UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de São José do Rio Preto

AMANDA BARBOSA SOBRINHO

MODELO DE *CYBER SITUATIONAL AWARENESS* PARA SISTEMAS AUTÔNOMOS UTILIZANDO *NETFLOW*

São José do Rio Preto
2021

AMANDA BARBOSA SOBRINHO

MODELO DE *CYBER SITUATIONAL AWARENESS* PARA SISTEMAS AUTÔNOMOS UTILIZANDO *NETFLOW*

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Orientador:
Prof. Dr. Adriano Mauro Cansian

São José do Rio Preto
2021

B238m

Barbosa Sobrinho, Amanda
Modelo de Cyber Situational Awareness para Sistemas Autônomos utilizando Netflow / Amanda Barbosa Sobrinho. -- São José do Rio Preto, 2021
107 f. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto
Orientador: Adriano Mauro Cansian

1. Sistemas de segurança. 2. Redes de computadores Monitorização. 3. Sistemas de detecção de intrusão (Medidas de segurança). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

AMANDA BARBOSA SOBRINHO

MODELO DE *CYBER SITUATIONAL AWARENESS* PARA SISTEMAS AUTÔNOMOS UTILIZANDO *NETFLOW*

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus de São José do Rio Preto.

Comissão Examinadora

Prof. Dr. Adriano Mauro Cansian
UNESP – *Campus* de São José do Rio Preto
Orientador

Prof. Dr. Kelton Augusto Pontara da Costa
UNESP – *Campus* de Bauru

Prof. Dr. Robson de Oliveira Albuquerque
UnB – Universidade de Brasília

São José do Rio Preto
10 de Setembro de 2021

Dedico esse trabalho a todos aqueles que me deram auxílio na confecção desse texto, e também me apoiaram durante os anos de estudo.

AGRADECIMENTOS

Agradeço, primeiramente, aos meus pais, à minha irmã e ao meu namorado, pelo carinho, suporte e paciência em todos os momentos.

Agradeço aos meus colegas do laboratório ACME!, João Otavio Gonçalves Calis e Bruno Ferreira Leal, pela valiosa ajuda fornecida durante todo o desenvolvimento desta pesquisa.

Agradeço ao meu orientador Adriano Mauro Cansian, pelo apoio, pelos ensinamentos fornecidos e pela contribuição para minha formação acadêmica e pessoal.

Agradeço ao professor Robson de Oliveira Albuquerque, pelas sugestões e ideias oferecidas, que contribuíram e enriqueceram substancialmente os resultados deste trabalho.

Agradeço, também, à Universidade Estadual Paulista "Júlio de Mesquita Filho", *campus* de São José do Rio Preto, pelo curso de pós-graduação ali ministrado por professores de excelência.

RESUMO

A expansão da *Internet*, apesar de possibilitar uma maior integração de usuários e serviços, acarretou também um aumento nos ataques contra as redes de computadores. Para mitigá-los, é necessário recuperar dados sobre a rede e compreender sua situação atual, de forma a identificar e detectar ameaças. Esta habilidade está relacionada ao conceito de Consciência Situacional em Cibersegurança (*Cyber Situational Awareness* – CSA), que se refere à capacidade de capturar a condição atual da rede, compreendê-la e projetar seu futuro próximo em qualquer momento do tempo. Os modelos de CSA têm recebido bastante destaque em estudos recentes, mas nota-se sua dificuldade em tratar de grandes volumes de informações, principalmente por utilizarem diversas fontes de dados como entrada, muitas vezes infringindo a privacidade dos usuários, e por demandarem alto custo computacional para processarem esse conjunto massivo de informações. Este trabalho apresenta um modelo de CSA flexível e escalável para redes de Sistemas Autônomos que utiliza apenas fluxos *Netflow* V9 como fonte de informação, o que faz com que não viole a privacidade dos usuários. O sistema proposto é capaz de processar o volume de fluxos exportado pelo roteador de borda do Sistema Autônomo da UNESP diariamente, fornecendo uma visão holística do tráfego recebido e dos eventos de segurança ocorrendo na organização, possibilitando a detecção desses eventos com um atraso mínimo, e auxiliando, assim, um administrador de rede a tomar medidas para mitigar esses eventos. Além de permitir a visualização de informações de tráfego e de eventos de segurança para o Sistema Autônomo inteiro, o modelo também apresenta alto grau de granularidade, viabilizando que essas mesmas análises sejam realizadas em redes específicas de instituições dentro da organização.

Palavras-Chave: Segurança de Redes, Detecção de Ataques, Consciência Situacional em Cibersegurança.

ABSTRACT

The expansion of the Internet, while allowing greater integration of users and services, also led to an increase in attacks against computer networks. To mitigate them, it is necessary to retrieve data from the network and understand its current situation in order to identify and detect threats. This capability is related to the concept of Cyber Situational Awareness (CSA), which refers to the ability to capture the current condition of the network, understand this condition and project its near future at any time. CSA models have received a lot of attention in recent studies, but it is noted the difficulty of the proposed models in dealing with large volumes of information, especially as they use various data sources as input, often violating the privacy of users, and because they demand high computational cost to process this massive set of information. This work proposes a flexible and scalable CSA model for Autonomous Systems that uses only Netflow V9 flows as source of information, which does not violate users' privacy. The proposed system is capable of processing the volume of flows exported by UNESP's Autonomous System border router on a daily basis, providing a holistic view of incoming traffic and security events occurring in the organization, allowing the detection of these events with a minimum delay, and thus helping a network administrator to take measures to mitigate these events. In addition to allowing the visualization of traffic information and security events for the entire Autonomous System, the model also presents a high granularity degree, enabling these same analyzes to be carried out in specific networks of institutions within the organization.

Keywords: Network Security, Attack Detection, Cyber Situational Awareness.

LISTA DE ILUSTRAÇÕES

Figura 1: Funcionamento simplificado da *Elastic Stack*. 32
Figura 2: Arquitetura do modelo de CSA proposto. 48
Figura 3: Trecho do *dashboard* de *Overview*..... 75
Figura 4: Trecho do *dashboard* de Geolocalização. 76
Figura 5: Trecho do *dashboard* de HTTP/HTTPS. 77
Figura 6: Trecho do *dashboard* de *Scan* de Hospedeiros..... 78
Figura 7: Trecho do *dashboard* de *Bruteforce*. 79
Figura 8: Trecho do *dashboard* de DoS/DDoS..... 80
Figura 9: Trecho do *dashboard* de *Miners*. 81
Figura 10: Barra de navegação para as redes do *campus* de São José do Rio Preto. . 90

LISTA DE TABELAS

Tabela 1: Estado da Arte em modelos de *Cyber Situational Awareness*. 43
Tabela 2: Lista de campos do *Netflow* V9 utilizados pelas assinaturas. 50
Tabela 3: Campos utilizados para a consulta de enriquecimento dos eventos..... 68
Tabela 4: Número de eventos detectados pelas assinaturas ao longo de sete dias..... 83
Tabela 5: Eventos detectados no *campus* de São José do Rio Preto em sete dias. 91
Tabela 6: Tempo médio de execução das assinaturas, ao longo de sete dias..... 93

