Universidade do Minho

Mestrado em Engenharia Informática

Mineração de dados

Network Intrusion Detection

Grupo 6

Ana Murta (PG50184)
Hugo Gomes (PG51242)
Manuel Novais (PG50575)

junho, 2023

Contents

1 Introdução

1.1 Contextualização

Atualmente, a deteção de intrusões na rede é um tema de extrema importância e relevância . Com o aumento dos ataques cibernéticos ao longo dos anos, é essencial que as organizações disponham de uma maneira eficaz de identificar e responder a atividades maliciosas nas suas redes. A deteção destas envolve a monitorização e análise contínua do tráfego de rede para, desse modo, identificar qualquer atividade suspeita ou potencialmente prejudicial. Isto pode incluir tentativas de invasão, exploração de vulnerabilidades, presença de *malware*, entre outros.

O avanço da tecnologia não proporcionou apenas melhores soluções de deteção de ataques na rede, também possibilitou o desenvolvimento de ataques mais sofisticados, que nem sempre são detetáveis. Deste modo, de forma a identificar ataques desconhecidos, as soluções de deteção precisam de evoluir constantemente, incluindo novas técnicas de análise comportamental, aprendizagem automática e inteligência artificial.

Além de reconhecer e responder a ciberataques em tempo real, a deteção de intrusões na rede desempenha igualmente um papel importante na investigação forense de incidentes de segurança. Por outras palavras, os registos e *logs* gerados pelos sistemas de deteção podem fornecer várias informações pertinentes como as origens dos ataques, as técnicas utilizadas ou eventuais vulnerabilidades exploradas.

Em conclusão, os ataques cibernéticos são tentativas indesejadas de roubar, danificar, alterar ou interromper dados e sistemas. Num mundo cada vez mais digital, é essencial priorizar a cibersegurança para proteger os ativos, a reputação e a privacidade das organizações e indivíduos, bem como garantir a continuidade das operações de negócio e a segurança da infraestrutura. Embora os programas de cibersegurança não consigam garantir proteção absoluta, é importante procurar resultados adequados recursos e tolerância ao risco da organização.

1.2 Motivação e Objetivos

A crescente incidência global de ataques cibernéticos nos últimos anos, constituí uma das principais preocupações para indivíduos e organizações no que diz respeito à sua segurança cibernética. Além disso, nos dias de hoje, a sociedade encontra-se profundamente dependente da tecnologia, o que realça a relevância da segurança cibernética como um tema de extrema importância. Também a interdependência entre a população e a tecnologia ressalta a necessidade de proteger os dados pessoais e informações confidenciais contra potenciais ameaças.

Para além disto, o crescente número de pessoas trabalhando remotamente, especialmente após o período pós-pandemia, resultou numa utilização mais ampla de dispositivos pessoais para fins profissionais. Tal contexto intensificou a preocupação com a privacidade dos dados, tornando ainda mais imperativo a adoção de medidas efetivas de segurança cibernética.

Deste modo, o estudo de deteção de ataques em redes visa compreender a natureza e os mecanismos dos ataques cibernéticos, bem como identificar métodos de deteção e prevenção. Este tem como objetivo salvaguardar redes e dispositivos contra possíveis ameaças, por meio da identificação de padrões maliciosos de tráfego de rede que possam indicar atividades realizadas por *hackers* ou *malwares*. De forma a permitir a ser possível antecipar-se ataques cibernéticos e mitigar eventuais danos. Durante esta análise de dados para a deteção de anomalias, está também empregado a procura por comportamentos suspeitos por parte de utilizadores mal intencionados e identificação de fontes de ameaças, dado que tais abordagens possibilitam a deteção de atividades duvidosas, como tentativas de acesso não autorizado a recursos restritos ou a transferência de informações confidenciais para fora da rede. Desta forma, medidas adequadas poderão ser tomadas para evitar danos e garantir a segurança dos sistemas.

Posto isto, o objetivo principal deste trabalho é o desenvolver um sistema de deteção de ataques em redes capaz de categorizar cada conexão como normal ou anomalia. De modo a alcançar este, serão realizados estudos e implementações de técnicas e algoritmos de deteção de intrusões, bem como a análise de dados de tráfego de rede, com o intuito de identificar padrões maliciosos e comportamentos suspeitos. Este estudo tem como propósito contribuir para a proteção da segurança cibernética ao

desenvolver um sistema de deteção de intrusões em redes, proporcionando uma defesa eficaz contra ameaças cibernéticas e assegurando a integridade e confidencialidade dos dados.

2 Fontes de dados

No trabalho realizado foram utilizadas duas fontes de dados: *NF-UQ-NIDS-v2 Network Intrusion Detection Dataset*, *SIMARGL2021 Network Intrusion Detection Dataset*. Estas foram recolhidas da plataforma kaggle.

2.1 NF-UQ-NIDS-v2

Esta fonte de dados representa os benefícios de conjuntos de atributos de *datasets* partilhados, onde a fusão de vários conjuntos mais pequenos é possível. Eventualmente, isto levará a conjuntos de dados NIDS maiores e mais universais, contendo fluxos de múltiplas configurações de rede e diferentes definições de ataque.

Para além disto, este contém uma *feature label* adicional que identifica o conjunto de dados original de cada fluxo, sendo que isto pode ser utilizado para comparar os mesmos cenários de ataque realizados em duas ou mais redes de teste diferentes. Neste *dataset* também foram modificadas as categorias de ataque para combinar todas as categorias pai. Nesta fonte de dados, os ataques denominados ataques DoS-Hulk, ataques DoS-SlowHTTPTest, ataques DoS-GoldenEye e ataques DoS-Slowloris foram renomeados para a categoria pai DoS. Os ataques denominados ataque DDOS-LOIC-UDP, ataque DDOS-HOIC e ataques DDoS-LOIC-HTTP foram renomeados para DDoS. Os ataques denominados FTP-BruteForce, SSH-Bruteforce, Brute Force -Web e Brute Force -XSS foram combinados como uma categoria de *brute-force*. Por fim, os ataques de Injeção SQL foram incluídos na categoria de ataques de injeção. Por último, o *dataset* NF-UQ-NIDS contém um total de 11.994.893 registros, dos quais 9.208.048 (76,77%) são fluxos benignos e 2.786.845 (23,23%) são ataques.

O anexo A contém a lista das *features* presentes no conjunto de dados em questão, enunciando resumidamente o significado de cada um, assim como o tipo de dados usados para a sua representação.

2.2 SIMARGL2021

Este *dataset* apresenta os efeitos da utilização de métodos de deteção de intrusão, baseados em *machine learning*, no tráfego de rede de uma arquitetura real. Aliás, a contribuição principal deste trabalho é um conjunto de dados proveniente de uma rede académica do mundo real. Inicialmente, foi recolhido tráfego da vida real e, após a realização de uma série de ataques, foi construído um conjunto de dados . O esquema de dados de rede está no formato Netflow v9 e, contém 44 *features* únicas e uma *label* que descreve cada *frame*. Este é financiado pelo Projeto SIMARGL - *Secure Intelligent Methods for Advanced RecoGnition of malware and stegomalwar*e, com o apoio da Comissão Europeia e do Programa Horizon 2020, sob o Contrato de Subvenção No. 833042.

O anexo B contém a lista das *features* presentes no conjunto de dados em questão, enunciando resumidamente o significado de cada um, assim como o tipo de dados usados para a sua representação.

