

Projeto 2 de SRC

Universidade de Aveiro

João Torrinhas, Diogo Torrinhas,



universidade de aveiro

Projeto 2 de SRC

Departamento de Electrónica, Telecomunicações e
Informática
Universidade de Aveiro

João Torrinhas, Diogo Torrinhas
(98435) joao.torrinhas@ua.pt, (98440) diogotorrinhas@ua.pt

June 28, 2023

Contents

1	Introdução	2
2	Análise do Dataset Sem Anomalias	3
2.1	Distribuição das portas usadas no DataSet	3
2.2	Distribuição dos protocolos usados no DataSet	4
2.3	Identificar o ip de origem com mais comunicações	4
2.4	Identificar o ip destino com mais comunicações	5
2.5	Tráfego durante diferentes períodos do dia	6
3	Exfiltração	7
3.1	O que é?	7
3.2	Regras SIEM	7
3.2.1	Exfiltração por DNS	7
3.2.2	Resultados Exfiltração por DNS	7
3.2.3	Exfiltração por HTTPS	8
3.2.4	Resultados Exfiltração por HTTPS	8
3.2.5	Outros meios de detecção de Exfiltração	11
3.2.6	Resultados	12
4	Botnets	13
4.1	O que é?	13
4.2	Regras SIEM	13
4.3	Resultados	14
5	CC DNS	16
5.1	O que é?	16
5.2	Regras SIEM	16
5.3	Resultados	18
6	Comunicações com Países Esquisitos	19
6.1	Regras SIEM	19
6.2	Resultados	21
7	Conclusão	22

Chapter 1

Introdução

Este relatório tem como objectivo analisarmos em detalhe dois ficheiros *data5.parquet* e *test5.parquet* onde no primeiro não foi detetado nenhum comportamento esquisito e no segundo pode haver comportamentos anômalos resultantes de atividades ilícitas dentro da rede.

Posto isto, o objetivo é tentarmos identificar atividades de exfiltração de dados, botnet, CC remoto de dispositivos e comunicações com países esquisitos. Ao longo deste relatório vamos explicar as regras que usámos para identificar as mesmas e explicar também os resultados obtidos para cada uma das situações referidas acima.

Chapter 2

Análise do Dataset Sem Anomalias

Neste capítulo vamos apresentar algumas maneiras que nós usámos para efetuar a análise do dataset sem anomalias e apresentar gráficos com os devidos resultados. **De salientar**, que nos capítulos mais à frente de maneira a detetarmos algumas anomalias também foram feitas algumas análises ao dataset normal como por exemplo, comunicações por ip destino privado.

2.1 Distribuição das portas usadas no DataSet

Numa primeira análise, vemos a distribuição das portas de destino no dataset normal procurando identificar as portas mais usadas para obter uma compreensão dos padrões de uso da porta. A Figura em baixo mostra os resultados. De acordo com os mesmos, é fácil interpretar que a porta 433 é a mais usada o que pode indicar que neste dataset há um elevado número de tráfego por *HTTPS*, como por exemplo vários utilizadores acederem a vários websites que usam o protocolo *HTTPS*.

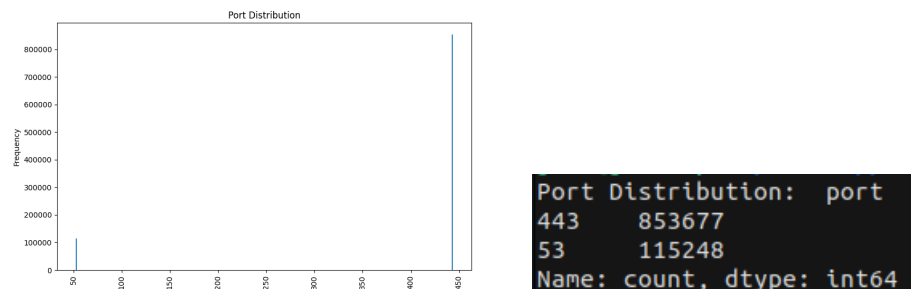


Figure 2.1: Resultados para a porta com mais comunicações.

2.2 Distribuição dos protocolos usados no DataSet

Nesta seção, vamos contar o número de ocorrências de cada protocolo para identificar qual é o mais usado nas comunicações deste dataset.

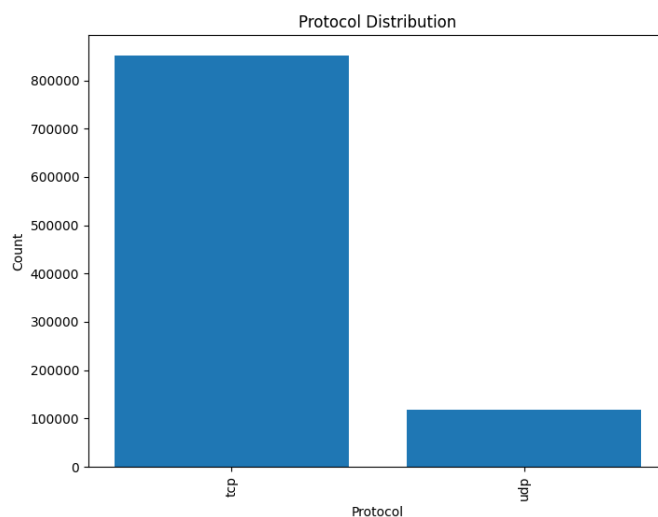


Figure 2.2: Distribuição dos protocolos no dataset.

