a1

December 8, 2022

1 Task A1 - Initial access - (Log analysis) Points: 10

1.1 Problem Statement

> We believe that the attacker may have gained access to the victim's network by phishing a legitimate users credentials and connecting over the company's VPN. The FBI has obtained a copy of the company's VPN server log for the week in which the attack took place. Do any of the user accounts show unusual behavior which might indicate their credentials have been compromised? Note that all IP addresses have been anonymized. Enter the username which shows signs of a possible compromise.

Given VPN logs, we need to return a username who is seemingly malicious. First, use Python to analyze the given file `data/vpn.log`.

[1]:
```
import pandas as pd
from datetime import datetime, timedelta
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
df = pd.read_csv("data/a1/vpn.log")
df.head()
```

[2]:

	Node	Username	Start Time	Duration	Service	Active	Auth	Real Ip	Vpn Ip	Proto	Port	Bytes Total	Error
0	openvpn-server	Doris.X	2022.06.27 07:48:08 EDT	37494.0	VPN	0	1	172.28.168.133	10.128.20.108	UDP	1194	2.030863e+09	NaN
1	openvpn-server	Kelly.G	2022.06.27 08:15:32 EDT	32314.0	VPN	0	1	172.19.185.189	10.128.20.194	UDP	1194	2.363123e+09	NaN
2	openvpn-server	Dorothy.D	2022.06.27 08:43:39 EDT	18184.0	VPN	0	1	172.22.90.19	10.128.20.43	UDP	1194	2.009805e+09	NaN
3	openvpn-server	James.V	2022.06.27 08:54:02 EDT	2266.0	VPN	0	1	172.25.206.19	10.128.20.78	UDP	1194	1.379042e+08	NaN
4	openvpn-server	Joan.P	2022.06.27 09:28:35 EDT	20074.0	VPN	0	1	172.23.234.95	10.128.20.106	UDP	1194	1.318902e+09	NaN

This result shows `data/vpn.log`'s columns. Some of the tables seem useless in terms of analysis, so let's remove them.

[3]:
```
df = df.drop(columns=["Node", "Service", "Active", "Proto", "Port", "Error"])
```

After this procedure, use `Start Time` and `Duration` to create a new column. This column name should be `End Time` opposition to `Start Time`.

[4]:
```
from dateutil import parser
df[["Duration"]] = df[["Duration"]].fillna(value=0)
df['Start Time'] = pd.to_datetime(df['Start Time'])
df["Duration"] = df["Duration"].apply(lambda x: timedelta(0, x))
end_time = df["Start Time"] + df["Duration"]
df.insert(2, "End Time", end_time)
```

Let's check `Start Time` and `End Time`. Assume you were an attacker, what do you do to deceive data? You may want to use VPN from multiple computers. See if there is an overlap between them.

[5]:
```
df = df.sort_values("Username")
usernames = df["Username"].unique()
suspect = []
for user in usernames:
    user_rows = df[df["Username"] == user]
    intervals = []
    for i, row in user_rows.iterrows():
        start_time = row["Start Time"]
        end_time = row["End Time"]
        real_ip = row["Real Ip"]
        intervals.append((start_time, end_time, real_ip))
    intervals.sort()
    # see if there is an overlap in intervals
    is_suspicious = False
    for i in range(1, len(intervals)):
        prev_left, prev_right, prev_ip = intervals[i - 1]
        cur_left, cur_right, cur_ip = intervals[i]
        if prev_left <= cur_left <= prev_right and prev_ip != cur_ip:
            is_suspicious = True
        if is_suspicious:
            suspect.append((user, prev_ip, cur_ip))
            break
print(suspect)
```

```
[('Moses.K', '172.30.122.56', '172.27.26.101')]
```

Yes! We've found a suspicious user that has an overlap in accessing. He seems to have enter VPN using two different IPs at the same time.


I COMPLETED
TASK
A1
OF THE
CODEBREAKER
CHALLENGE
2022
NATIONAL SECURITY AGENCY
UNITED STATES OF AMERICA