LISTA DE ABREVIATURAS E SIGLAS

API: *Application Programming Interface*

APT: *Advanced Persistent Threat*

AS: *Autonomous Systems*

ASN: *Autonomous System Number*

CERT: Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança

CSA: *Cyber Situational Awareness*

DoS: *Denial of Service*

DDoS: *Distributed Denial of Service*

DNS: *Domain Name System*

ELK: *Elasticsearch, Logstash* e *Kibana*

FIN: *Finished*

FTP: *File Transfer Protocol*

HTTP: *Hyper Text Transfer Protocol*

HTTPS: *Hyper Text Transfer Protocol Secure*

ICMP: *Internet Control Message Protocol*

IETF: *Internet Engineering Task Force*

IoC: *Indicator of Compromise*

IP: *Internet Protocol*

IPFIX: *Internet Protocol Flow Information Export*

IRC: *Internet Relay Chat*

NAT: *Network Address Translation*

RNN: *Recurrent Neural Network*

RST: *Reset*

SQL: *Structured Query Language*

SMTP: *Simple Mail Transfer Protocol*

SSH: *Secure Shell*

SVM: *Support Vector Machine*

SYN: *Synchronize*

TCP: *Transmission Control Protocol*

URL: *Uniform Resource Locator*

VPS: *Virtual Private Server*

SUMÁRIO

CAPÍTULO 1 - Introdução 15

1.1 Introdução e Motivação 15

1.2 Justificativa 16

1.3 Objetivo 19

1.4 Organização do Trabalho 19

CAPÍTULO 2 - Fundamentação Teórica 21

2.1 Considerações Iniciais 21

2.2 Conceitos de Redes de Computadores 21

2.2.1 Fluxo de Rede 22

2.2.2 Protocolo *Netflow* 22

2.3 Ataques às Redes de Computadores 23

a) Ataques de *Bruteforce* 24

b) Ataques de *Scan* 24

c) Ataques de DoS (*Denial of Service*) 25

2.4 *Cyber Situational Awareness* (CSA) 26

2.5 *Threat Intelligence* e *Cyber Threat Intelligence* 27

2.5.1 *Feeds* de *Cyber Threat Intelligence* 27

2.6 Assinaturas de Detecção de Eventos 28

2.7 *Elastic Stack* 29

2.7.1 *Elasticsearch* 29

2.7.2 *Kibana* 30

2.7.3 *Filebeat* 31

2.7.4 Funcionamento da *Elastic Stack* 31

2.8 MITRE ATT&CK 32
CAPÍTULO 3 - Trabalhos Relacionados 34
3.1 Considerações Iniciais 34
3.2 Desafios em *Cyber Situational Awareness* 34
3.3 Modelos de *Cyber Situational Awareness* 38
3.3.1 *Elastic Stack* aplicada em modelos de *Cyber Situational Awareness* 40
3.4 Conclusões sobre os Trabalhos Relacionados 42
CAPÍTULO 4 - Metodologia 46
4.1 Considerações Iniciais 46
4.2 Visão geral 46
4.3 Coleta e indexação de fluxos *Netflow* 48
4.4 Detecção de eventos de segurança 49
4.4.1 Detecção de ataques 52
a) Detecção de ataques de *Scan* de Hospedeiro 53
b) Detecção de ataques de *Bruteforce* 54
c) Detecção de ataques de DoS/DDoS 56
4.4.2 Detecção de comunicações com hospedeiros maliciosos 60
a) Detecção de comunicações advindas de hospedeiros comprometidos 60
b) Detecção de comunicações com mineradores 62
c) Detecção de comunicações com hospedeiros de Comando e Controle 63
d) Detecção de comunicações com hospedeiros em uma *blacklist* 64
e) Detecção de comunicações com *Malwares* 66
4.5 Enriquecimento dos eventos detectados 67
4.6 Visualização de dados 70
4.7 Análise de dados 70
4.8 Persistência dos fluxos recebidos e eventos detectados 72
4.9 Manutenção da *Elastic Stack* 72

4.10 Automatização da instalação da *Elastic Stack* via *Ansible* 73
CAPÍTULO 5 - Resultados 74
5.1 Considerações Iniciais 74
5.2 *Dashboards* construídos 74
5.2.1 *Dashboard* de *Overview* 75
5.2.2 *Dashboard* de Geolocalização 76
5.2.3 *Dashboard* de HTTP/HTTPS 76
5.2.4 *Dashboard* de eventos de *Scan* de Hospedeiros 78
5.2.5 *Dashboard* de eventos de *Bruteforce* 79
5.2.6 *Dashboard* de DoS/DDoS 80
5.2.7 *Dashboards* de *feeds* de *Cyber Threat Intelligence* 80
5.3 Resultados apresentados pelos *dashboards* de tráfego 82
5.4 Avaliação dos eventos de segurança detectados 83
5.4.1 Observações gerais 83
5.4.2 Avaliação dos resultados de *Scan* de Hospedeiros 85
5.4.3 Avaliação dos resultados de *Bruteforce* 86
5.4.4 Avaliação das comunicações advindas de hospedeiros comprometidos ... 87
5.4.5 Avaliação das comunicações com mineradores 88
5.4.6 Avaliação das comunicações com hospedeiros de Comando e Controle .. 88
5.4.7 Avaliação das comunicações com hospedeiros em uma *blacklist* 89
5.4.8 Avaliação das comunicações com *Malwares* 89
5.5 Resultados para o *campus* de São José do Rio Preto 90
5.6 Avaliação das execuções das assinaturas de eventos 93
5.7 Considerações sobre os resultados obtidos 95
CAPÍTULO 6 - Conclusões 97
6.1 Considerações Iniciais 97
6.2 Conclusões gerais 97

6.3 Dificuldades encontradas..... 99

6.4 Trabalhos futuros..... 99

REFERÊNCIAS..... 101

CAPÍTULO 1 - Introdução

1.1 Introdução e Motivação

Com o avanço da *Internet* e sua expansão, integrando cada vez mais usuários e sendo palco de um número crescente de serviços, negócios e oportunidades, a quantidade de informações sensíveis que trafegam nessa rede tem crescido exponencialmente. Em contrapartida, também aumentaram proporcionalmente os ataques contra as redes de computadores e seus membros, como pode ser observado por meio das estatísticas disponibilizadas pelo Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.BR, 2020).

Esses dados são corroborados analisando incidentes recentes. Ataques de *ransomware*[1] a grandes empresas foram noticiados com frequência na primeira metade de 2021; algumas companhias notáveis que tiveram seus dados criptografados foram a fornecedora de *softwares* de monitoramento remoto Kaseya (KREBS, 2021), o escritório jurídico Campbell Conroy & O'Neil, P.C. (VAAS, 2021) e a fornecedora de carne bovina JBS (TUNG, 2021).

Para que seja possível defender sistemas contra invasores, é necessária a habilidade de recuperar dados sobre a rede e compreender os processos acontecendo neste ambiente, de forma a detectar ameaças para que sejam tomadas medidas apropriadas. Esta habilidade está intimamente ligada a um conceito conhecido como

[1] Código malicioso que torna inacessíveis os dados armazenados em um equipamento usando criptografia e que exige pagamento de resgate (*ransom*) para restabelecer o acesso ao usuário.

Consciência Situacional em Cibersegurança (*Cyber Situational Awareness* – CSA), que se refere à consciência da disponibilidade, confidencialidade, operações e integridade dos serviços de rede, e também abrange decisões sobre configuração, política e estratégias de monitoramento, detecção e respostas a eventos da rede (VARGA; WINKELHOLZ; TRÄBER-BURDIN, 2016).

A Consciência Situacional em Cibersegurança prevê a definição de um modelo que seja capaz de capturar a condição atual da rede, compreendê-la e projetar seu futuro próximo em qualquer momento do tempo, de forma a auxiliar o administrador de rede a tomar ações quando assim for necessário (ADENUSI, et al., 2015). O estudo de modelos de CSA tem recebido bastante destaque (HUSÁK; JIRSÍK; YANG, 2020), pois, diferentemente de outros sistemas de cibersegurança, os quais são voltados apenas para a detecção de ataques ou atividades maliciosas, a Consciência Situacional fornece uma visão da tendência de segurança da rede de forma global.

1.2 Justificativa

Apesar de diversos modelos de Consciência Situacional em Cibersegurança terem sido propostos ao longo dos anos em diferentes contextos (ADENUSI et al., 2015; CINQUE; DELLA CORTE; PECCHIA, 2020; HAN et al., 2019; JIRSIK; CELEDA, 2018, 2020; LAKKARAJU; YURCIK; LEE, 2004; LIU et al., 2019; YONG; XIAOBIN; HONGSHENG, 2007), notam-se mais recentemente duas tendências principais nos trabalhos publicados, também apontadas por Husák, Jirsík e Yang (2020): a primeira é o desenvolvimento de modelos de CSA capazes de gerenciar *Big Data*[2], e a segunda é a utilização de múltiplos tipos de fontes de dados como entrada para os modelos de CSA, como arquivos, sessões, informações de memória, registros, serviços, etc.

A primeira tendência é compreensível na medida em que grandes volumes de dados têm se tornado cada vez mais comuns, uma vez que o número de dispositivos nas redes, bem como o número de usuários, também tem sofrido um aumento exponencial. Segundo o Relatório Anual de Internet da Cisco (CISCO, 2020), havia

[2] Conjuntos de dados grandes demais para serem tratados e analisados por sistemas tradicionais.

um total de 18,4 bilhões de dispositivos conectados na rede em 2018, e espera-se chegar a 29,3 bilhões de dispositivos conectados em 2023. Já a utilização de diferentes tipos de fontes de dados de entrada para os modelos de CSA propostos deve-se ao aumento tanto da disponibilidade quanto da facilidade de acesso e armazenamento desses dados, que possuem cada vez mais detalhes sobre os dispositivos de rede e ações de seus usuários.