Tal como é possível observar pela Figura 2.2, é fácil de identificar que o protocolo mais usado é o *tcp*, o que pode indicar, juntamente com o elevado uso da porta 433, que há uma quantidade considerável de comunicação e troca de dados acontecendo em conexões seguras *HTTPS*.

2.3 Identificar o ip de origem com mais comunicações

Em seguida, procurámos por identificar os principais endereços IP de origem que aparecem com mais frequência no dataset. Esta análise, pode ajudar a entender quais endereços IP são responsáveis pela maior parte do tráfego de rede.

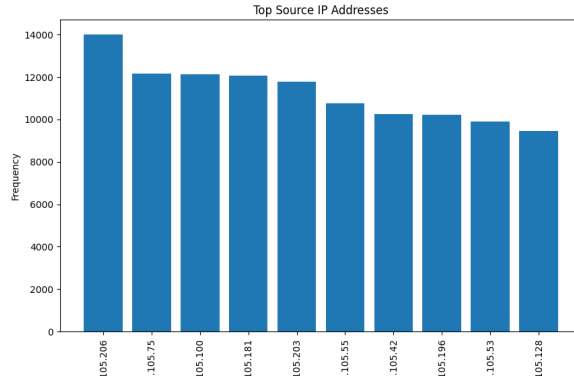


Figure 2.3: Top Source Ip Address.

No gráfico representado pela Figura 3.3, estão presentes os 10 endereços ips com mais frequência nas comunicações. Através do mesmo, é possível identificar que o possível responsável pela maior parte do tráfego de rede seria o ip 192.168.105.206.

2.4 Identificar o ip destino com mais comunicações

Em contrapartida, procurámos identificar os 10 principais endereços IP destino que recebem mais tráfego, o que pode fornecer informações sobre os destinos mais frequentemente contactados.

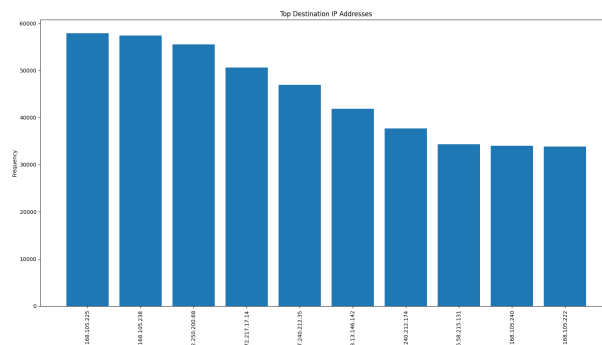


Figure 2.4: Top Destination Ip Address.

Tal como é possível ver pelo gráfico da Figura 2.4, conseguimos identificar que o endereço ip destino mais frequentemente contactado é o 192.168.105.225.

2.5 Tráfego durante diferentes períodos do dia

Por fim, analisámos o volume de tráfego (bytes de upload e download) em diferentes períodos do dia e criámos um gráfico com os resultados de forma a visualizar quais os períodos do dia onde há mais e menos quantidade de tráfego.

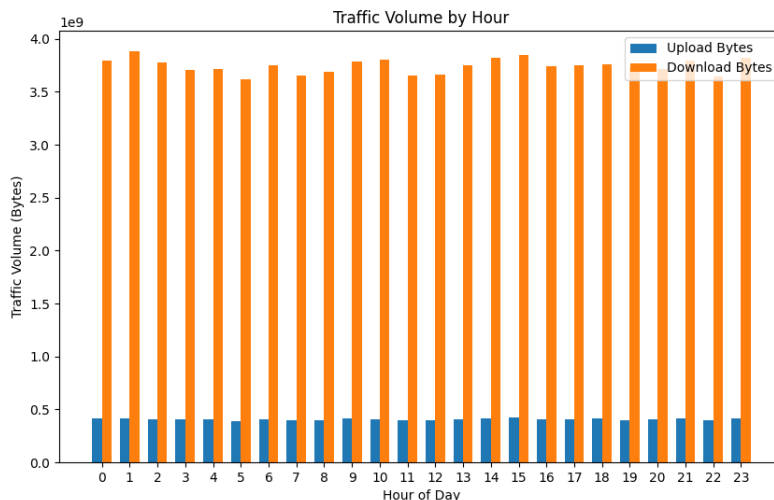


Figure 2.5: Volume de tráfego em vários períodos do dia.

De acordo com o gráfico de cima, é possível verificar que onde houve um menor volume de tráfego foi durante a madrugada, por volta das 5h da manhã e, onde houve maiores quantidades de tráfego foi durante a tarde, por volta das 15h, no final da noite e até no início da madrugada.