3 Visualização e Exploração de dados

O trabalho com o *dataset NF-UQ-NIDS-v2* segue-se com a elaboração de um estudo geral do estado inicial dos dados, de modo a determinar o tratamento necessário para a utilização dos mesmos na criação de modelos de aprendizagem automática.

A exploração estatística efetuada permitiu conhecer, de um modo geral, o conteúdo das *features* que integram o conjunto de dados anteriormente apresentado. Inicialmente foi efetuada uma exploração relativamente às *features* de *dtype objetc* através do uso da função `describe(include=[object])`. A Figura 1 apresenta o *output* gerado por esta, que permitiu concluir que um único neste *dataset* existe 21 ataques únicos, sendo o mais frequente o *Benign*.

A Figura 2 apresenta o *output* da função `describe()` da biblioteca *Pandas*, para os dados numéricos, o que permitiu obter informação estatísticas acerca dos mesmos, desde valores extremos das diversas *features* numéricas, da sua média e desvio padrão. Relativamente aos *missing values*, valores

	IPV4_SRC_ADDR	IPV4_DST_ADDR	Attack	L7_PROTO_NAME	PROTOCOL_MAP
count	3000000	3000000	3000000	2270817	3000000
unique	37465	11937	21	58	6
top	192.168.100.148	192.168.100.3	Benign	CBT	tcp
freq	421688	565344	992765	992316	1846223

Figure 1: Dados sobre as *object features*.

que devem ser tratados aquando da fase de pré-processamento, verificou-se que apenas a *feature L7_PROTO_NAME* apresentava *missing values*.

	L4_SRC_PORT	L4_DST_PORT	PROTOCOL	IN_BYTES	IN_PKTS	OUT_BYTES	OUT_PKTS	TCP_FLAGS	FLOW_DURATION_MILLISECONDS	MIN_IP_PKT_LEN	I
count	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	
mean	4.038722e+04	3.350640e+03	1.024170e+01	9.915695e+02	9.670526e+00	3.034575e+03	4.649585e+00	2.418990e+01	2.315053e+06	2.356534e+01	
std	1.842122e+04	1.068101e+04	5.654876e+00	8.326632e+04	5.763028e+02	2.536018e+05	1.928144e+02	5.845195e+01	2.140399e+06	2.696914e+01	
min	0.000000e+00	0.000000e+00	0.000000e+00	3.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	
25%	2.724000e+04	8.000000e+01	6.000000e+00	5.600000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	
50%	4.573000e+04	8.000000e+01	6.000000e+00	1.120000e+02	2.000000e+00	0.000000e+00	0.000000e+00	2.000000e+00	4.293061e+06	0.000000e+00	
75%	5.504500e+04	4.430000e+02	1.700000e+01	2.800000e+02	3.000000e+00	1.520000e+02	1.000000e+00	2.200000e+01	4.294029e+06	4.000000e+01	
max	6.553500e+04	6.553500e+04	2.550000e+02	7.405536e+07	1.926100e+05	1.554980e+08	1.045650e+05	2.230000e+02	4.294967e+06	5.470000e+02	

MAX_IP_PKT_LEN	SRC_TO_DST_SECOND_BYTES	DST_TO_SRC_SECOND_BYTES	RETRANSMITTED_IN_BYTES	RETRANSMITTED_IN_PKTS	RETRANSMITTED_OUT_BYTES
3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06
2.649258e+02	2.196007e+298	1.386680e+151	8.017808e+01	3.958197e-01	5.691843e+02
4.320319e+02	inf	inf	6.931635e+03	1.240083e+01	1.408053e+04
2.800000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
4.000000e+01	6.800000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
1.000000e+02	1.408000e+03	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
1.400000e+02	3.806000e+03	1.640000e+02	0.000000e+00	0.000000e+00	0.000000e+00
6.521200e+04	6.588022e+304	4.160041e+157	6.321251e+06	1.272900e+04	3.953335e+06

RETRANSMITTED_OUT_PKTS	TCP_WIN_MAX_IN	TCP_WIN_MAX_OUT	Label
3.000000e+06	3.000000e+06	3.000000e+06	3.000000e+06
5.734313e-01	6.023915e+03	8.379874e+03	6.690783e-01
1.007526e+01	1.324776e+04	1.769573e+04	4.705450e-01
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
0.000000e+00	5.120000e+02	0.000000e+00	1.000000e+00
0.000000e+00	4.096000e+03	0.000000e+00	1.000000e+00
2.909000e+03	6.553500e+04	6.553500e+04	1.000000e+00

Figure 2: Dados estatísticos relativamente aos valores numéricos.

A Figura 3 apresenta uma representação gráfica da distribuição das *features* categóricas através de gráficos de circulares. A partir destes conseguimos observar e reconhecer tendências nos dados, concluindo que nenhuma das *features* está uniformemente distribuída. Aliás, devido à grande variedade de valores únicos em cada uma destas *features*, os valores com menos de 2% foram agrupados, sendo estes os conjunto *Others*.

Em relação à distribuição dos dados das *features* numéricas, como podemos ver pela Figura 4, algumas das *features* apresentam uma faixa de valores ampla. Contudo, existe uma concentração de valores em torno de determinados valores.

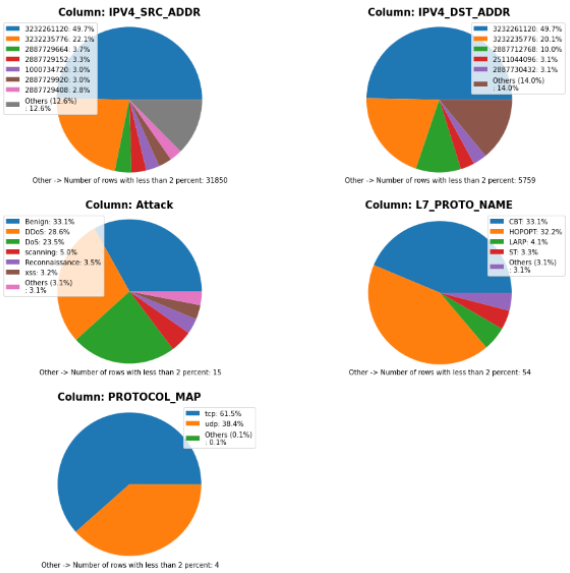

Figure 3: Distribuição dos dados das *features* categóricas.

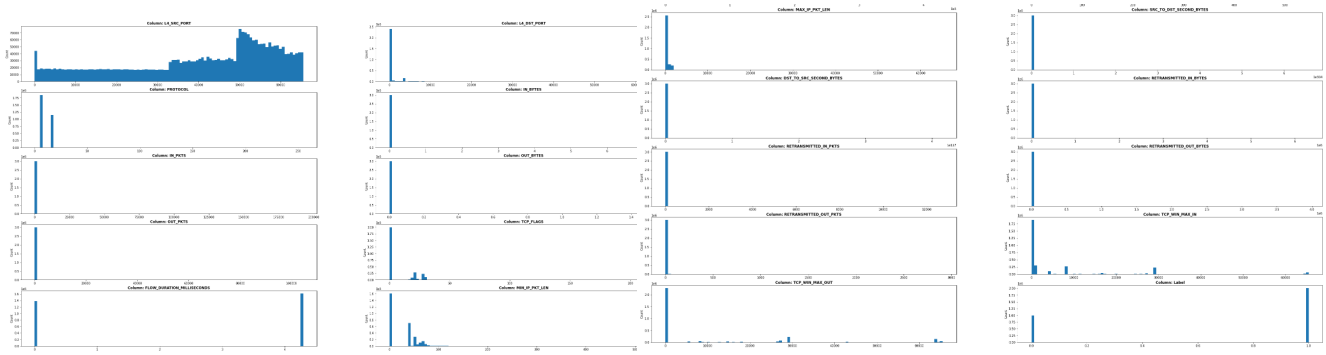

Figure 4: Distribuição dos dados das *features* numéricas.