No entanto, é possível constatar que ambas as tendências ainda podem ser exploradas em conjunto, especialmente considerando o alto custo de processamento de dados ao utilizar diferentes fontes de informação, particularmente em ambientes de *Big Data* (HUSÁK; JIRSÍK; YANG, 2020). Nessa perspectiva, é possível notar que são necessários recursos computacionais consideráveis para que diferentes tipos de dados sejam coletados, armazenados e analisados em alguns dos modelos de CSA recentemente propostos (CINQUE; DELLA CORTE; PECCHIA, 2020; HAN et al., 2019; JIRSIK; CELEDA, 2018, 2020; LIU et al., 2019). Isso ressalta a relevância de se estruturar novos modelos de CSA que consigam tratar eficientemente dos dados gerados por redes de grande porte, como as redes de Sistemas Autônomos (ASs – *Autonomous Systems*).

Um Sistema Autônomo é um grupo de roteadores[3] sob o mesmo controle administrativo (pertencentes a uma mesma rede corporativa ou organização, por exemplo) e que possuem uma mesma política de roteamento[4] (KUROSE; ROSS, 2013). De forma geral, a *Internet* pode ser considerada um conjunto de Sistemas Autônomos, e, portanto, o estudo situacional da cibersegurança considerando um AS em sua totalidade é de grande relevância, uma vez que permitiria visualizar de maneira holística os eventos de segurança ocorrendo na organização simultaneamente, independentemente de os hospedeiros envolvidos estarem ou não na mesma rede. Esta é uma visão que só é possível por meio do acesso a informações de rede de um Sistema Autônomo em sua totalidade.

Além disso, sobre o uso de diferentes fontes de informações de entrada para os modelos de CSA, há um fator que deve ser considerado especificamente no Brasil. A Lei Nº 12.965, de 23 de Abril de 2014, conhecida como Marco Civil da *Internet*, estabelece no art. 9°, § 3°, que "Na provisão de conexão à *Internet*, onerosa ou

[3] Roteador é um dispositivo de rede responsável por encaminhar de um enlace de comunicação de entrada para um enlace de comunicação de saída.
[4] A política de roteamento define qual algoritmo é utilizado para realizar o roteamento de pacotes.

gratuita, bem como na transmissão, comutação ou roteamento, é vedado bloquear, monitorar, filtrar ou analisar o conteúdo dos pacotes de dados, respeitado o disposto neste artigo.” (BRASIL, 2014). Sendo assim, o único tipo de informação que pode ser usada por modelos de CSA de forma a respeitar o Marco Civil da *Internet* são os fluxos de rede, uma vez que estes utilizam apenas o tráfego como fonte de informação, em detrimento do conteúdo dos pacotes. Como todas as outras fontes de dados leem e empregam o conteúdo dos pacotes como base para seu funcionamento, elas não foram usadas neste trabalho.

Outro fator que ratifica o uso de fluxos como única entrada do modelo proposto é o alto custo de processamento de dados quando se tem várias fontes de informação diferentes (HUSÁK; JIRSÍK; YANG, 2020). Em alguns modelos de CSA propostos na literatura atual (CHAUDHARI et al., 2020; HAN et al., 2019; LIU et al., 2019), percebe-se a necessidade de recursos computacionais consideráveis para que as informações de diferentes tipos sejam capturadas, armazenadas e analisadas, principalmente em um âmbito de *Big Data*. Além disso, espera-se que, em um sistema de CSA, as informações sobre a rede sejam disponibilizadas o mais rápido possível, para que o operador consiga tomar decisões em tempo hábil (HUSÁK; JIRSÍK; YANG, 2020). Uma das vantagens do uso exclusivo de fluxos de rede é o baixo custo computacional para seu tratamento e processamento, já que são dados que ocupam menos espaço e têm uma menor complexidade quando comparados com conteúdos de pacotes, o que consequentemente reduz o tempo necessário para a obtenção de resultados a partir da análise de fluxos (LI et al., 2013; SPEROTTO et al., 2010; VIEGAS et. al., 2019).

Neste contexto, a elaboração de um modelo de Consciência Situacional em Cibersegurança para Sistemas Autônomos que utilize apenas fluxos de rede como fonte de informação é algo relevante, uma vez que permite a análise em organizações inteiras, sem a necessidade de recursos computacionais de custo impeditivo, mesmo tendo um grande volume diário de dados para ser analisado. A proposta desenvolvida neste trabalho possibilita estabelecer este modelo por meio da visualização de informações gerais de tráfego de rede, bem como da detecção e visualização de eventos de segurança, do Sistema Autônomo da UNESP. Para este propósito, foi utilizada a *Elastic Stack* (ELASTIC, 2021f) para realizar a indexação e armazenamento dos fluxos de rede, assim como a detecção de eventos de rede

maliciosos, por meio de consultas parametrizadas que são executadas em espaços de tempo predefinidos.

1.3 Objetivo

O objetivo deste trabalho é desenvolver um modelo de Consciência Situacional em Cibersegurança para Sistemas Autônomos utilizando fluxos *Netflow* V9. Este objetivo principal foi alcançado por meio de quatro objetivos específicos:

- Proposta de um modelo distribuído e escalável que permite a ingestão, processamento, enriquecimento, armazenamento, análise e visualização dos fluxos *Netflow* V9 do Sistema Autônomo da UNESP (ASN 53166) e dos eventos de segurança identificados por meio desses fluxos;
- Definição de um modelo de consulta flexível para detectar eventos de segurança usando os fluxos de entrada;
- Definição de um processo de análise de fluxos *Netflow* V9 que permite fazer o cruzamento de informações presentes nos fluxos crus com *feeds* de *Threat Intelligence* para identificar novas ameaças na rede monitorada;
- Estruturação de visualizações das características históricas e recentes do tráfego da rede, bem como dos eventos de segurança atuais e anteriormente detectados.

1.4 Organização do Trabalho

Este trabalho está dividido em seis capítulos, incluindo o atual. No capítulo 2 é apresentada a fundamentação teórica do trabalho, necessária para que o leitor se contextualize a respeito dos conceitos e tecnologias utilizadas. No capítulo 3 são apresentados os trabalhos correlatos e o estado da arte sobre pesquisas que tratam de modelos de Consciência Situacional em Cibersegurança. No capítulo 4 é descrita a metodologia proposta para o cumprimento dos objetivos, com o detalhamento de

cada uma das etapas do modelo. No Capítulo 5 são expostos os resultados obtidos e, por fim, no Capítulo 6, são apresentadas as conclusões, as dificuldades encontradas durante o desenvolvimento desta pesquisa e os trabalhos futuros propostos.

CAPÍTULO 2 - Fundamentação Teórica

2.1 Considerações Iniciais

Este capítulo tem como objetivo introduzir os conceitos fundamentais para o entendimento do trabalho apresentado. Na seção 2.2, são abordados conceitos de redes de computadores; na seção 2.3, é abordada a definição de ataques às redes de computadores e são descritos três tipos de ataques; na seção 2.4, é discutido o conceito de *Cyber Situational Awareness* (CSA); na seção 2.5, apresenta-se *Threat Intelligence* e *Cyber Threat Intelligence*; na seção 2.6, são abordadas assinaturas de detecção de eventos; na seção 2.7, são descritos a *Elastic Stack*, seus componentes e seu funcionamento básico; e, por fim, na seção 2.8 é apresentado o *framework* MITRE ATT&CK.

2.2 Conceitos de Redes de Computadores

Nesta seção, são definidos alguns conceitos utilizados no estudo de redes de computadores que serão úteis para a compreensão do funcionamento da metodologia proposta neste trabalho.

2.2.1 Fluxo de Rede

Segundo Li et al. (2013), um fluxo de rede é "uma sequência uni ou bidirecional de pacotes entre dois usuários finais, ou hospedeiros, com atributos comuns". Alguns exemplos de tais atributos podem ser duração, endereços IP de origem e destino, portas de transporte de origem e destino, protocolos da camada de rede e de transporte e *bytes* transferidos no fluxo.

Os fluxos podem ser obtidos a partir de algum ponto de observação da rede, que pode ser um *switch*[5] ou um roteador capaz de exportar fluxos e que fornece o comportamento da rede da perspectiva de onde está inserido. Usando esses dados de fluxo, é possível realizar análise e classificação do tráfego, além de detecção de eventos maliciosos, uma vez que armazenam detalhes de cada conexão estabelecida por qualquer máquina pertencente ao ambiente monitorado.

É importante ressaltar que limitações de *hardware* e *software* no equipamento utilizado para fazer a exportação podem ocasionar atrasos entre o momento em que o fluxo foi observado e o momento em que ele é exportado e posteriormente indexado (HUSÁK; JIRSÍK; YANG, 2020; SPEROTTO et al., 2010). Esse atraso é refletido em todos os processos que utilizarão os fluxos exportados como entrada.