Chapter 3

Exfiltração

3.1 O que é?

A exfiltração refere-se à extração ou transferência não autorizada de dados de um ambiente seguro ou controlado para um local externo

3.2 Regras SIEM

3.2.1 Exfiltração por DNS

Em primeiro lugar, vamos procurar detetar exfiltração por **DNS** e, em seguida, detetar exfiltração por **HTTPS**.

Primeiramente, fazemos a leitura dos dois datasets (dataset normal e dataset test), e em seguida, procuramos os ips usados nas comunicações DNS (baseado no protocolo UDP e porta 53) e calculamos para cada um desses ips o número total de bytes download e upload para depois compararmos com o dataset normal com o objetivo de procurarmos por alguma atividade suspeita.

Se para um endereço ip qualquer o número de bytes downloaded/uploaded no dataset test for consideravelmente maior que o número de bytes download/uploaded no dataset normal (para o mesmo ip), consideramos esse ip como um suspeito de exfiltração.

3.2.2 Resultados Exfiltração por DNS

As imagens abaixo mostram o número de bytes downloaded e uploaded para cada ip associado às comunicações/flows DNS (Só aparecem os ips com maior número de bytes downloaded/uploaded).

```
ip downlaod bytes 192.168.105.106 9004248
ip downlaod bytes 192.168.105.116 218474
ip downlaod bytes 192.168.105.141 22348264
ip downlaod bytes 192.168.105.168 176646
```

Figure 3.1: Download Bytes.

```
ip upload bytes 192.168.105.106 3898239
ip upload bytes 192.168.105.116 93699
ip upload bytes 192.168.105.141 9698486
ip upload bytes 192.168.105.168 77288
```

Figure 3.2: Upload Bytes.

Tal como é possível observar pela print do terminal, verificamos que há uma grande discrepância de bytes downloaded/uploaded nos ips 192.168.105.106 e 192.168.105.141. Posto isto, é possível concluir que estes ips representam possíveis anomalias/ataques de exfiltração e devem ser aplicadas as devidas medidas de segurança para prevenir os mesmos.

3.2.3 Exfiltração por HTTPS

Relativamente à exfiltração por **HTTPS**, procuramos os endereços ips nos fluxos *HTTPS* (protocolo TCP e porta 443). Em primeiro lugar, calculamos o número total de uploads para cada ip no dataset test e filtramos apenas os ips cujo valor de upload é superior à média de bytes uploaded. Por fim, são percorridos esses ips e usamos a biblioteca *pygeoip* para obter a informação da geolocalização desses endereços ip.

Depois, é calculado o número total de flows com bytes downloaded superior à média (bytes downloaded). Em seguida vai se filtrar esses resultados de maneira a obter os ips onde a taxa de bytes de download do dataset test é superior a 50% da taxa de bytes download do dataset normal ou os ips que aparecem apenas no dataset com anomalias.

Por fim, para cada um desses ips vamos analisar os padrões de comunicação, analisamos os bytes downloaded nos horários das comunicações que estão no dataset test e não estão no dataset normal apresentando em seguida um gráfico com os resultados.

3.2.4 Resultados Exfiltração por HTTPS

Na Figura abaixo, estão presentes os endereços ip, cuja taxa de bytes upload é superior à média de taxa de bytes uploaded, juntamente com as informações dos países e da organização associados aos mesmos.

```

Source IP: 192.168.105.118, Destination IP: 104.244.42.193
US AS13414 TWITTER
Source IP: 192.168.105.182, Destination IP: 104.244.42.129
US AS13414 TWITTER
Source IP: 192.168.105.183, Destination IP: 104.244.42.1
US AS13414 TWITTER
Source IP: 192.168.105.188, Destination IP: 13.107.42.32
US AS8068 MICROSOFT-CORP-MSN-AS-BLOCK
Source IP: 192.168.105.20, Destination IP: 13.107.42.48
US AS8068 MICROSOFT-CORP-MSN-AS-BLOCK

```

Figure 3.3: Informação dos ips com upload maior que a média.

De acordo com a imagem acima, é possível verificar que há exfiltração por parte de serviços legítimos como é o exemplo do twitter e da Microsoft. Estes resultados mostram que é possível haver exfiltração por meio de serviços considerados legítimos.

Em seguida, tal como foi dito anteriormente, vai-se calcular o número total de flows com bytes donwloaded superior à media e filtra-mos o resultado de maneira a obter os ips que aparecem apenas no dataset com anomalias ou os ips com taxa de bytes downloaded do dataset test superior a 50% da taxa de bytes downloaded do dataset normal. Por fim, pegamos nesses ips e vamos examinar os bytes downloaded nas horas que aparecem apenas no dataset test.

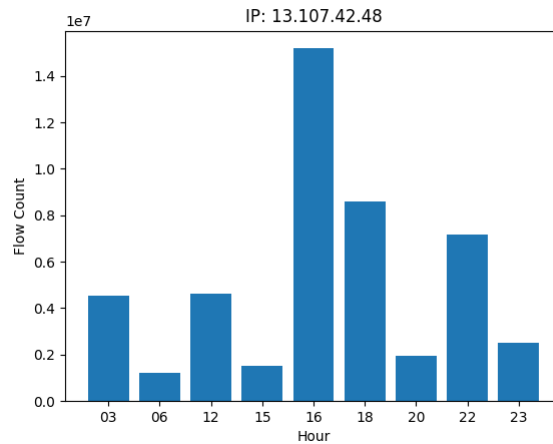


Figure 3.4: Padrão de comunicação para o ip 13.107.42.48.