A Figura 5 oferece uma representação da dispersão dos dados das *features* numéricas. Na observação destas, foi aplicado a estratégia de remover os *outliers* para obter uma melhor visualização dos gráficos.

Por último, para este *dataset* também foi criado um gráfico (figura 6) de dispersão onde os pontos são posicionados nas coordenadas definidas pelas *features L4_SRC_PORT* e *L4_DST_PORT*. A cor dos pontos é determinada pela *feature Attack*, possibilitando visualizar a relação entre as portas de origem e destino e se um ataque ocorreu ou não.

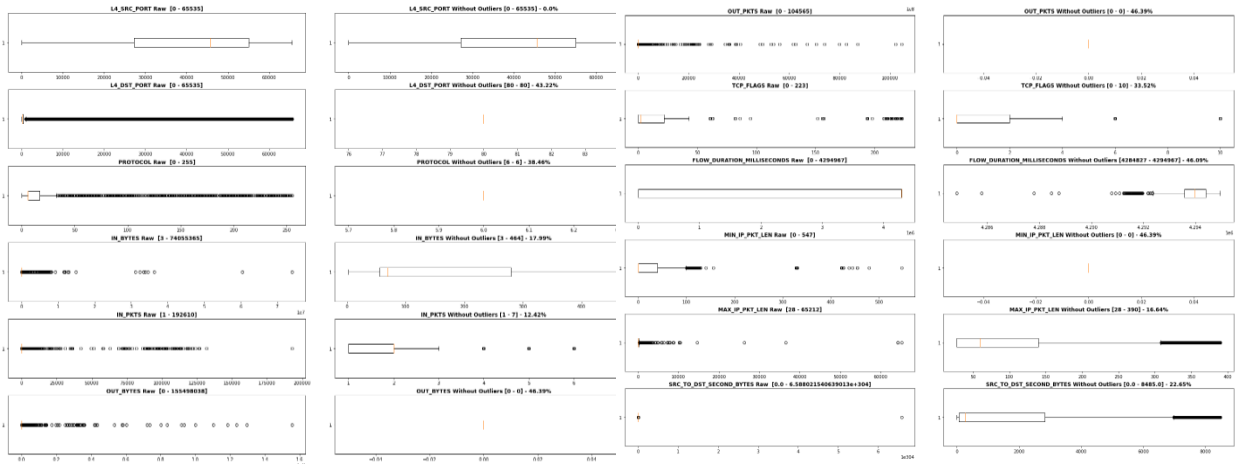

Figure 5: Dispersão dos dados das *features* numéricas.

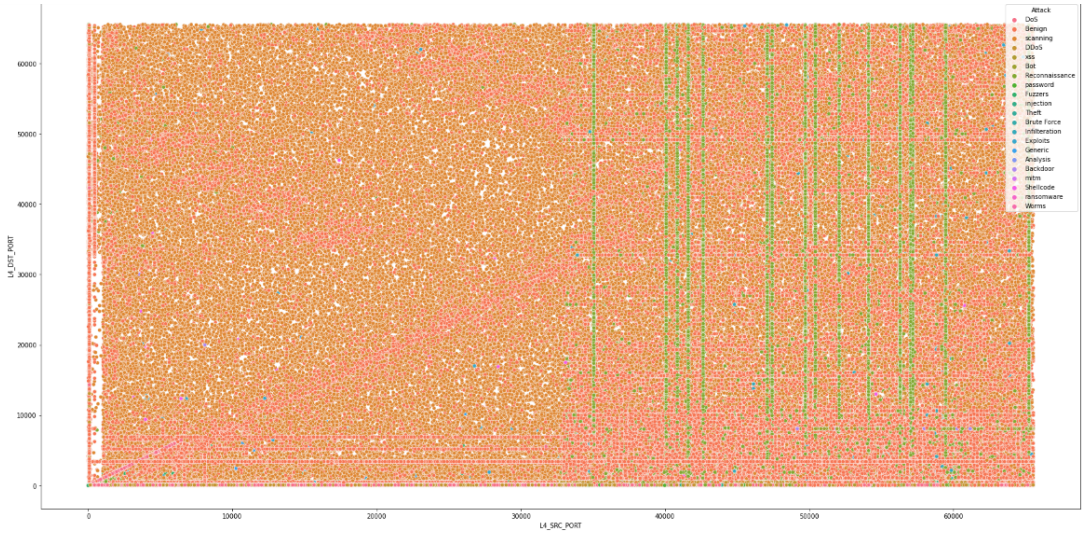

Figure 6: Relação entre as portas de origem e destino e se um ataque ocorreu ou não

Relativamente ao trabalho com o *dataset SIMARGL2021*, em seguida está apresentada a elaboração de um estudo geral do estado inicial dos dados, de modo a determinar o tratamento necessário para a utilização dos mesmos na criação de modelos de aprendizagem automática.

Inicialmente foi efetuada uma exploração relativamente às *features* de *dtype objetc* através do uso da função `describe(include=[object])`. A Figura 7 apresenta o *output* gerado por esta, que permitiu concluir que um único neste *dataset* existe 4 ataques únicos, sendo o mais frequente o *SYN Scan - aggressive*.

	DST_TO_SRC_SECOND_BYTES	IPV4_DST_ADDR	IPV4_SRC_ADDR	PROTOCOL_MAP	SRC_TO_DST_SECOND_BYTES	L7_PROTO_NAME	Attack
count	4901358	4901358	4901358	4901358	4901358	4901358	4901358
unique	220714	151868	38982	5	232608	411	4
top	40	10.114.226.5	10.114.241.166	tcp	44	Unknown	SYN Scan - aggressive
freq	2168095	434478	2512625	3625011	2524868	2950884	2496814

Figure 7: Dados sobre as *object features*.

A Figura 8 apresenta o *output* da função `describe()` da biblioteca *Pandas*, para os dados numéricos, o que permitiu obter informação estatísticas acerca dos mesmos, desde valores extremos das diversas *features* numéricas, da sua média e desvio padrão.

