch or based on an existing workbook.

[Learn more](#) about Workbooks.

[Learn more](#) about OOTB content and Content hub.

Before adding my desired widget, I need to delete any default ones by clicking '**Edit**' and deleting the widgets. Then I will click '**Add**' and '**Add query**'

## New workbook



law-honeynet



This Azure Sentinel Report has no content.



Use the add button below to add items.

+ Add ▾

- Add text
- Add parameters
- Add links/tabs
- Add query
- Add metric
- Add group

Add query

I am going to paste the same query used earlier for the Custom Log Table, and then I am going to run it. After running it I will click '**Map**' under "Visualization", and then we will be able to see our map!

## New workbook

law-honeynet

Done Editing Open Settings Style Advanced Editor

1 Editing query item: query - 0

Settings Advanced Settings Style Advanced Editor

Run Query Samples Logs Data source Resource type Log Analytics works... Time Range Visualization Size

Column Settings

Log Analytics workspace Logs Query

```

FAILED_RDP_WITH_GEO_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) | where destination != "samplehost" | where sourcehost != "" | summarize(timestamp, label, country, state, sourcehost, username, destination, longitude, latitude)

```

timestamp label country state

2023-11-30 03:19:29	United States - 70.187.116.105	United States	Nevada
2023-11-30 03:12:39	United States - 38.124.40.8	United States	Illinois
2023-11-30 03:10:41	United States - 50.254.48.78	United States	Illinois
2023-11-30 03:03:26	Canada - 184.68.120.182	Canada	Alberta
2023-11-30 02:47:42	United States - 50.254.48.78	United States	Illinois
2023-11-30 02:37:19	United States - 50.254.48.78	United States	Illinois
2023-11-30 02:17:50	United States - 204.9.187.98	United States	Florida

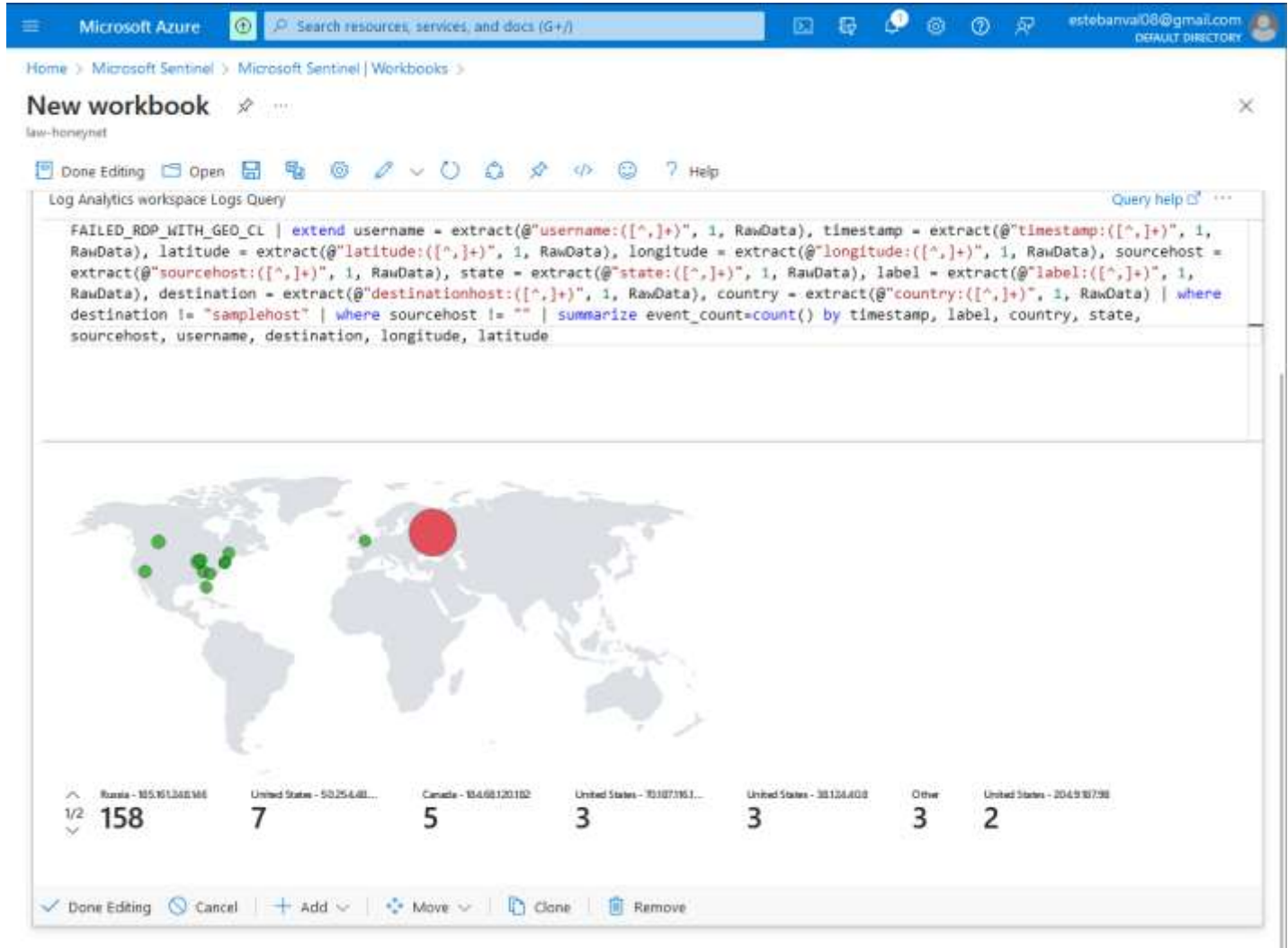
Visualization dropdown menu:

- Set by query
- Grid
- Area chart
- Bar chart
- Bar chart (Categorical)
- Bar chart (Unstacked)
- Line chart
- Pie chart
- Scatter chart
- Time chart
- Tiles
- Graph
- Map
- Text

Query help

After approximately 1 hour we can see very intriguing log data illustrated in our Sentinel map about failed RDP login attempts to our honeypot VM. We can infer that the IP address in Russia is performing some sort of brute force attack due to the amount of login attempts in such a short period of time.





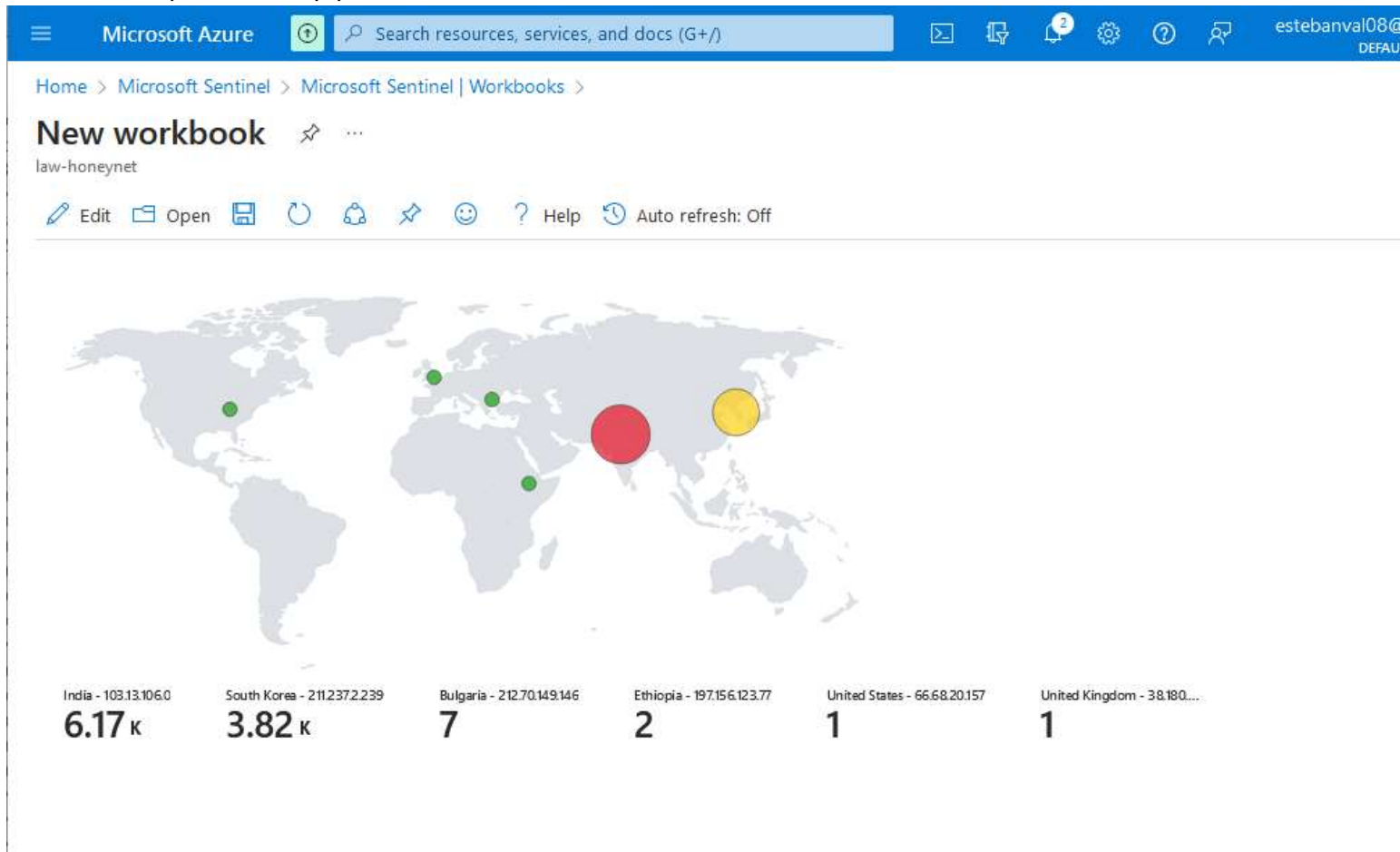
Going back to our PowerShell script inside of our VM we can observe real-time log recollection of these events.

```

latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:1.28272,longitude:103.85120,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:172.104.168.228,state:Central Singapore Community De
11-30 23:31:25
latitude:38.94886,longitude:-77.32938,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:96.126.72.91,state:Virginia,label:United States - 9
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:1.28272,longitude:103.85120,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:139.177.185.53,state:Central Singapore Community De
-30 23:30:39
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:29.76078,longitude:-95.36952,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:64.112.124.86,state:Texas,label:United States - 64
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:32.82222,longitude:-96.87004,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:104.249.63.207,state:Texas,label:United States - 10
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:40.97673,longitude:-73.69483,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:104.219.239.239,state:New York,label:United States - 10
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:39.05232,longitude:-77.48270,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:46.37.100.201,state:Virginia,label:United States - 4
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:1.28272,longitude:103.85120,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:139.177.187.139,state:Central Singapore Community De
11-30 23:26:18
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:40.78606,longitude:-74.07444,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:173.225.108.90,state:New Jersey,label:United States