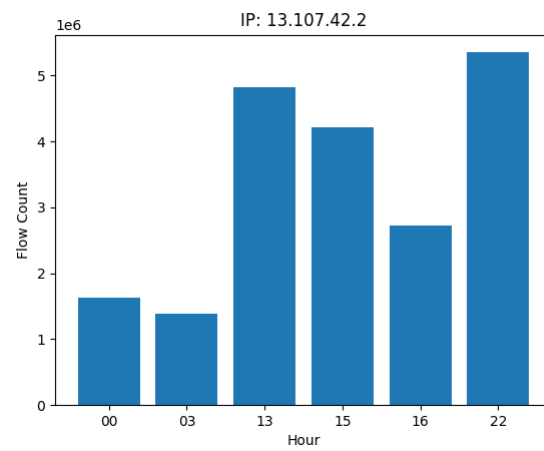


Figure 3.5: Padrão de comunicação para o ip 13.107.42.2.

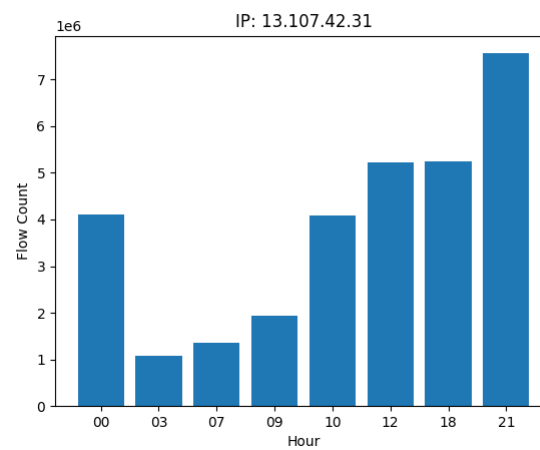


Figure 3.6: Padrão de comunicação para o ip 13.107.42.31.

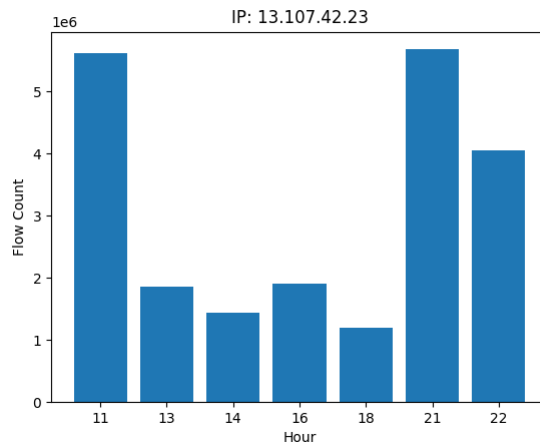


Figure 3.7: Padrão de comunicação para o ip 13.107.32.23.

Com base nos resultados, é possível verificar que as anomalias observadas não estão apenas relacionadas com a taxa de bytes downloaded/uploaded mas também estão relacionadas com os horários das comunicações entre os diferentes ips.

3.2.5 Outros meios de detecção de Exfiltração

A regra que definimos deteta possíveis tentativas de exfiltração analisando o tráfego de rede e identifica instâncias em que a proporção de bytes de download para upload excede um determinado limite.

Em primeiro lugar, vamos ler os dois ficheiros *parquet* através da biblioteca *pandas*. Em seguida, vamos filtrar os dados em ambos os ficheiros, pois só queremos ips privados e por fim vamos calcular no ficheiro sem anomalias, o ratio de download/upload e depois calculamos essa média.

```
private_data1 = data1[(data1['dst_ip'].apply(lambda x: ipaddress.IPv4Address(x) in NET)) ]
private_data = data[(data['dst_ip'].apply(lambda x: ipaddress.IPv4Address(x) in NET)) ]

download_bytes_ratio_data1 = private_data1['down_bytes'] / private_data1['up_bytes']
avg_ratio_data1 = download_bytes_ratio_data1.mean()
print(avg_ratio_data1)
```

Figure 3.8: Calcular downloads/upload ratio.

Depois, calculamos o número de downloads/uploads no ficheiro com anomalias e em seguida vamos verificar quando esse número é 3 vezes maior que o average downloads/uploads do ficheiro sem anomalias, detetando assim possíveis situações de exfiltração.

```
download_bytes_ratio_data = private_data['down_bytes'] / private_data['up_bytes']

# Verify if download_bytes_ratio_data is 4 times higher than the average
is_ratio_higher = download_bytes_ratio_data > (3 * avg_ratio_data1)

# Filter data where ratio is higher
high_ratio_data = private_data[is_ratio_higher]

# Print the filtered data
print(high_ratio_data)
high_ratio_data.to_csv('high_ratio_data.csv', index=False)
```

Figure 3.9: Comparar o ratio de download/upload com o average ratio.