2.2.2 Protocolo *Netflow*

O Protocolo *Netflow* é um protocolo de exportação de fluxos desenvolvido pela *Cisco Systems* por volta de 1996. Seu objetivo é padronizar o formato em que os fluxos de rede são exportados, com campos pré-definidos, de forma que possam ser usados para vários propósitos, como gerenciamento e planejamento de redes, bem como a detecção de atividades maliciosas (CISCO, 2011; CLAISE et al., 2004).

Um fluxo *Netflow* é exportado nas seguintes situações (CISCO, 2012): ficou inativo por 15 segundos; ficou ativo por mais de 30 minutos; uma *flag* TCP indicou o término do fluxo (*flags* FIN ou RST); a tabela de fluxos no equipamento ficou cheia.

---

[5] Um *switch* é um dispositivo de camada de enlace responsável por fazer comutação de pacotes desta camada.

Uma vez exportados, os fluxos podem ser armazenados para serem analisados posteriormente.

Diversos formatos para exportação de fluxos foram desenvolvidos à medida que o protocolo *Netflow* evoluiu. A versão mais recente é a 9, que é baseada em *templates*; o uso de *templates* permite definir quais campos do *Netflow* devem ser exportados, e facilita a adição/remoção de campos no momento da exportação, uma vez que implementações anteriores que empregam este padrão não precisam ser refeitas para que a alteração de campos ocorra.

Atualmente, o *Netflow* V9 possui 104 campos disponíveis para serem utilizados. Alguns destes campos são:

- **IN_BYTES:** número de *bytes* de entrada associados a um fluxo;
- **OUT_BYTES:** número de *bytes* de saída associados a um fluxo;
- **IN_PKTS:** número de pacotes de entrada associados a um fluxo;
- **OUT_PKTS:** número de pacotes de saída associados a um fluxo;
- **TCP_FLAGS:** acumulativo de todas as *flags* TCP vistas no fluxo;
- **IPV4_SRC_ADDR:** endereço IPv4 de origem;
- **IPV4_DST_ADDR:** endereço IPv4 de destino;
- **IPV6_SRC_ADDR:** endereço IPv6 de origem;
- **IPV6_DST_ADDR:** endereço IPv6 de destino;
- **L4_SRC_PORT:** número da porta de origem da camada de transporte;
- **L4_DST_PORT:** número da porta de destino da camada de transporte.

2.3 Ataques às Redes de Computadores

No contexto da segurança da informação, ataques podem ser definidos como tentativas, bem-sucedidas ou não, de subverter a autenticidade, confidencialidade, integridade e disponibilidade de um sistema computacional (GOLLMAN, 2011). Esse conceito também é válido para redes de computadores, que são alvo de diversos tipos de ataques com objetivos variados e diversas características específicas.

Os três tipos de ataques listados a seguir podem ser identificados utilizando fluxos *Netflow*, pois suas características representativas estão contidas nos campos que podem ser exportados pelo protocolo *Netflow* V9.

a) Ataques de *Bruteforce*

Os ataques de *Bruteforce* (PINKAS; SANDER, 2002) ou força bruta consistem em tentativas sucessivas de autenticação em um sistema por meio de combinações de credenciais. Exemplos de serviços visados por esse tipo de ataque são SSH[6], FTP[7] e *email*, comumente utilizados pela *Internet*. Essa classe de ataque se subdivide em duas categorias:

- **Ataque por exaustão:** testa todas as possibilidades de autenticação de um sistema. É eficaz quando o sistema não limita o número de tentativas de acesso e quando as credenciais do alvo são fracas (senhas curtas), mas pode ser extremamente ineficaz se essas duas condições não forem satisfeitas, ou seja, se o sistema limitar as tentativas de autenticação ou se a senha for muito longa;

- **Ataque de dicionário:** variante otimizada do ataque por exaustão que utiliza uma lista de palavras previamente construída (dicionário), a qual é composta por palavras normalmente usadas por usuários na composição de senhas. As tentativas de autenticação são feitas fixando um usuário e testando as palavras do dicionário como senha.

b) Ataques de *Scan*

Em um ataque de *Scan* ou varredura, o atacante tem como objetivo realizar um levantamento de informações sobre os dispositivos em uma rede e os serviços

---

[6] *Secure Shell* é um protocolo de rede utilizado para *login* remoto de usuários em hospedeiros.
[7] *File Transfer Protocol* é um protocolo de rede utilizado para troca de arquivos entre hospedeiros.

executados por eles (LEE; ROEDEL; SILENOK, 2003). As informações obtidas, que incluem portas abertas, endereços IP, versões de programas e versões de sistemas operacionais, podem ser usadas futuramente em outros tipos de ataques.

A detecção de um ataque de *Scan* pode auxiliar significativamente administradores de rede na defesa do ambiente. Supondo que, após uma varredura, outro ataque pode ocorrer, o administrador pode elevar sua atenção para tráfegos com a mesma origem do hospedeiro que fez a varredura.

Essa classe de ataque se subdivide em duas categorias:

- ***Scan* de Rede:** trata-se do processo utilizado para descobrir quais hospedeiros estão ativos na rede **(**KINABLE, 2008). Há diversas formas de se realizar este ataque, mas a mais comum consiste em enviar mensagens de solicitação de eco ICMP[8] para os endereços IP disponíveis na rede;

- ***Scan* de Hospedeiro:** consiste em levantar os serviços sendo executados em um hospedeiro específico (LEE; ROEDEL; SILENOK, 2003). Normalmente é executado determinando um intervalo de portas do alvo, ou realizando a varredura em todas as portas disponíveis em um IP de destino.

c) Ataques de DoS (*Denial of Service*)

Os ataques de DoS ou negativa de serviço são utilizados para subverter a disponibilidade de um serviço ou sistema por meio do esgotamento de recursos do alvo (SCHUBA et al., 1997). Estes ataques são executados por atacantes ou *botnets*[9] e têm como principal objetivo tornar os serviços indisponíveis para usuários legítimos ao enviar uma grande quantidade de solicitações de conexão até o servidor se tornar incapaz de responder novas solicitações, legítimas ou não.

---

[8] Protocolo utilizado para enviar informações da camada de rede, como relatórios de erros de comunicação.

[9] Rede formada por centenas de computadores zumbis e que permite potencializar ações danosas executadas por programas maliciosos controlados remotamente (*bots*).

Para que sua chance de sucesso aumente, esse processo é frequentemente executado de forma distribuída, configurando um ataque distribuído de negação de serviço ou *Distributed Denial of Service* (DDoS) (MIRKOVIC; REIHER, 2004).

2.4 *Cyber Situational Awareness* (CSA)

O termo *Situational Awareness* (ou Consciência Situacional, em tradução livre), de forma geral, foi originalmente apresentado por Endsley (1988), que o definiu como "a percepção dos elementos no ambiente dentro de um volume de tempo e espaço, a compreensão de seu significado e a projeção de seu estado no futuro próximo". As palavras "percepção", "compreensão" e "projeção" podem ser usadas para denotar níveis de consciência progressivamente crescentes, começando com (i) percepção básica de dados importantes, (ii) interpretação e combinação de dados em algum tipo de conhecimento, e (iii) a habilidade de prever eventos futuros e suas implicações.

O termo *Cyber Situational Awareness* (ou Consciência Situacional em Cibersegurança, em tradução livre) foi originalmente cunhado por Bass (2000), que propôs a fusão de dados de diversos sensores na rede para gerar um *framework* de detecção de intrusão de próxima geração, de forma a atingir uma Consciência Situacional em Cibersegurança. Franke e Brynielsson (2014) fizeram uma revisão sistemática de literatura e definiram *Cyber Situational Awareness* como a Consciência Situacional no ambiente de redes de computadores, devendo tratar de quaisquer atividades suspeitas ocorrendo nesse ambiente.

Dessa forma, o modelo de *Cyber Situational Awareness* deve compreender três estágios, estabelecidos a partir dos níveis de *Situational Awareness* apresentados por Endsley:

- **Percepção:** provê o conhecimento de ocorrências na rede, que pode ser obtido por meio de alertas reportados, com registros do momento em que ocorreram;
- **Compreensão:** utiliza técnicas, metodologias e processos para analisar e agregar dados de evidências percebidas na etapa anterior. Neste momento, deve ocorrer a visualização de eventos identificados como

relevantes, de forma significativa e que faça sentido no contexto abordado;

- **Projeção:** permite a realização de previsões futuras sobre eventos na rede baseadas no conhecimento extraído das dinâmicas dos elementos do ambiente e da compreensão das ocorrências.

2.5 *Threat Intelligence* e *Cyber Threat Intelligence*

O termo *Threat Intelligence* refere-se ao processo de compreensão de informações, de maneira precisa, relevante e em tempo suficiente para que as informações possam ser usadas de forma acionável (ABU et al., 2018; DE MELO E SILVA, 2020; FRIEDMAN; BOUCHARD, 2015). Assim, o processo de *Threat Intelligence* é capaz de prover auxílio no desenvolvimento de mecanismos de resposta a ameaças, o que diz respeito a um tipo de inteligência acionável.