	FLOW_DURATION_MILLISECONDS	IN_BYTES	IN_PKTS	L4_DST_PORT	L4_SRC_PORT	MAX_IP_PKT_LEN	MIN_IP_PKT_LEN	OUT_BYTES	OUT_PKTS	PROTOCOL
count	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4901358.0	4901358.0	4.901358e+06	4.901358e+06	4.901358e+06
mean	2.274823e+03	6.826010e+03	1.474963e+01	2.062448e+04	4.534470e+04	0.0	0.0	1.986429e+04	1.785704e+01	8.326119e+00
std	1.283415e+04	2.004935e+06	1.555223e+03	2.192566e+04	1.477386e+04	0.0	0.0	1.705709e+06	1.276004e+03	4.786715e+00
min	0.000000e+00	2.800000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.0	0.0	0.000000e+00	0.000000e+00	1.000000e+00
25%	0.000000e+00	4.400000e+01	1.000000e+00	4.430000e+02	4.895800e+04	0.0	0.0	4.000000e+01	1.000000e+00	6.000000e+00
50%	0.000000e+00	4.400000e+01	1.000000e+00	1.068100e+04	4.947000e+04	0.0	0.0	4.000000e+01	1.000000e+00	6.000000e+00
75%	2.600000e+01	9.800000e+01	1.000000e+00	4.008800e+04	4.998300e+04	0.0	0.0	1.230000e+02	1.000000e+00	6.000000e+00
max	1.199990e+05	2.914678e+09	2.087191e+06	6.553500e+04	6.553500e+04	0.0	0.0	2.277959e+09	1.528392e+06	5.800000e+01

RETRANSMITTED_IN_BYTES	RETRANSMITTED_IN_PKTS	RETRANSMITTED_OUT_BYTES	RETRANSMITTED_OUT_PKTS	TCP_FLAGS	TCP_WIN_MAX_IN	TCP_WIN_MAX_OUT	Label
4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06	4.901358e+06
1.367366e+02	1.297324e-01	1.580767e+03	1.110030e+00	1.387824e+01	5.988169e+03	3.564571e+03	5.126067e-01
1.934274e+04	1.349416e+01	5.565988e+04	3.831337e+01	1.177566e+01	1.619278e+04	1.282901e+04	4.998411e-01
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.200000e+01	1.024000e+03	0.000000e+00	1.000000e+00
0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	2.200000e+01	1.024000e+03	0.000000e+00	1.000000e+00
1.881421e+07	1.318300e+04	3.511003e+07	2.342900e+04	2.470000e+02	6.553500e+04	6.553500e+04	1.000000e+00

Figure 8: Dados estatísticos relativamente aos valores numéricos.

No que toca ao estudo das *features* categóricas, o grupo optou, mais uma vez, por grafos circulares para entender a distribuição das mesmas e para identificar padrões ou tendências nos dados. Como podemos ver pela Figura 9, *features* está uniformemente distribuída. Aliás, devido à grande variedade de valores únicos em cada uma destas *features*, os valores com menos de 2% foram agrupados, sendo estes os conjunto *Others*.

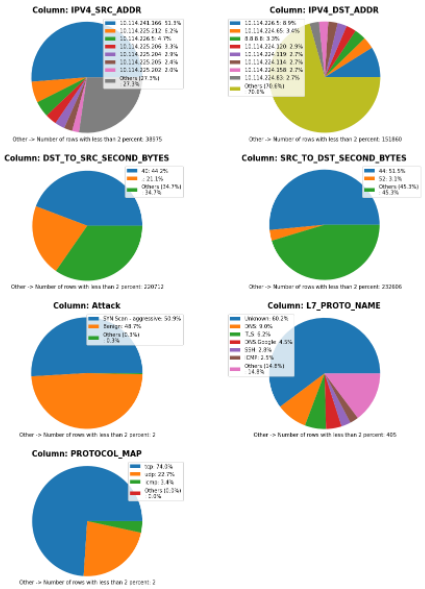

Figure 9: Distribuição dos dados das *features* categóricas.

Em relação à distribuição dos dados das *features* numéricas, algumas das *features* apresentam uma faixa de valores ampla. Contudo, existe uma concentração de valores em torno de determinados valores. Relativamente à dispersão dos dados das *features* numéricas, para a observação destas, foi aplicada a estratégia de remover os *outliers* para obter uma melhor visualização dos gráficos, ou seja, foram observados e analisados a dispersão dos valores com e sem *outliers*.

Por último, para este *dataset* também foi criado um gráfico de dispersão onde os pontos são posicionados nas coordenadas definidas pelas *features L4_SRC_PORT* e *L4_DST_PORT*. A cor dos pontos é determinada pela *feature Attack*, possibilitando visualizar a relação entre as portas de origem e destino e se um ataque ocorreu ou não. Na figura 10 podemos observar este mesmo gráfico.

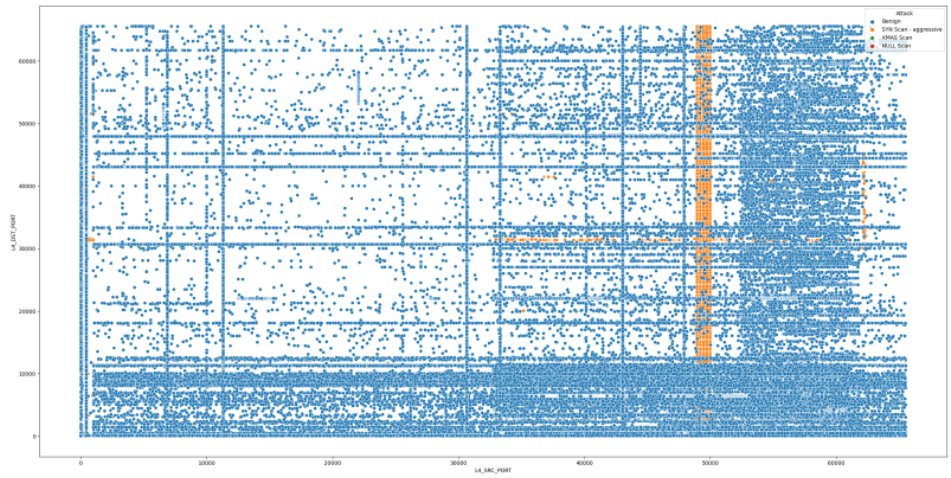

Figure 10: Relação entre as portas de origem e destino e se um ataque ocorreu ou não

As imagens utilizadas no relatório estão com um tamanho pequeno. Deste modo, no repositório do grupo estão disponibilizados dois *notebooks*, *data_exploration_nf* e *data_exploration_sigarml*, que contém a análise e exploração do *dataset NF-UQ-NIDS-v2* e do *dataset SIMARGL2021*, respetivamente. Também é importante referir, que durante o projeto foram realizadas outras análises e explorações que se encontram nos outros *notebooks*. Como, por exemplo, o *notebook data_exploration* que contém um pouco da exploração realizada no início do trabalho, após o *merge* das duas fontes de dados. Por fim, também o *notebook data_processing*, contém exploração das fontes de dados em separado, para além do tratamento de dados e do treino dos modelos.

4 Tratamento de dados

Após a análise da informação do *dataset*, segue-se a tarefa de tratamento e limpeza dos dados, sendo esta constituída por várias etapas. Neste processo foram aplicadas as técnicas ***Feature Selection*** e ***Label Enconding***.

Assim sendo, a técnica *Feature Selection* levou à remoção das *features* PROTOCOL_MAP, IPV4_SRC_ADDR e IPV4_DST_ADDR. A eliminação destas colunas deveu-se ao facto de a PROTOCOL_MAP ser valor categórico da PROTOCOL, e as colunas dos IPs de origem e destino não fazerem sentido para o treino do modelo, uma vez que estas identificam especificamente os intervenientes dos pedidos e não contribuírem para a generalização do problema.

Em seguida, foi aplicada a técnica *Label Enconding* nas *features* L7_PROTO_NAME e a Attack e eliminiadas as linhas com valores duplicados.