3.2.6 Resultados

avg ratio 6.031858113001418							
	timestamp	src_ip	dst_ip	proto	port	up_bytes	down_bytes
index							
326220	3044180	192.168.105.176	192.168.105.229	tcp	443	6053	113071
291985	3163932	192.168.105.167	192.168.105.240	tcp	443	10205	187984
190254	3351739	192.168.105.150	192.168.105.240	tcp	443	19725	372284
378794	3353254	192.168.105.16	192.168.105.229	tcp	443	920	16908
421530	3664677	192.168.105.120	192.168.105.240	tcp	443	3546	69213
89768	4097551	192.168.105.177	192.168.105.229	tcp	443	10049	204317
178172	4742794	192.168.105.64	192.168.105.223	tcp	443	3903	75842
94336	5049141	192.168.105.158	192.168.105.222	tcp	443	6351	121432
934057	6683100	192.168.105.115	192.168.105.222	tcp	443	5621	102570

Figure 3.10: Resultados exfiltration.

Como se pode ver pela imagem acima, o número de downloads bytes é muito superior ao número de upload bytes o que indica situações de possível exfiltration

Chapter 4

Botnets

4.1 O que é?

Uma *botnet* é uma rede de computadores infectados por malware que estão sob o controle de um único atacante, conhecido como “*bot-herder*”. Cada máquina individual sob o controle do *bot-herder* é conhecida como *bot* e pode ser usada para fins maliciosos uma vez que é controlada pelo atacante.

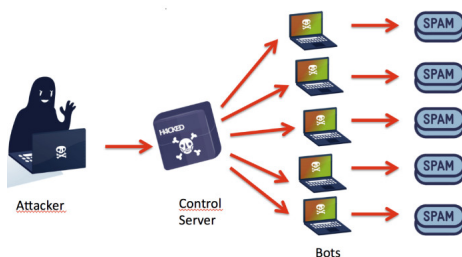


Figure 4.1: Botnets.

4.2 Regras SIEM

Posto isto, de maneira a detetar-mos este tipo de ataques, criámos uma regra que identifica endereços IP de origem que exibem padrões de comunicação suspeitos com vários endereços IP de destino privados, indicando um possível comportamento de *botnet*. Além disso, para cada ip de origem suspeito de atividade botnet, foram calculadas as médias de bytes *downloaded* e *uploaded* e comparadas com a média total com o objetivo de procurar por ips com maior atividade *botnet*.

Primeiro, procuramos por endereços ip privados destino no dataset sem anomalias e no dataset *test* com possíveis anomalias, tal como mostra a Figura 4.2, de maneira a procurarmos por comunicações esquisitas, i.e, que fujam à regra, internamente.

```
#Find private ip addresses
NET=ipaddress.IPv4Network('192.168.105.0/24')
bprivate=data_dataset5.apply(lambda x: ipaddress.IPv4Address(x['dst_ip']) in NET,axis=1)
bprivate_test=data_test.apply(lambda x: ipaddress.IPv4Address(x['dst_ip']) in NET,axis=1)

# Find unique destination IP addresses in the test dataset that are private
test_destinations_private = set(data_test[bprivate_test]['dst_ip'])

# Find unique destination IP addresses in the normal dataset that are private
normal_destinations_private = set(data_dataset5[bprivate]['dst_ip'])
```

Figure 4.2: Identificação de ips privados e procura de ips privados no destino.

Em seguida, vamos percorrer cada um desses ip's privados (destino) no dataset *test* e vamos ver todas as comunicações com esse ip, ou seja, procuramos vários *src_ip address* que tenham esse ip como destino e vamos fazer uma comparação com os *src_ip address* para o mesmo endereço ip destino, no dataset **sem anomalias**, de maneira a procurarmos alterações em quem falou com quem e, caso haja alterações, consideramos esses *src_ip address's* como suspeitos de atividade *botnet*. A Figura 4.3 ilustra o procedimento acima explicado.

```
# Initialize a list to store botnet source IP addresses
botnet_sources = []

# Iterate over each private destination IP address in the test dataset
for destination_ip in test_destinations_private:
    # Get the source IP addresses in the test dataset for the current destination IP
    test_sources = set(data_test[data_test['dst_ip'] == destination_ip]['src_ip'])

    # Get the source IP addresses in the normal dataset for the current destination IP
    normal_sources = set(data_dataset5[data_dataset5['dst_ip'] == destination_ip]['src_ip'])

    # Find the source IP addresses that are in the test dataset but not in the normal dataset
    botnet_sources.extend(list(test_sources - normal_sources))

# Remove duplicates from the list of botnet source IP addresses
botnet_sources = list(set(botnet_sources))

print("Botnet Sources: ", botnet_sources)
```

Figure 4.3: Procedimento para detetar alterações em quem falou com quem.

4.3 Resultados

Após correr este script, obtivemos os resultados evidenciados na Figura 4.4.

```
Botnet Sources: ['192.168.105.10', '192.168.105.108', '192.168.105.116', '192.168.105.151', '192.168.105.73', '192.168.105.19', '192.168.105.88', '192.168.105.172', '192.168.105.72']
```

Figure 4.4: Resultados.