O conceito de *Cyber Threat Intelligence* pode ser considerado uma inteligência acionável construída com base em evidências de incidentes no ambiente cibernético, e provê conhecimento sobre métodos que podem ajudar na tomada de decisão para resposta a ameaças (DE MELO E SILVA, 2020).

Assim, fica nítido o relacionamento entre *Cyber Threat Intelligence* e *Cyber Situational Awareness*, pois ambas buscam obter conhecimento sobre o estado de uma rede e sobre eventos de segurança nesse ambiente de maneira holística, auxiliando da forma mais clara possível na mitigação desses eventos.

2.5.1 *Feeds* de *Cyber Threat Intelligence*

Os *feeds* de *Cyber Threat Intelligence* são arquivos de dados que fornecem informações atualizadas sobre ameaças de segurança. Eles contêm uma lista de indicadores de comprometimento (*Indicators of Compromise* – IoCs) que incluem, dentre outras informações, URLs e endereços IP envolvidos em ataques e atividades maliciosas.

Como são frequentemente atualizados, e são publicados por empresas de segurança conhecidas, esses *feeds* proveem informações valiosas para auxiliar na detecção de novas ameaças em potencial e de hospedeiros comprometidos em uma rede, uma vez que esses hospedeiros internos não deveriam, em situações normais, estar se comunicando com as URLs e IPs listados em algum desses *feeds*.

O cruzamento dos *feeds* de *Cyber Threat Intelligence* com os fluxos de uma rede permite identificar de forma eficiente essas comunicações maliciosas e, portanto, é uma maneira de executar os dois primeiros estágios de *Cyber Situational Awareness* apresentados na seção 2.4.

2.6 Assinaturas de Detecção de Eventos

Assinaturas de detecção são estruturas que possibilitam a identificação de eventos que ocorrem em hospedeiros ou na rede. Os eventos identificados pelas assinaturas podem ser eventos quaisquer, não necessariamente maliciosos. Em segurança de redes de computadores, assinaturas são frequentemente utilizadas para detectar ataques e eventos de segurança e, nesse contexto, uma assinatura é criada por meio da observação de características específicas no tráfego gerado pela atividade maliciosa analisada (AFEK; BREMLER-BARR; LANDAU FEIBISH, 2013).

A criação de assinaturas requer análise de tráfego malicioso em busca de padrões. Isso pode ser feito de duas formas distintas: por meio da análise de *payload*[10] ou de fluxos de rede. Por ter acesso a dados mais refinados, a abordagem com inspeção de *payload* é mais precisa; no entanto, necessita de acesso aos dados dos usuários da rede, o que pode acarretar problemas com relação a políticas de privacidade ou ao se utilizar criptografia ponto a ponto, além de ter maior custo computacional para processamento (BOUKHTOUTA et al., 2016).

As assinaturas criadas por meio da análise de fluxo de rede se utilizam dos atributos presentes nos fluxos (endereços IP de origem e destino, portas de transporte de origem e destino, entre outros) e de cálculos e comparações realizados sobre esses

[10] *Payload* é uma referência ao conteúdo dos pacotes de rede; é um termo bastante utilizado na literatura sobre detecção de intrusão.

atributos para identificar um evento de segurança no ambiente monitorado (UMER; SHER; BI, 2017).

Dessa forma, o uso de fluxos possibilita observar características em comum contidas nos pacotes sem utilizar o campo de dados deles e, portanto, não infringe a privacidade do usuário, estando em conformidade com o Marco Civil da *Internet* (BRASIL, 2014). Além disso, sua análise exige menor custo computacional e a utilização de criptografia ponto a ponto não interfere no estudo dos dados (UMER; SHER; BI, 2017). A desvantagem do uso de fluxos é que eles carregam informações mais limitadas do que os *payloads* e, portanto, as assinaturas geradas por meio deles precisam ser bastante específicas para serem eficazes.

2.7 *Elastic Stack*

A *Elastic Stack*, anteriormente conhecida como ELK *Stack*, é um conjunto de produtos *Open Source* que se integram para criar um sistema de armazenamento, indexação, recuperação e visualização de informações ágil e funcional (ELASTIC, 2021f). Neste trabalho, os produtos da *Elastic Stack* utilizados foram o *Elasticsearch*, o *Filebeat* e o *Kibana*.

2.7.1 *Elasticsearch*

O *Elasticsearch* é um mecanismo de busca e um banco de dados distribuído NoSQL[11] orientado a documentos capaz de trabalhar com *terabytes* de dados. De um modo geral, o *Elasticsearch* implementa uma camada de serviços acima da biblioteca de busca e indexação de texto *Apache Lucene* (THE APACHE SOFTWARE FOUNDATION, 2021b), tornando-se assim um intermediário entre a aplicação que faz acesso aos dados armazenados e a API[12] de busca e indexação. Escalabilidade é o

[11] Um banco de dados NoSQL é não relacional, ou seja, faz o armazenamento e recuperação de dados de forma diferente das relações tabulares presentes nos bancos relacionais.
[12] *Application Programming Interface* refere-se a um conjunto de rotinas para acesso a um *software*.

foco de sua arquitetura, que possibilita elaborar um *cluster*[13] de nós (ou *nodes*, instâncias do *Elasticsearch*) e realizar a comunicação entre os mesmos.

Quando comparado com outros bancos de dados NoSQL, o *Elasticsearch* apresenta melhor desempenho em operações de leitura do que os demais (ABUBAKAR; ADEYI; AUTA, 2014), principalmente quando se tem volumes de documentos consideráveis a serem indexados e consultados.

No contexto do *Elasticsearch*, um index é uma estrutura equivalente a uma tabela nos bancos relacionais; um documento é equivalente a uma linha da tabela, e os campos dos documentos são comparáveis às colunas das tabelas. No entanto, diferentemente dos bancos relacionais, no *Elasticsearch* é feito um mapeamento automático dos campos, de forma que, caso seja inserido um documento cujos campos diferem dos que já haviam sido indexados, o mapeamento é refeito automaticamente, de forma que o documento seja indexado com os novos campos, sem afetar os documentos já armazenados anteriormente.

2.7.2 *Kibana*

O *Kibana* é uma interface de visualização de dados capaz de funcionar em conjunto com um servidor *Elasticsearch* (ELASTIC, 2021b). Nessa interface, é possível realizar consultas aos dados indexados e refinar os resultados graficamente por meio de opções ou da sintaxe definida para a interface, a *Kibana Query Language* (ELASTIC, 2021c). Além disso, o *Kibana* permite filtrar os resultados mostrados por períodos de tempo, possibilitando visualizar tanto eventos quase em tempo real, bem como eventos que ocorreram em janelas de tempo passadas (limitando-se aos dados armazenados no *Elasticsearch*).

A principal utilidade do *Kibana* é fornecer *templates* de gráficos, que podem ser mais simples, como os de setores, barras, linhas, etc., ou mais sofisticados, como mapas de calor, mapas de regiões geográficas, tabelas de dados, etc. A ferramenta também permite agregar todos os gráficos em um *dashboard* (ELASTIC, 2021a) que possui um objetivo ou um tema, possibilitando uma visão de vários gráficos diferentes sobre um mesmo tipo de evento.

---

[13] *Cluster* é um conjunto de computadores trabalhando de forma a compor um único sistema.

2.7.3 *Filebeat*

Trata-se de uma ferramenta de agregação que é usada para enviar dados de logs para o *Elasticsearch*, estruturando-os em um padrão pré-definido (ELASTIC, 2021d). O *Filebeat* é um dos sete *Beats* existentes, família de agentes responsáveis por exportar diferentes tipos de dados para o *Elasticsearch*. Uma das grandes vantagens do uso do *Filebeat* (e dos *Beats* em geral) é sua facilidade de configuração e seu baixo consumo de recursos.

O *Filebeat* possui um conjunto de módulos que permitem definir tipos de arquivos de log que serão agregados e enviados ao *Elasticsearch* para serem mostrados no *Kibana*. Um módulo é composto das seguintes definições:

- Configurações de entrada do *Filebeat*, que contêm os caminhos padrão para a localização dos arquivos de log. Esses caminhos variam de acordo com o Sistema Operacional analisado;
- Definição do *pipeline* de ingestão do *Elasticsearch*, usado para analisar as linhas de logs e obter as informações relevantes;
- Definições de campos, usadas para configurar o *Elasticsearch* com os tipos de dados corretos para cada campo;
- *Dashboards* padrão do *Kibana*, que podem ser usados para visualizar os arquivos de log ingeridos.