Posteriormente, para o *dataset* SIMARGL, foi necessário processar as colunas SRC_TO_DST_SECOND_BYTES e DST_TO_SRC_SECOND_BYTES. Estas colunas contém uma lista de bytes transferidos por segundo da *source* ao destino e vice-versa, respetivamente, sendo que esta está representada por uma string com os valores separados por vírgulas, como por exemplo "12,,,3,,45,,31,51". Deste modo, estas listas foram convertidas para o valor total transferido. No entanto, como estes valores eram muito grandes para o cálculo de algumas métricas, foram-lhes aplicadas transformações logarítmicas, reduzindo assim a escala e tamanho em memória necessário para os guardar. Para além disto, também foi aplicada a transformação logarítmica nos restantes dados numéricos contínuos devido a estes estarem bastante enviesados para a esquerda, o que ajudou a resolver alguns *outliers* e a melhorar a performance do modelo. O impacto desta transformação será abordada e analisada mais à frente.

O tratamento de dados foi efetuado no notebook *data_processing*, onde foram experimentados diferentes abordagens a esta tarefa mas que no final a pipeline está resumida na função *prepare_dataset()*.

5 Treino e avaliação dos modelos

Tendo disponível dois *datasets* diferentes, foi decidido treinar os modelos no *dataset* NF e testar no SIMARGL. Desta forma conseguimos avaliar a performance do modelo ao generalizar para redes diferentes em vez de ser só apenas para uma rede em específico. No entanto, sendo que o tipo de ataques capturados nos diferentes *datasets* são diferentes, os testes da identificação de que ataque em específico foi executado apenas será avaliado dentro dos mesmos *datasets*. Adicionalmente, toda a experimentação com o treino dos modelos foi efetuada no ficheiro *data_processing*. Os datasets são constituídos pelas colunas comuns ao dois, sendo estas as seguintes:

- L4_SRC_PORT
- L4_DST_PORT
- PROTOCOL
- IN_BYTES
- OUT_BYTES
- TCP_FLAGS
- FLOW_DURATION_MILLISECONDS
- MIN_IP_PKT_LEN
- MAX_IP_PKT_LEN
- SRC_TO_DST_SECOND_BYTES
- DST_TO_SRC_SECOND_BYTES
- RETRANSMITTED_IN_BYTES
- RETRANSMITTED_OUT_BYTES
- RETRANSMITTED_IN_PKTS
- RETRANSMITTED_OUT_PKTS
- TCP_WIN_MAX_IN
- TCP_WIN_MAX_OUT
- L7_PROTO_NAME
- Label
- Attack

5.1 Deteção de ataques

Neste grupo de testes, serão treinados e testados modelos que consigam fazer a previsão correta da *Label*, ou seja, se um registo corresponde a um ataque informático ou não.

A tabela 1 apresenta os resultados dos testes iniciais. Como podemos observar, os resultados apresentados são bastante negativos, onde no conjunto de teste a performance foi substancialmente inferior à dos conjuntos de treino e validação.

Após alguma análise destes resultados e dos dados dos *datasets*, concluímos que isto poderia ser das *features* L4_SRC_PORT e L4_DST_PORT, uma vez que estas podiam ser demasiado específicas para o *dataset* de treino e no *dataset* de teste ser uma realidade completamente diferente. Para isso, voltou-se a treinar os *datasets* sem estas duas *features* e os resultados obtidos foram registados na tabela 2.

Esta tabela mostra que a performance no *dataset* de teste duplicou, onde o modelo mostrou já conseguir obter uma performance minimamente satisfatória para o conjunto de teste. Este modelo também foi testado para outros números de estimadores, o que melhorou ainda mais os resultados,

Train + validation set	Test set	Log Transform	F1-score			Nº estimators
			Train	Val	Test	
NF	SIMARGL	Não	0.9997	0.9899	0.3730	150
NF	SIMARGL	Sim	0.9996	0.9895	0.3823	150

Table 1: Tempos de execução dos *merges*

Train + validation set	Test set	Log Transform	F1-score			Nº estimators
			Train	Val	Test	
NF	SIMARGL	Não	0.9776	0.9758	0.6842	150
NF	SIMARGL	Sim	0.9773	0.9756	0.6934	150
NF	SIMARGL	Sim	0.9773	0.9756	0.7081	250
NF	SIMARGL	Sim	0.9772	0.9743	0.8101	350
NF	SIMARGL	Sim	0.9772	0.9744	0.8104	450

Table 2: Testes da Random Forest sem as features L4_SRC_PORT e L4_DST_PORT

tendo sido o melhor com 450 estimadores para um f1-*score* de 0.8104. O grupo gostaria de ter experimentado com mais parâmetros, no entanto, o tempo de treino era imenso. este demorava mais de 20 minutos por treino.

Adicionalmente, através do método de *feature permutation*, o grupo foi tentar encontrar as *features* mais importantes para esta tarefa. De seguida, o modelo foi testado selecionando as primeiras n *features* determinadas como mais importantes, onde o valor n era variável. Nesta tarefa foi utilizado o melhor modelo encontrado previamente, ou seja, aquele que é constituído por 450 estimadores. As *features* mais importantes estão apresentadas na figura 11.

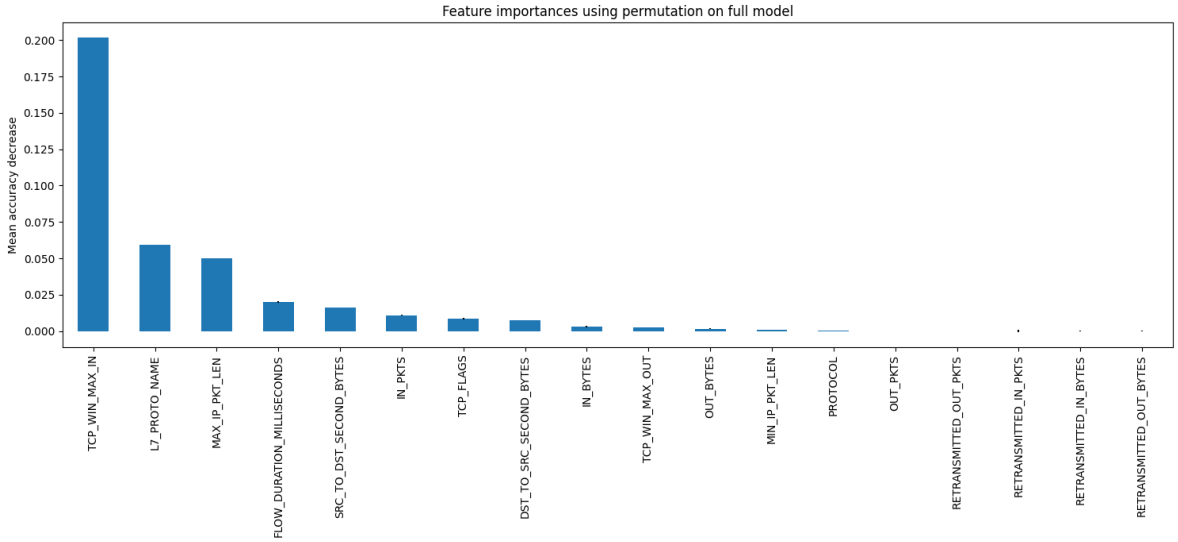

Figure 11: *Features* mais importantes para a deteção de ataques

Na figura 11, conseguimos observar que a *feature* TCP_WIN_MAX_IN é a mais importante e com mais impacto nos resultados, seguida da *feature* L7_PROTO_NAME e da MAX_IP_PKT_LEN. Analisando os resultados da tabela 3, podemos ver que para obter resultados minimamente satisfatórios são precisas pelo menos as 9 *features* mais importantes, contudo, para obter os melhores resultados foram necessárias as 14 *features* mais importantes, com uma diferença de 0.01, mostrando que as *features* que retratam as retransmissões acabam até por penalizar os resultados.