Estes resultados mostram possíveis endereços ips com atividade *botnet* uma vez que nestes ips foram encontradas alterações em quem falou com quem com-

parando os dois dataset, o sem anamolias e o com possíveis anomalias. Além disso, é possível ver pela Figura 4.5 que os ips suspeitos comunicam entre si, o que é mais um indício de que os mesmos podem pertencer a uma rede botnet.

```

Destination IP: 192.168.105.19
test_sources: {'192.168.105.151', '192.168.105.172', '192.168.105.88'}
Destination IP: 192.168.105.19
normal_sources: set()
Destination IP: 192.168.105.172
test_sources: {'192.168.105.19', '192.168.105.151', '192.168.105.88'}
Destination IP: 192.168.105.172
normal_sources: set()
Destination IP: 192.168.105.88
test_sources: {'192.168.105.19', '192.168.105.172', '192.168.105.151'}
Destination IP: 192.168.105.88
normal_sources: set()

```

Figure 4.5: Possível rede Botnet.

Por fim, de maneira a detalhar mais as situações onde pode haver mais atividade botnet, percorremos a botnet_sources (Figura 4.4) e calculamos a média de bytes *downloaded* e *uploaded* para cada ip e comparámos com a média total de bytes *downloaded* e *uploaded* do dataset normal. A Figura 4.6 mostra os resultados obtidos.

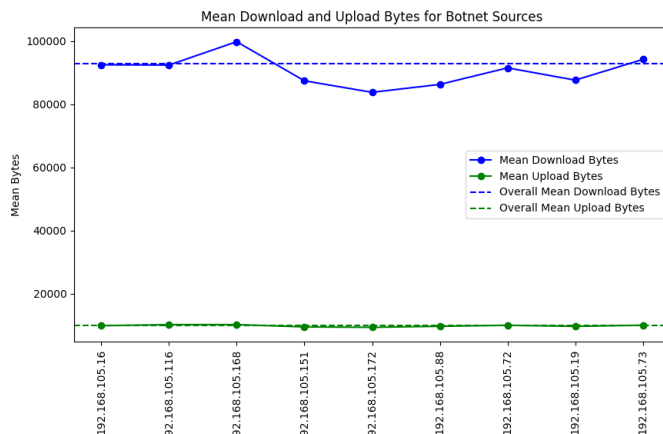


Figure 4.6: Gráfico.

É possível verificar que alguns ips "fogem" à média total de bytes, como por exemplo o ip 192.168.105.168. Nesta situação, poderíamos, por exemplo, deduzir que onde havia mais atividade *botnet* seria na máquina com o endereço ip 192.168.105.168.

Posto isto, tendo detetado a atividade suspeita, uma maneira de a eliminar seria ir a cada máquina com os endereços ip's suspeitos de atividade *botnet* e eliminar o *malware* presente na mesma.

Chapter 5

CC DNS

Neste capítulo vamos explicar a maneira que nós abordamos para detetar CC e respetivos resultados.

5.1 O que é?

CC refere-se à capacidade de controlar dispositivos eletrônicos à distância, por meio de uma conexão remota.

5.2 Regras SIEM

A regra SIEM foi projetada para detectar tentativas de comunicação remota de Comando e Controle (CC) em uma rede. Esta regra utiliza dados de tráfego de rede e concentra-se na identificação de conexões de endereços IP de origem para endereços IP de destino na porta 53 (DNS). A regra identifica em cada um dos *parquets* os ips privados e guarda-os em dois *dataframes* diferentes. A regra então conta o número de conexões de cada ip origem e por fim compara ambos os *dataframes* com o número de conexões e verifica se houve um aumento ou decrémento exponencial de conexões.

Em primeiro lugar, vamos ler os dois ficheiros *parquet* através da biblioteca *pandas*. Em seguida, vamos filtrar os dados em ambos os ficheiros, pois só queremos ips privados com flows na porta 53(DNS), e depois contamos o número de conexões de cada ip origem. O *dataframe* `connectioncountsdata1` é o número de conexões, de cada ip, do *parquet* sem anomalias e o outro do *parquet* com anomalias.

```
# Filter the data based on private addresses and port 53
private_data1 = data1[(data1['src_ip'].apply(lambda x: ipaddress.IPv4Address(x) in NET)) & (data1['port'] == 53)]
private_data = data[(data['src_ip'].apply(lambda x: ipaddress.IPv4Address(x) in NET)) & (data['port'] == 53)]

# Count the number of connections from each source IP to private destination IPs
connection_counts_data1 = private_data1['src_ip'].value_counts().reset_index()
connection_counts_data1.columns = ['src_ip', 'connection_count_data1']
connection_counts_data1.to_csv('connection_counts_data1.csv', index=False)

connection_counts_data = private_data['src_ip'].value_counts().reset_index()
connection_counts_data.columns = ['src_ip', 'connection_count_data']
connection_counts_data.to_csv('connection_counts_data.csv', index=False)
```

Figure 5.1: Cálculo do número de conexões de cada srcip privado na porta 53.