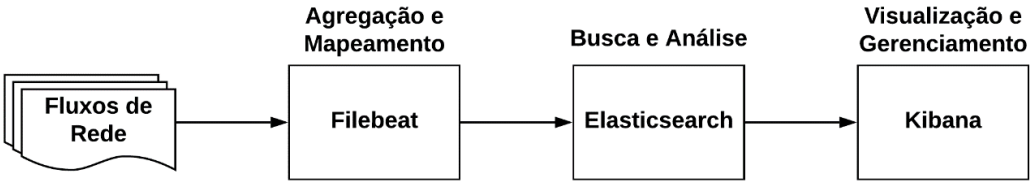
No contexto de fluxos, o *Filebeat* possui um módulo de *Netflow* que simplifica a coleta, normalização, armazenamento e visualização de dados de fluxo *Netflow* (ELASTIC, 2021e), facilitando seu tratamento. Este módulo suporta a versão 9 do *Netflow*, mapeando os campos dos fluxos automaticamente para esse protocolo.

2.7.4 Funcionamento da *Elastic Stack*

A *Elastic Stack* integra as três ferramentas descritas nas seções 2.7.1 a 2.7.3 de forma que cada uma forneça para a próxima os dados necessários para seu funcionamento. O *Filebeat* é a ferramenta agregadora, que coleta todos os dados de interesse (fluxos de rede), formata-os segundo o mapeamento estabelecido, e os

encaminha para o *Elasticsearch*. Este, por sua vez, faz a análise e permite que sejam feitas buscas nos dados, além de cálculos mais complexos com os campos dos documentos indexados. O conjunto de dados passados ao *Elasticsearch*, bem como os resultados das buscas efetuadas por ele, são mostrados em gráficos no *Kibana*, que também atua como interface para buscas mais simplificadas em tempo real no *Elasticsearch*. Na Figura 1 apresenta-se um diagrama do funcionamento da *Elastic Stack*.

Figura 1: Funcionamento simplificado da *Elastic Stack*.



Fonte: elaborada pela própria autora.

2.8 MITRE ATT&CK

O MITRE ATT&CK é um *framework* que concentra uma extensa base de conhecimento sobre Ameaças Persistentes Avançadas (APTs). Por meio da enumeração e categorização das principais táticas, técnicas e procedimentos utilizados por essas ameaças, o *framework* garante uma visão geral das diferentes fases de seu ciclo de vida, seus principais alvos e, principalmente, seu comportamento (STROM et al., 2018). Além disso, o *framework* MITRE ATT&CK foi construído de modo que qualquer pessoa ou organização possa utilizá-lo como base para o desenvolvimento de outros modelos de detecção de ameaças, bem como de novos produtos de cibersegurança (MITRE, 2021c).

O MITRE ATT&CK aborda um conjunto de 14 táticas que cobrem os principais objetivos dos atacantes durante uma tentativa de comprometer algum alvo. Essas táticas podem ser alcançadas por meio de diversas técnicas, e os métodos para a aplicação de cada técnica são diversos e evoluem constantemente ao longo do tempo.

A aplicabilidade do MITRE ATT&CK é dada em função de sua completude e nível de detalhamento, além da forma estruturada como o *framework* organiza as informações relacionadas aos diferentes ataques em APTs, o que garante a fundamentação necessária não apenas para identificar comportamentos maliciosos em ambientes comprometidos, mas também a possibilidade de sua utilização por processos de *Cyber Threat Intelligence*. Assim, fica nítida a relação que pode ser feita entre o MITRE ATT&CK e *Cyber Situational Awareness*, pois o primeiro pode ser empregado para padronizar os tipos de eventos detectados na etapa de Percepção de CSA.

CAPÍTULO 3 - Trabalhos Relacionados

3.1 Considerações Iniciais

Neste capítulo são apresentados os trabalhos publicados semelhantes ao sistema apresentado neste trabalho. Eles constituem o estado da arte com relação ao desenvolvimento de modelos e aplicações de *Cyber Situational Awareness* (CSA) e contribuem para corroborar a importância do sistema proposto. Na seção 3.2, são abordados os desafios encontrados na aplicação de modelos de CSA; na seção 3.3, são discutidos os modelos de CSA que correspondem ao estado da arte; e, por fim, na seção 3.4, são apresentadas as conclusões sobre os trabalhos relacionados.

Os trabalhos correlatos apresentados neste capítulo foram encontrados por meio de buscas nas plataformas Google Acadêmico (GOOGLE, 2021), *IEEE Xplore* (IEEE, 2021) e *ScienceDirect* (ELSEVIER, 2021). Os termos combinados para encontrar os resultados adequados foram: (*Situational Awareness* OU *Cyber Situational Awareness* OU *Cybersecurity Situational Awareness*) E (*Network Flow* OU *Netflow* OU *Incident Response* OU *Cyberattack Detection* OU *Visualization*).

3.2 Desafios em *Cyber Situational Awareness*

Em 2019, Li et al. fizeram um estudo sobre os métodos de CSA publicados até aquele momento, de forma a determinar as perspectivas de desenvolvimento da área. Os autores propuseram decompor o processo de CSA em cinco etapas:

aquisição de elementos, representação do modelo, estabelecimento de métricas, análise da solução e projeção situacional. Na primeira etapa, os dados de interesse da rede são classificados e sumarizados. Destaca-se que os tipos de dados coletados devem ser condizentes com as informações que se deseja obter a partir do processamento desses dados; assim, ter muitos tipos diferentes de dados de entrada não significa que a qualidade do resultado final será obrigatoriamente superior.

Na segunda etapa, define-se o modelo que será usado para processar os dados capturados, ou seja, quais tipos de análises serão feitas com os dados. Os autores discutem que não há um modelo considerado melhor em relação aos outros, e que seja mais utilizado e bem aceito; cada modelo já proposto tem suas vantagens e desvantagens, e novos modelos ainda devem ser apresentados.

A terceira etapa consiste em refinar o valor de cada dado capturado na etapa um. Assim, cada potencial ameaça detectada deve ser classificada quanto a sua severidade, frequência e probabilidade de sucesso, de forma a reduzir a ocorrência de informações falsas na etapa de análise subsequente.

A quarta etapa é o núcleo de CSA, e seu objetivo é calcular o estado de segurança da rede na forma de um algoritmo de solução de modelo. Essa etapa usa várias métricas pré-estabelecidas, bem como os valores refinados da etapa três, para determinar o estado atual de segurança da rede. Por fim, a quinta etapa tem o propósito de criar visualizações para o que foi modelado na etapa dois e o que foi calculado na etapa quatro. Além disso, nessa etapa também deve ser aplicado algum tipo de algoritmo de previsão que indique potenciais ameaças futuras com base nos resultados das etapas anteriores.

Li et al. (2019) destacaram, ao final, que a etapa cinco é a que menos foi abordada na literatura, uma vez que a projeção correta de eventos de rede não é trivial, e que precisam ser desenvolvidas soluções para este problema. Outro ponto destacado pelos autores é com relação à aplicação de CSA em *Big Data*: eles discutem que, como a segurança da informação tem se tornado um problema de análise em *Big Data*, é preciso desenvolver soluções capazes de tratar grandes volumes de dados sem a necessidade de recursos computacionais exorbitantes. Como as soluções de CSA desenvolvidas até então são custosas quando aplicadas a volumes de dados massivos, os autores defendem que o estudo de modelos de CSA no contexto de *Big Data* é uma oportunidade para descobertas na área.

Em 2020, Jirsik e Celeda fizeram um novo estudo sobre *Cyber Situational Awareness* propondo melhorias para as técnicas de monitoramento de fluxos para aprimorar a percepção de uma rede de computadores, além de sugerirem uma nova abordagem para o monitoramento de fluxos que permite obter CSA em tempo real. Os autores iniciaram sua discussão destacando a relevância de se utilizar fluxos para obter *Cyber Situational Awareness*, pois, comparado à inspeção de pacotes, o uso de fluxos possibilita processar um volume muito maior de tráfego com velocidade superior, em troca de menor quantidade de informações nos dados. Em seguida, observaram que, como o propósito original dos fluxos quando foram desenvolvidos era prover uma visão holística da rede para emitir relatórios regulares, há três problemas que dificultam seu uso no contexto de CSA.

O primeiro problema se refere à falta de informações sobre as aplicações para o uso efetivo de fluxos em cibersegurança; para isso, eles testaram três algoritmos para obter algumas informações dos protocolos HTTP e HTTPS[14] e exportá-las com os dados do fluxo. O impacto de acrescentar essas informações adicionais foi uma redução de 50% no desempenho do analisador de fluxos exportados, quando comparado com a mesma análise sobre fluxos sem as informações extras.