Train + validation set	Test set	Nº features mais importantes	F1-score		
			Train	Val	Test
NF	SIMARGL	3	0.9660	0.9758	0.2669
NF	SIMARGL	6	0.9759	0.9729	0.2866
NF	SIMARGL	9	0.9771	0.9741	0.7210
NF	SIMARGL	12	0.9772	0.9742	0.8030
NF	SIMARGL	14	0.9772	0.9743	0.8204
NF	SIMARGL	Completo	0.9772	0.9744	0.8104

Table 3: Resultados do modelo com as *features* mais importantes para a deteção de ataques

5.2 Identificação de ataques

Neste grupo iremos treinar o modelo não só para detetar se os dados se referem ou não a um ataque, mas também para identificar que tipo de ataque. Sendo que os dois *datasets* utilizados neste trabalho têm identificados ataques diferentes, estes testes foram realizados apenas com o NF-UQ-NIDS-v2, dividindo-o em 3 conjuntos semelhantes, o de treino (80%), validação (10%) e teste (10%). Desta forma, não conseguimos testar a capacidade do modelo generalizar para outras redes, mas podemos foi possível ter uma ideia do seu comportamento para essa situação através da interpretação dos resultados.

Log Transform	F1-score			Nº estimators
	Train	Val	Test	
Não	0.8904	0.6981	0.6342	150
Sim	0.8875	0.6951	0.6337	150
Sim	0.8908	0.6972	0.6342	250
Não	0.8878	0.6941	0.6327	250
Sim	0.8908	0.6992	0.6339	350
Sim	0.8908	0.6983	0.6342	450
Não	0.8879	0.6938	0.6326	450

Table 4: Testes da *Random Forest* para classificação de ataques

Na tabela 4 podemos observar os resultados dos vários modelos testados para este problema, e conseguimos logo observar que esta se mostrou uma tarefa mais difícil do que apenas identificar se ocorreu um ataque ou não. Os resultados mostraram-se bastante constantes, independentemente do número de estimadores usados. A transformação logarítmica nesta situação também não favoreceu os resultados obtidos.

De modo, a podermos encontrar as *features* mais importantes para esta tarefa, recorremos novamente ao método de *feature permutation*. Posteriormente, foram também avaliados os modelos com as n melhores *features* à semelhança da tarefa anterior. O modelo utilizado para este teste foi a *Random Forest* com 250 estimadores.

Train + validation set	Test set	Nº features mais importantes	F1-score		
			Train	Val	Test
NF	SIMARGL	6	0.8324	0.6255	0.6340
NF	SIMARGL	9	0.8441	0.6728	0.6437
NF	SIMARGL	12	0.8441	0.6728	0.6381
NF	SIMARGL	14	0.8905	0.6976	0.6347
NF	SIMARGL	Completo	0.8879	0.6938	0.6326

Table 5: Resultados do modelo com as *features* mais importantes para a identificação de ataques

Comparando com as *features* mais importantes para a deteção de ataques, as para a identificação dos ataques mostra que a MAX_IP_PKT_LEN apresenta uma maior importância, tal como as L7_PROTO_NAME, FLOW_DURATION_MILLISECONDS e TCP_WIN_MAX_IN mostram ser

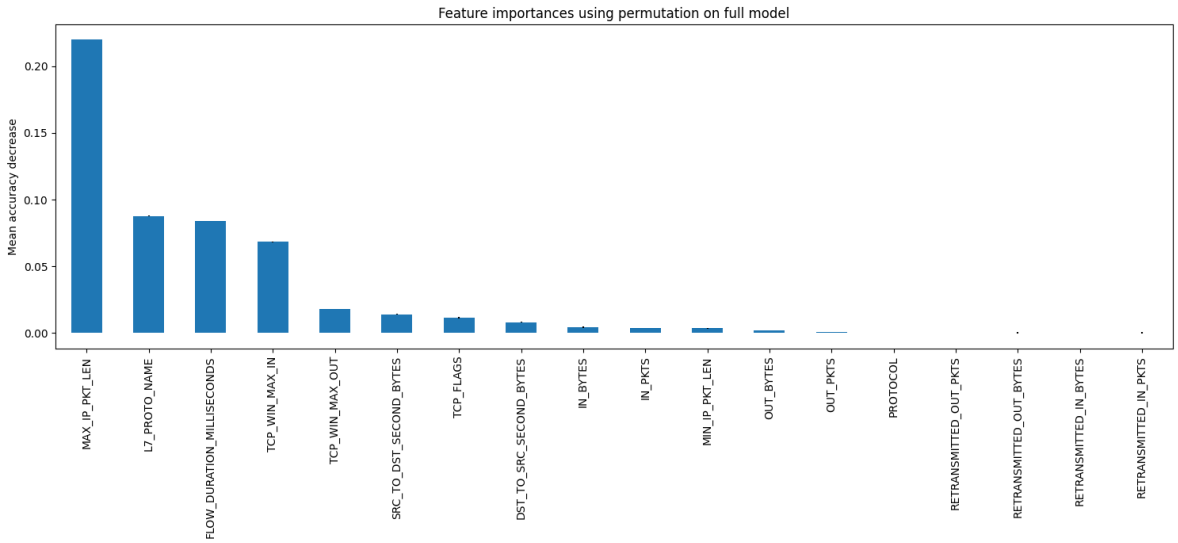

Figure 12: *Features* mais importantes para a deteção de ataques

também mais importantes para a classificação correta dos ataques. Isto era esperado pois para esta tarefa não só tem de detetar se um input é ou não um ataque, como também tem de analisar com mais precisão certas *features* para classificar corretamente o ataque. Analisando a tabela 5, podemos observar que o modelo apenas usando as 9 *features* mais importantes consegue obter melhores resultados do que com as *features* todas, com uma diferença de 0.0111.

6 Packet Sniffer

Neste projeto, foi desenvolvido um *packet sniffer* (*sniffer.py*) personalizado para extrair dados de pacotes de rede e, em seguida, aplicar no modelo de deteção de ataques de rede implementado para classificar esses pacotes como maliciosos ou benignos, para além disso também tem a capacidade de identificar o tipo de ataque.

A implementação do *packet sniffer* foi realizada usando a linguagem de programação **Python** e a biblioteca **Scapy**. Esta biblioteca fornece uma interface flexível receber e analisar pacotes de rede.

Após a implementação do *packet sniffer*, foi possível capturar pacotes de rede em tempo real. Esses pacotes foram processados para extrair informações relevantes, como protocolos utilizados, portas envolvidas e outros campos necessários. Esses dados foram então fornecidos ao modelo de deteção de ataques, que classificou os pacotes.

A combinação do *packet sniffer* com o modelo de deteção de ataques permitiu uma análise mais aprofundada do tráfego de rede. Ao extrair informações dos pacotes e submetê-los ao modelo, foi possível identificar possíveis ataques de rede, como *scanning* de portas, injeção de pacotes, ataques DoS (Denial of Service), entre outros. Isso fornece uma camada adicional de segurança, ajudando a identificar atividades maliciosas e proteger os sistemas contra possíveis ameaças.