Em seguida, estamos a fazer uma comparação e ver quais ips tiveram um aumento ou decrémento exponencial de conexões, neste caso 4x mais ou 4x menos. Case exista, então é detetado CC.

```
# Merge the connection counts from both dataframes based on the source IP
merged_data = pd.merge(connection_counts_data1, connection_counts_data, on='src_ip', how='inner')
merged_data.to_csv('merged_data.csv', index=False)

# Filter the merged data to keep only the IP addresses with connection counts four times higher
filtered_data = merged_data[merged_data['connection_count_data'] > (4 * merged_data['connection_count_data1'])]

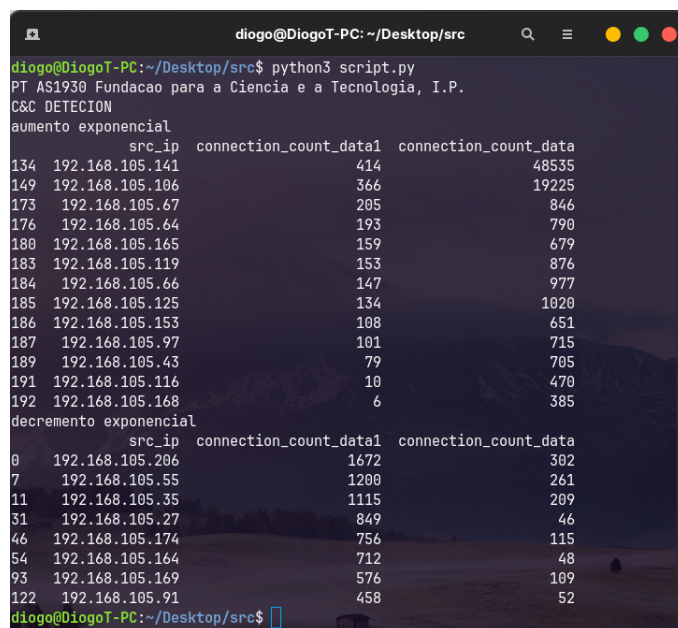
print("C&C DETECTION")
filtered_data.to_csv('aumento.csv', index=False)
print("aumento exponencial")
print(filtered_data)

# Filter the merged data to keep only the IP addresses with connection counts four times higher
filtered_data2 = merged_data[merged_data['connection_count_data'] < (0.25 * merged_data['connection_count_data1'])]

filtered_data2.to_csv('decremento.csv', index=False)
print("decremento exponencial")
print(filtered_data2)
```

Figure 5.2: Verificação se houve aumento/decremento exponencial de número de flows.

5.3 Resultados



```
diogo@DiogoT-PC: ~/Desktop/src
diogo@DiogoT-PC:~/Desktop/src$ python3 script.py
PT AS1930 Fundacao para a Ciencia e a Tecnologia, I.P.
C&C DETECTION
aumento exponencial
src_ip connection_count_data1 connection_count_data
134 192.168.105.141 414 48535
149 192.168.105.106 366 19225
173 192.168.105.67 205 846
176 192.168.105.64 193 790
180 192.168.105.165 159 679
183 192.168.105.119 153 876
184 192.168.105.66 147 977
185 192.168.105.125 134 1020
186 192.168.105.153 108 651
187 192.168.105.97 101 715
189 192.168.105.43 79 705
191 192.168.105.116 10 470
192 192.168.105.168 6 385
decremento exponencial
src_ip connection_count_data1 connection_count_data
0 192.168.105.206 1672 302
7 192.168.105.55 1200 261
11 192.168.105.35 1115 209
31 192.168.105.27 849 46
46 192.168.105.174 756 115
54 192.168.105.164 712 48
93 192.168.105.169 576 109
122 192.168.105.91 458 52
diogo@DiogoT-PC:~/Desktop/src$
```

Figure 5.3: Resultados.

De acordo com a print, é possível observar os ips que tiveram um aumento, ou decremento exponencial, no número de flows, detetando assim possíveis situações de CC.

Chapter 6

Comunicações com Países Esquisitos

Neste capítulo vamos explicar a maneira que nós abordamos para detetar comunicações com países esquisitos e respetivos resultados.

6.1 Regras SIEM

Esta regra SIEM é projetada para detetar a comunicação com países considerados "esquisitos" com base nos dados de tráfego de rede (bytes *downloaded/uploaded*).

Ela identifica endereços IP de destino associados a endereços IPv4 públicos e localiza-os geograficamente para determinar o seu país de origem. A regra então compara esses países nos dois datasets, de maneira a procurar por países esquisitos que não pertençam ao dataset normal e pertençam ao dataset *test* e, por fim, compara a comunicação desses países incomuns com o tráfego geral da rede para identificar possíveis anomalias e riscos de segurança.