O segundo problema diz respeito à dificuldade que se tem para identificar hospedeiros apenas utilizando seu endereço IP, uma vez que esses endereços podem ser alterados dinamicamente, e que NATs[15] utilizam apenas um endereço IP público apesar de representarem mais de um hospedeiro. Desta forma, seria necessário utilizar métodos auxiliares para identificar hospedeiros sem utilizar unicamente seu IP, e os autores propuseram associar o endereço IP ao Sistema Operacional do hospedeiro (informação contida em um dos campos do fluxo) para identificá-lo com maior precisão.

O terceiro problema aborda os atrasos existentes nos processos de criação, análise e exportação dos fluxos, o que impede sua utilização para CSA em tempo real. Para tratar deste problema, os autores sugeriram uma arquitetura em que os fluxos coletados são enviados via *Apache Kafka* (THE APACHE SOFTWARE FOUNDATION, 2021a), que é uma plataforma de *stream* distribuído de dados, para

---

[14] HTTP é o protocolo base de comunicação de dados na *Web*. HTTPS é a sua implementação utilizando uma conexão criptografada.

[15] *Network Address Translation* (Tradução de Endereço de Rede) é uma técnica de mapeamento de um conjunto de endereços IP privados para um endereço IP público.

o *Apache Spark* (THE APACHE SOFTWARE FOUNDATION, 2021c), sistema que processa e analisa esses *streams*, e é capaz de processar mais de 500 mil fluxos por segundo. Os resultados da análise do *Apache Spark* são armazenados em outra base de dados, que pode ser visualizada por meio do *Kibana* ou de alguma outra ferramenta de visualização. Os autores propuseram mesclar os resultados dessa análise de *streams* de fluxos com a abordagem tradicional de monitoramento de fluxos exportados para que se tenha ambas as perspectivas da rede.

Também em 2020, Husák, Jirsík e Yang fizeram um estudo sobre problemas e desafios atuais para implementar CSA em segurança de redes. A princípio, os autores estruturaram um histórico sobre a evolução do campo de *Cyber Situational Awareness* e destacaram que houve um aumento anual das publicações na área a partir de 2015, o que corrobora sua relevância. O foco dos autores são os desafios atuais para a CSA, sendo que estes são elencados sob duas perspectivas: dados e ferramentas.

Na perspectiva de dados, destaca-se a dificuldade de se detectar atividades maliciosas no crescente conjunto de informações analisado pelos sistemas de CSA diariamente, pois há cada vez mais ruído nos dados de entrada. Além disso, outro ponto que precisa ser tratado é a homogeneização dos dados recebidos, pois sistemas de CSA normalmente recebem entradas de diversas fontes, e cada uma possui um padrão de exportação diferente. Para que a análise dessas informações possa ocorrer de maneira uniforme, todos os dados de entrada precisam estar no mesmo formato.

Na perspectiva de ferramentas, Husák, Jirsík e Yang apontaram como dificuldade a variedade de ferramentas de detecção normalmente empregadas por sistemas de CSA (tráfego de rede, logs de equipamentos, detecções de IDSs, etc.), as quais frequentemente apresentam dificuldade de interoperabilidade, de forma que o operador precisa integrar suas saídas manualmente. A visualização de eventos em redes de larga escala também é citada como um problema, pois ela deve ser escalável em nível de detalhes e no tempo para que o operador consiga entender informações gerais e detalhadas, o que não é algo facilmente obtido. A última dificuldade listada se refere aos desafios de escalabilidade das ferramentas, que de forma geral apresentam atrasos no processamento e no tempo de resposta, o que dificulta a atuação do operador de CSA, que idealmente precisa de novas informações o mais rápido possível para que possa tomar decisões a tempo de mitigar ataques.

3.3 Modelos de *Cyber Situational Awareness*

Liu et al. (2019) sugeriram um Modelo de Controle de Consciência Cognitivo (*Cognitive Awareness-Control Model* – CACM) para CSA, cujo objetivo é adquirir, compreender, projetar e mostrar as tendências de evolução de ameaças em um ambiente de rede em larga escala.

Os autores destacaram que um dos desafios no campo de CSA é a incapacidade dos modelos anteriores de quebrar as barreiras entre diferentes camadas da rede. Para isso, é necessário um método de consciência de fusão, para que as informações advindas das diferentes camadas possam ser unificadas e interpretadas em conjunto. O modelo sugerido pelos autores propõe uma estrutura *cross-layer*, que faz um *feedback* em *loop* entre as camadas e utiliza um algoritmo de fusão para fazer a integração entre elas. Todo o processo é executado por um mecanismo de regulação cognitiva, que controla o funcionamento do sistema de forma autônoma.

Os testes foram feitos em uma rede experimental utilizando um conjunto de dados de eventos de segurança da DARPA; os autores não entraram em detalhes sobre qual base foi usada, mas sabe-se que a base é composta de arquivos no formato pcap[16] que simula tráfego HTTP, SMTP[17] e DNS[18] entre uma rede e a *Internet*, ao longo de 10 dias. O modelo proposto foi capaz de compreender e projetar as tendências de evolução da situação de segurança da rede de forma geral, utilizando as três métricas propostas no trabalho (Valor da Situação de Segurança dos Hospedeiros, Valor da Situação de Segurança dos Serviços e Valor da Situação de Segurança da Rede), mas não possui a funcionalidade de projetar a ocorrência de ataques e eventos de segurança específicos. Além disso, as visualizações disponibilizadas são apenas gráficos de linha que mostram a evolução dos valores das três métricas propostas em períodos de tempo; não foram desenvolvidos outros tipos de visualização mais instintivos.

Han et al. (2019) propuseram um sistema de CSA chamado YHSAS, que se baseia na aquisição de vários dados de entrada, e usa análise e mineração de dados,

---

[16] O formato pcap é o formato padrão para arquivos de tráfego de rede.
[17] *Simple Mail Transfer Protocol* é o protocolo simples de transferência de correio eletrônico.
[18] *Domain Name System* é o sistema utilizado para tradução de nomes de domínio para endereços IP.

bem como métodos de dedução inteligente, para descobrir incidentes de segurança, determinar seus potenciais de dano e prever seu desenvolvimento.

A arquitetura proposta pelos autores prevê a coleta de diversos tipos de dados de entrada, como arquivos, pacotes, fluxos de rede, sessões, informações de memória dos equipamentos, protocolos e serviços. O sistema suporta armazenar 10 PB de dados, e é capaz de integrar 187 tipos de dispositivos de rede, e, portanto, foi visivelmente desenvolvido para tratar de grandes volumes de dados de entrada. Os dados capturados são armazenados em uma estrutura de grafo[19] de larga escala, também proposta no trabalho. Após a coleta dos dados, é feita a detecção de ataques por meio de um método que combina correspondência de subgrafos com padrões de atividade capturada; o sistema é capaz de detectar atividades de *botnets* e ataques de DDoS lentos.

Depois da detecção dos ataques, é realizado um cálculo para determinar a situação de segurança geral da rede; este cálculo é executado utilizando algoritmos de clusterização. Também são feitas visualizações dos grafos utilizados nas etapas anteriores. Ao final, realiza-se a projeção da tendência de segurança da rede por um período de tempo no futuro, prevendo-se a propagação dos ataques identificados. Essa projeção se utiliza de séries temporais e de *Support Vector Machines* (SVMs)[20].

Han et al. (2019) finalizaram seu trabalho discutindo que o sistema desenvolvido apresentou alto desempenho em tempo real, com uma taxa de acerto na detecção de ataques satisfatória (não fornecem valores concretos) e uma taxa de acerto na projeção de ataques de 90%. Os autores ainda destacaram que o sistema precisa ser aperfeiçoado, uma vez que há muitos fatores a serem considerados para que a detecção e projeção sejam feitas com assertividade.

Cinque, Della Corte e Pecchia (2020) propuseram um método de CSA para filtrar e priorizar a ocorrência de eventos, como *scans* SSH e ataques de DoS e de modificação de dados em logs textuais não estruturados de aplicações de um sistema de informação de Controle de Tráfego Aéreo em um contexto real. O sistema proposto usa um esquema de ponderação de termos com entropia logarítmica para fazer amostragens dos logs em trechos de duração fixa para computar pontuações, as

---

[19] Estrutura matemática que mostra graficamente o relacionamento de elementos (nós) por meio de ligações (arestas).

[20] *Support Vector Machines* são uma categoria de algoritmos de aprendizado de máquina supervisionado.

quais são empregadas para filtrar e ranquear os eventos identificados. Foram analisados logs a uma taxa média de 6.500 linhas por minuto.