7 Conclusão

No presente trabalho, desenvolvemos e implementámos um sistema de deteção de ataques de rede utilizando modelos de aprendizagem e um *packet sniffer* para extrair dados dos pacotes em tempo real. A nossa abordagem mostrou-se razoavelmente eficaz na deteção de possíveis ataques, no entanto, os resultados para a classificação dos mesmos não foram os mais satisfatórios.

Ao utilizar *datasets* relevantes e diversificados para treinar o nosso modelo, pudemos obter resultados promissores na deteção de ataques de rede. Através de técnicas de aprendizagem, o modelo foi capaz de aprender padrões e características que podem ser usados para a deteção de ataques. O modelo resultante conseguiu obter um f*1-score* de 0.8204, um resultado satisfatórios. Sendo que o modelo foi treinado num *dataset* e testado noutro, este mostrou ter a capacidade de generalizar para redes diferentes. Para a tarefa de classificação de ataques o modelo já teve algumas dificuldades em executar a mesma com boa precisão, tendo obtido um f1-*score* apenas de 0.6437, apenas precisando de 14 *features*. Sendo que este modelo foi testado apenas num único *dataset*, não conseguimos diretamente avaliar a capacidade de generalização para outras redes, no entanto, sendo que para o mesmo *dataset* os resultados foram baixos, para outras redes podemos assumir que os resultados possivelmente serão ainda menores, como aconteceu durante a tarefa da deteção de ataques. Vimos também que este modelo apenas precisa de 9 *features* para ter estes resultados.

A integração de um *packet sniffer* no sistema permitiu que os dados dos pacotes fossem capturados em tempo real, proporcionando um fluxo contínuo de informações para análise e deteção de possíveis ameaças. Isso possibilitou uma resposta mais rápida a incidentes de segurança, uma vez que ataques em execução podem ser identificados em tempo real.

No entanto, é importante ressaltar que a deteção de ataques de rede é um desafio em constante evolução, uma vez que novas técnicas e estratégias são desenvolvidas pelos agressores. Portanto, é fundamental manter o modelo de deteção atualizado e em constante adaptação, incorporando novos conjuntos de dados e técnicas de aprendizagem à medida que surgem.

Em suma, nosso trabalho demonstrou que a combinação do treino de modelo utilizando *datasets* e a utilização de um *packet sniffer* em tempo real é uma abordagem promissora para a deteção de ataques de rede. Essa integração oferece uma solução eficiente e em tempo real para proteger as redes contra ameaças em constante evolução, contribuindo para a segurança e confiabilidade dos sistemas de informação.

8 Anexos

8.1 Anexo A

- IPV4_SRC_ADDR - endereço de origem IPv4 (atributo qualitativo nominal, representado sob a forma de *string*);
- L4_SRC_PORT - endereço de destino IPv4 (atributo qualitativo nominal, representado sob a forma de *integer*);
- IPV4_DST_ADDR - número da porta de origem IPv4 (atributo qualitativo nominal, representado sob a forma de *string*);
- L4_DST_PORT - número da porta de destino IPv4 (atributo qualitativo nominal, representado sob a forma de *integer*);
- PROTOCOL - identificador do protocolo IP byte (atributo qualitativo nominal, representado sob a forma de *integer*);
- L7_PROTO - protocolo da camada 7, numérico (atributo qualitativo nominal, representado sob a forma de .....);
- IN_BYTES - número de bytes de entrada (atributo quantitativo contínuo, representado sob a forma de *integer*);
- IN_PKTS - número de pacotes recebidos (atributo quantitativo contínuo, representado sob a forma de *integer*);
- OUT_BYTES - número de bytes de saída (atributo quantitativo contínuo, representado sob a forma de *integer*);
- OUT_PKTS - número de pacotes de saída (atributo quantitativo contínuo, representado sob a forma de *integer*);
- TCP_FLAGS - acumulativo de todos os sinalizadores TCP (atributo quantitativo discreto, representado sob a forma de *integer*);
- CLIENT_TCP_FLAGS - acumulado de todos os sinalizadores TCP do cliente (atributo quantitativo discreto, representado sob a forma de *integer*);
- SERVER_TCP_FLAGS - acumulado de todos os sinalizadores TCP do servidor (atributo quantitativo discreto, representado sob a forma de *integer*);
- FLOW_DURATION_MILLISECONDS - duração do fluxo em milissegundos (atributo quantitativo contínuo, representado sob a forma de *integer*);
- DURATION_IN - duração do fluxo do cliente para o servidor (mseg) (atributo quantitativo contínuo, representado sob a forma de *integer*);
- DURATION_OUT - duração do fluxo do cliente para o servidor (mseg) (atributo quantitativo contínuo, representado sob a forma de *integer*);
- MIN_TTL - TTL mínimo do fluxo (atributo quantitativo discreto, representado sob a forma de *integer*);
- MAX_TTL - TTL máximo do fluxo (atributo quantitativo discreto, representado sob a forma de *integer*);
- LONGEST_FLOW_PKT - pacote mais longo (bytes) do fluxo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- SHORTEST_FLOW_PKT - pacote mais curto (bytes) do fluxo (atributo quantitativo contínuo, representado sob a forma de *integer*);

- MIN_IP_PKT_LEN - comprimento do pacote IP mais pequeno do fluxo observado (atributo quantitativo contínuo, representado sob a forma de *integer*);
- MAX_IP_PKT_LEN - comprimento do maior pacote IP do fluxo observado (atributo quantitativo contínuo, representado sob a forma de *integer*);
- SRC_TO_DST_SECOND_BYTES - bytes/segundo da origem para destino (atributo quantitativo contínuo, representado sob a forma de *float*);
- DST_TO_SRC_SECOND_BYTES - destino para origem bytes/segundo (atributo quantitativo contínuo, representado sob a forma de *float*);
- RETRANSMITTED_IN_BYTES - número de bytes de fluxo TCP retransmitidos (origem-¿destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_IN_PKTS - número de pacotes de fluxo TCP retransmitidos (origem-¿destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_OUT_BYTES - número de bytes de fluxo TCP retransmitidos (destino-¿origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_OUT_PKTS - número de pacotes de fluxo TCP retransmitidos (destino-¿origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- SRC_TO_DST_AVG_THROUGHPUT - taxa de transferência média da fonte para a fonte (bps) (atributo quantitativo contínuo, representado sob a forma de *integer*);
- DST_TO_SRC_AVG_THROUGHPUT - taxa de transferência média do destino para a origem (bps) (atributo quantitativo contínuo, representado sob a forma de *integer*);
- NUM_PKTS_UP_TO_128_BYTES - pacotes cujo tamanho IP ≤ 128 (atributo quantitativo discreto, representado sob a forma de *integer*);
- NUM_PKTS_128_TO_256_BYTES - pacotes cujo tamanho do IP é $\succ 128$ e ≤ 126 (atributo quantitativo discreto, representado sob a forma de *integer*);
- NUM_PKTS_256_TO_512_BYTES - pacotes cujo tamanho de IP $\succ 256$ e ≤ 512 (atributo quantitativo discreto, representado sob a forma de *integer*);
- NUM_PKTS_512_TO_1024_BYTES - pacotes cujo tamanho de IP $\succ 512$ e ≤ 1024 (atributo quantitativo discreto, representado sob a forma de *integer*);
- NUM_PKTS_1024_TO_1514_BYTES - pacotes cujo tamanho de IP $\succ 1024$ e ≤ 1514 (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MAX_IN - janela TCP máxima (origem-¿destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MAX_OUT - janela máxima de TCP (destino-¿origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- ICMP_TYPE - tipo ICMP * 256 + código ICMP (atributo quantitativo discreto, representado sob a forma de *integer*);
- ICMP_IPV4_TYPE - tipo ICMP (atributo quantitativo discreto, representado sob a forma de *integer*);
- DNS_QUERY_ID - ID da transação de consulta DNS (atributo quantitativo discreto, representado sob a forma de *integer*);
- DNS_QUERY_TYPE - tipo de consulta DNS (por exemplo, 1=A, 2=NS...) (atributo quantitativo discreto, representado sob a forma de *integer*);