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:51.50643,longitude:-0.12719,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:101.36.97.108,state:England,label:United Kingdom - 10
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:40.78606,longitude:-74.07444,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:173.225.108.90,state:New Jersey,label:United States
latitude:37.52812,longitude:126.96888,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:211.180.132.154,state:Seoul,label:South Korea - 211
latitude:1.28272,longitude:103.85120,destinationhost:honey-net-vm,username:ADMINISTRATOR,sourcehost:172.105.115.209,state:Central Singapore Community De
11-30 23:10:42
  
```

After roughly 12 hours we can observe a few different IP's that tried to login, but the two that definitely attract the most attention are the IP's from India and South Korea, since we can observe that they definitely employed an extensive brute force dictionary attack in an attempt to gain access to my honeypot.

**\*Note\*** my API key malfunctioned and I had to issue a new one, hence why we might observe a bit different data than the previous map picture.



## Summary

In conclusion, the project aimed to bolster the security team's proactive measures by creating and monitoring a honeypot in Microsoft Azure. Focused on understanding attack techniques and detecting threats, the initiative involved building an Azure Virtual Machine, creating a Log Analytics Workspace, and connecting and configuring the virtual machine for monitoring failed RDP login attempts.

The deployment of a custom PowerShell script allowed for the extraction of valuable information from Windows Event Logs, enabling the creation of geolocation data based on source IP addresses. The integration of this data into Azure Sentinel facilitated the construction of a comprehensive map illustrating the geographical distribution of potential attackers and the frequency of unsuccessful attempts on the honeypot.

Through meticulous steps, including setting up custom log files, crafting queries, and constructing a Sentinel map, the project showcased the effectiveness of honeypots in attracting and analyzing global threat actor traffic. Real-time observations revealed diverse and persistent attackers engaging in brute force attempts, emphasizing the project's success in providing actionable insights to strengthen cybersecurity defenses.