Em primeiro lugar, vamos ler os dois ficheiros *parquet* através da biblioteca *pandas*. Em seguida, calculamos a média de bytes *downloaded* e *upload* do dataset sem anomalias e procuramos no dataset sem anomalias e no dataset com possíveis anomalias os endereços públicos e guardamos o resultado nas variáveis *bpublic* e *bpublic_test*, respetivamente. Posto isto, vamos buscar o *country code* para cada endereço ip público, em ambos os datasets (normal e com possíveis anomalias), através das variáveis *bpublic* e *bpublic_test*, tal como mostra a Figura abaixo.

```

#Calculate average of download and upload bytes
avg_down_bytes_total = data_dataset5['down_bytes'].mean()
avg_up_bytes_total = data_dataset5['up_bytes'].mean()

#Is destination IPv4 a public address?
NET=ipaddress.IPv4Network('192.168.105.0/24') #192.168.0.0/16
bpublic=data_dataset5.apply(lambda x: ipaddress.IPv4Address(x['dst_ip']) not in NET,axis=1)
#print(data_dataset5[bpublic]['dst_ip'])

bpublic_test=data_test.apply(lambda x: ipaddress.IPv4Address(x['dst_ip']) not in NET,axis=1)

#Geolocalization of public destination address
cc=data_dataset5[bpublic]['dst_ip'].apply(lambda y:gi.country_code_by_addr(y)).to_frame(name='cc')
cc_array = cc['cc'].array.unique()
print("CC NORMAL",cc_array)

cc_test = data_test[bpublic_test]['dst_ip'].apply(lambda y:gi.country_code_by_addr(y)).to_frame(name='cc')
cc_array_test = cc_test['cc'].array.unique()
print("CC WITH ANOMALIES",cc_array_test)

```

Figure 6.1: Cálculo das médias, descoberta de ips públicos e country codes.

Por fim, comparámos os *country codes* presentes no dataset normal e no dataset com possíveis anomalias de maneira a obtermos os *country codes* dos países que estão presentes no dataset com possíveis anomalias e não estão presentes no dataset sem anomalias, este resultado indica que esses países podem ser possíveis suspeitos.

De maneira a aprofundar mais a situação, fomos calcular a média de bytes *download* e *uploaded* para cada um desses países suspeitos e comparámos, através de um gráfico, essa média de bytes com a média total de bytes *downloaded* e *upload* do dataset sem anomalias com o objetivo de procurarmos por irregularidades nesses países considerados suspeitos. A figura abaixo mostra como foi feito este processo acima referido.

```

avg_down_bytes_cc = []
avg_up_bytes_cc = []
unusual_countries = []
for cc in cc_array_test:
    if cc not in cc_array:
        # Filter the data_test for the current country code to get the information of it
        cc_data = data_test[data_test['dst_ip'].apply(lambda ip: gi.country_code_by_addr(ip)) == cc]
        #print(cc_data)

        # Calculate the average downloaded bytes and uploaded bytes for this country
        avg_down_bytes_cc.append(cc_data['down_bytes'].mean())
        avg_up_bytes_cc.append(cc_data['up_bytes'].mean())

        unusual_countries.append(cc)

print("Unusual Countries: ", unusual_countries)

```

Figure 6.2: Cálculo dos bytes para cada país suspeito.

Na criação do gráfico, pegamos nos valores calculados acima e mostrámo-los no gráfico de maneira a ser mais fácil visualizar os resultados e identificar os países mais esquisitos.

6.2 Resultados

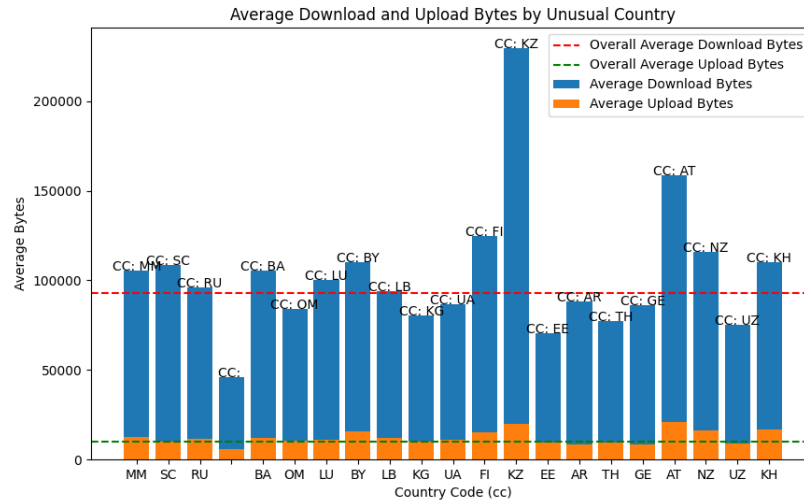


Figure 6.3: Gráfico resultante.

De acordo com o gráfico, é possível verificar algumas irregularidades. O cc *KZ* tem uma média de bytes *downloaded* extremamente alta comparado com a média total de bytes *downloaded* tal como a *FI*, *AT*, *NZ*, *KH*. Além disso, também podemos ver alguns países que têm menos bytes *downloaded* em relação à média, como é o exemplo do *EE*, onde podemos assumir que também é suspeito.

Desta forma, estas discrepâncias podem indicar comunicações com países esquisitos e podem ser aplicadas as devidas medidas de segurança.

Chapter 7

Conclusão

Em suma, achamos que conseguimos atingir o objetivo deste projeto e, além disso, ganhámos algum conhecimento sobre cada um destes métodos de ataque e maneiras de os detetar e posteriormente prevenir os mesmos. Sem dúvida que o conhecimento adquirido no desenvolvimento deste projeto vai ser importante no nosso futuro.