- DNS_TTL_ANSWER - TTL do primeiro registo A (se existir) (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FTP_COMMAND_RET_CODE - código de retorno do comando do cliente FTP (atributo quantitativo discreto, representado sob a forma de *float*);
- *Label* - label (atributo qualitativo nominal, representado sob a forma de *integer*);
- *Attack* - tipo de ataque (atributo qualitativo categórico, representado sob a forma de *string*);

8.2 Anexo B

- BIFLOW_DIRECTION - direção do fluxo de rede em uma conexão (atributo qualitativo, representado sob a forma de *integer*);
- DIRECTION - direção do tráfego de rede em um determinado registo (atributo qualitativo, representado sob a forma de *integer*);
- DST_TO_SRC_SECOND_BYTES - quantidade de bytes transferidos por segundo do destino para a origem durante uma conexão de rede (atributo quantitativo discreto, representado sob a forma de *string*);
- FIREWALL_EVENT - identifica se um evento registado é relacionado a um *firewall* (atributo qualitativo, representado sob a forma de *integer*);
- FIRST_SWITCHED - o momento em que o primeiro pacote da conexão foi registado ou *switched* (atributo quantitativo discreto, representado sob a forma de *integer*);
- FLOW_ACTIVE_TIMEOUT - duração máxima permitida para uma conexão antes de ser considerada inativa e encerrada (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_DURATION_MICROSECONDS - duração do fluxo de dados em microssegundos para cada registo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_DURATION_MILLISECONDS - duração do fluxo de dados em milissegundos para cada registo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_END_MILLISECONDS - tempo de término do fluxo de dados em milissegundos atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_END_SEC - o tempo de término do fluxo de dados em segundos (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_ID - identificador único para cada fluxo de rede (atributo qualitativo, representado sob a forma de *integer*);
- FLOW_INACTIVE_TIMEOUT - o tempo em que o fluxo ficou inativo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_START_MILLISECONDS - o tempo de início do fluxo, em milissegundos (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FLOW_START_SEC - o tempo de início do fluxo, em segundos (atributo quantitativo contínuo, representado sob a forma de *integer*);
- FRAME_LENGTH - o comprimento do frame (atributo quantitativo contínuo, representado sob a forma de *integer*);
- IN_BYTES - número de bytes de entrada (atributo quantitativo contínuo, representado sob a forma de *integer*);
- IN_PKTS - número de pacotes recebidos (atributo quantitativo contínuo, representado sob a forma de *integer*);

- IPV4_DST_ADDR - número do endereço de destino IPv4 (atributo qualitativo nominal, representado sob a forma de *string*);
- IPV4_SRC_ADDR - número do endereço de origem IPv4 (atributo qualitativo nominal, representado sob a forma de *string*);
- L4_DST_PORT - número da porta de destino do nível 4 (atributo qualitativo nominal, representado sob a forma de *integer*);
- L4_SRC_PORT - endereço de origem nível 4 (atributo qualitativo nominal, representado sob a forma de *integer*);
- LAST_SWITCHED - o último tempo registado em que ocorreu uma transição para este fluxo (atributo quantitativo discreto, representado sob a forma de *integer*);
- MAX_IP_PKT_LEN - o comprimento máximo do pacote IP dentro do fluxo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- MIN_IP_PKT_LEN - o comprimento mínimo do pacote IP dentro do fluxo (atributo quantitativo contínuo, representado sob a forma de *integer*);
- OOORDER_IN_PKTS - indica quantos pacotes de entrada chegaram fora de ordem no fluxo de rede (atributo quantitativo discreto, representado sob a forma de *integer*);
- OOORDER_OUT_PKTS - indica quantos pacotes de saída foram enviados fora de ordem no fluxo de rede (atributo quantitativo discreto, representado sob a forma de *integer*);
- OUT_BYTES - número de bytes de saída (atributo quantitativo contínuo, representado sob a forma de *integer*);
- OUT_PKTS - número de pacotes de saída (atributo quantitativo contínuo, representado sob a forma de *integer*);
- PROTOCOL - identificador do protocolo IP byte (atributo qualitativo nominal, representado sob a forma de *integer*);
- PROTOCOL_MAP - protocolos de rede usados nos fluxos (atributo qualitativo nominal, representado sob a forma de *string*);
- RETRANSMITTED_IN_BYTES - número de bytes de fluxo TCP retransmitidos (origem -> destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_IN_PKTS - número de pacotes de fluxo TCP retransmitidos (origem -> destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_OUT_BYTES - número de bytes de fluxo TCP retransmitidos (destino -> origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- RETRANSMITTED_OUT_PKTS - número de pacotes de fluxo TCP retransmitidos (destino -> origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- SRC_TO_DST_SECOND_BYTES - o número de bytes por segundo enviados do endereço origem para o endereço de destino durante fluxo
- TCP_FLAGS - os sinais TCP
- TCP_WIN_MAX_IN - janela TCP máxima (origem -> destino) (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MAX_OUT - janela máxima de TCP (destino -> origem) (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MIN_IN - o tamanho mínimo da janela de receção TCP no fluxo de entrada (atributo quantitativo discreto, representado sob a forma de *integer*);

- TCP_WIN_MIN_OUT - o tamanho mínimo da janela de receção TCP no fluxo de saída (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MSS_IN - o valor do Maximum Segment Size (MSS) TCP no fluxo de entrada (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_MSS_OUT - o valor do Maximum Segment Size (MSS) TCP no fluxo de saída (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_SCALE_IN - o fator de escala da janela de receção TCP no fluxo de entrada (atributo quantitativo discreto, representado sob a forma de *integer*);
- TCP_WIN_SCALE_OUT - o fator de escala da janela de receção TCP no fluxo de saída (atributo quantitativo discreto, representado sob a forma de *integer*);
- SRC_TOS - o valor Type of Service (ToS) do endereço de origem no pacote (atributo quantitativo discreto, representado sob a forma de *integer*);
- DST_TOS - o valor ToS do endereço de destino no pacote (atributo quantitativo discreto, representado sob a forma de *integer*);
- L7_PROTO_NAME - o nome do protocolo de nível 7 (atributo qualtitativo categórico, representado sob a forma de *string*);
- SAMPLING_INTERVAL - o período de tempo entre as amostras do fluxo de rede (atributo quantitativo contínuo, representado sob a forma de *integer*);
- TOTAL_FLOWS_EXP - o número total de fluxos (atributo quantitativo discreto, representado sob a forma de *integer*);
- LABEL - label (atributo qualitativo categórico, representado sob a forma de *string*);