Os autores destacaram que a filtragem de eventos permite tratar de grandes volumes de dados ao realizar o reconhecimento de eventos que devem ser verificados por analistas de segurança, pois podem ajudar a revelar ataques e usos irregulares da rede. A saída do sistema consiste em uma lista de eventos ordenados de forma descendente, baseado no valor das pontuações calculadas.

3.3.1 *Elastic Stack* aplicada em modelos de *Cyber Situational Awareness*

Jirsik e Celeda (2018) propuseram um *framework* que fornece *Cyber Situational Awareness* abrangendo a rede inteira e tratando de *Big Data* em tempo real. O modelo sugerido foi idealizado para superar os problemas advindos da aplicação de CSA de forma extensiva em ambientes reais, como o excesso de dados crus de entrada, baixa velocidade de reação, e a falta de contexto e de uma visão unificada da rede.

O *framework* proposto, chamado *Stream4Flow*, foi capaz de receber fluxos de uma rede com 25 mil hospedeiros a uma taxa média de 8.500 fluxos por segundo, empregando o *Elasticsearch* para armazenar os fluxos analisados por meio da biblioteca GraphX (THE APACHE SOFTWARE FOUNDATION, 2021d), que permite criar grafos entre eles e examiná-los. Os resultados obtidos são mostrados no *Kibana*. Ao final, os autores destacaram que o modelo efetuou com sucesso os estágios de Percepção e Compreensão da CSA, apresentando resultados e visualizações precisas, que transmitem informações valiosas sobre o estado da rede. Os autores também evidenciaram que o desempenho do sistema atingiu a proposta de expor dados relevantes em tempo real, mesmo tendo um grande volume de informações de entrada. No entanto, eles apontaram como problema o fato de o terceiro estágio da CSA (Projeção) não ter sido abordado pelo *framework*, e que isso seria tratado no futuro.

Yang et al. (2019) propuseram um sistema que agrega e visualiza dados de fluxos *Netflow* da rede utilizando a *Elastic Stack*, e emprega um modelo de *deep*

learning[21] com Redes Neurais Recorrentes (RNNs – *Recurrent Neural Networks*) para realizar a detecção de ataques. As RNNs são um tipo de rede neural artificial em que a saída de cada camada é acrescentada como entrada para ela própria juntamente ao próximo vetor de entrada, permitindo que a rede aprenda abstrações mais amplas a partir das sequências de entrada. Os autores exportaram os fluxos *Netflow* (não especificam a versão) para a *Elastic Stack* e fizeram uma análise preliminar dos dados utilizando visualizações no *Kibana*, permitindo a filtragem por sub-redes e contabilizando fluxos para os IPs de destino. Depois, a etapa de detecção utilizou um modelo de RNN já treinado para detectar especificamente atividades dos *worms*[22] CodeRed e Nimda, apresentando acurácia de 95%.

Yang et al. (2019) concluíram seu trabalho com a proposta de coletar características de ataques diferentes no futuro, sem especificar quais tipos. No entanto, não mencionaram a utilização de visualizações no *Kibana* para as detecções efetuadas, nem mesmo como melhoria futura para o sistema.

Chaudhari et al. (2020) também utilizaram a *Elastic Stack* para agregar e analisar logs advindos de equipamentos de rede distintos, como *firewalls*[23], roteadores, Sistemas de Detecção de Intrusão[24] (IDSs) e servidores de *email*, DNS e *web*, além de tráfego *Netflow* exportado. A rede analisada no estudo se trata de uma rede de porte médio de uma organização não especificada, com cerca de 2.500 nós. Foram criadas visualizações e *dashboards* no *Kibana* para contabilizar ataques de *bruteforce* nos serviços SSH e *Telnet*[25], acessos a *websites* especificando endereços IP e usuários que os efetuaram, detalhes de hospedeiros infectados por *malwares*[26] e detalhes de tráfego de entrada/saída com IPs e números de porta de origem e destino.

Os autores destacaram a importância de se utilizar a *Elastic Stack* para que fosse possível analisar em tempo real os cerca de 70 milhões de logs semanais gerados na rede, entre logs de sistemas e fluxos exportados. Segundo eles, as filtragens realizadas durante a análise dos logs permitiram reduzir o espaço ocupado

---

[21] Subárea do Aprendizado de Máquina em que as características empregadas no modelo são automaticamente aprendidas a partir do processo de treinamento.
[22] Código malicioso cuja disseminação ocorre de maneira automática, sem a intervenção do usuário.
[23] Sistemas que analisam pacotes e são capazes de bloquear (ou permitir) aqueles que correspondem a regras predeterminadas.
[24] Sistemas que detectam tentativas de intrusão na rede analisando tanto cabeçalhos de pacotes como seu conteúdo.
[25] Protocolo de aplicação que permite o interfaceamento entre processos de terminal (POSTEL; REYNOLDS, 1983).
[26] Programa malicioso desenvolvido para executar ações danosas em um computador.

pelos dados, uma vez que apenas as informações que seriam posteriormente utilizadas para cada tipo de log foram armazenadas. Eles sugeriram como trabalho futuro implementar *scripts* para tomar medidas proativas quando comportamentos maliciosos fossem encontrados, além de efetuar a detecção de anomalias e a predição de comportamentos da rede baseada em Aprendizado de Máquina, de forma a melhorar o gerenciamento de segurança e planejar ações de mitigação.

O *ElastiFlow* (2020) é um sistema que faz a análise de fluxos de rede (*Netflow*, *sFlow*[27] e IPFIX[28]) usando a *Elastic Stack*. Ele recebe apenas fluxos como entrada, e fornece *dashboards* do *Kibana* que permitem visualizar informações sobre tráfego de rede, hospedeiros que mais se comunicaram, bem como ameaças identificadas na rede por meio de enriquecimentos oferecidos por ferramentas de terceiros. Sua primeira versão foi proposta em 2017.

3.4 Conclusões sobre os Trabalhos Relacionados

Na seção 3.2, destacou-se que Jirsik e Celeda (2020) e Husák, Jirsík e Yang (2020) fizeram um levantamento de problemas encontrados ao se utilizar fluxos de rede como entrada para sistemas de CSA. O problema apontado com relação à dificuldade de uso de fluxos em cibersegurança devido às poucas informações presentes neles não se aplica neste trabalho, pois os dados naturalmente contidos nos fluxos são suficientes para realizar a detecção de atividades maliciosas da forma proposta no capítulo 4. O mesmo ocorre com relação à dificuldade que se tem para identificar hospedeiros apenas utilizando seu endereço IP; como os fluxos usados neste trabalho são os do Sistema Autônomo da UNESP (ASN 53166), todos os hospedeiros internos da rede alvo avaliada possuem endereços IP públicos, não havendo, então, dificuldade para identificá-los.

Outro problema elencado pelos autores diz respeito à dificuldade de se detectar atividades maliciosas com o aumento do volume dos dados de entrada, uma vez que sistemas de CSA tendem a receber dados de várias fontes diferentes, e que,

---

[27] Protocolo de exportação de fluxos em que obrigatoriamente se faz amostragem estatística dos fluxos coletados.
[28] Padrão do IETF para exportação de fluxo de rede.

portanto, precisam ser padronizados. Este problema também não é enfrentado pelo sistema proposto neste trabalho, pois, como já foi justificado na seção 1.2, são utilizados apenas fluxos *Netflow* como entrada, reduzindo o volume de informações recebidas e garantindo a homogeneidade de todos os dados. A dificuldade com relação à escalabilidade de detalhes e de tempo que deve existir nas visualizações de CSA é superada pelo sistema proposto neste trabalho por meio de sua própria metodologia, que armazena os eventos detectados separadamente dos fluxos recebidos, e também por meio do uso do *Kibana*, pois essa interface facilita a criação de visualizações flexíveis e interativas, que são editáveis em tempo real com relação ao escopo de tempo e de diversas outras variáveis que podem ser utilizadas como filtros.

O único desafio destacado por Jirsik e Celeda (2020) e Husák, Jirsík e Yang (2020) que se aplica ao sistema proposto neste trabalho é com relação ao atraso que pode ocorrer durante a exportação de fluxos, que impacta na velocidade com que os dados de detecções são disponibilizados para serem analisados pelo operador.

Na Tabela 1 é apresentado um resumo dos modelos de CSA discutidos na seção 3.3. Nela, constam os tipos de dados de entrada utilizados por cada modelo, se ele emprega a *Elastic Stack*, e o método que o modelo usa para realizar a detecção de eventos maliciosos.

Tabela 1: Estado da Arte em modelos de *Cyber Situational Awareness*.

Trabalho	**Tipos de dados de entrada**	***Elastic Stack*?**	**Método de detecção de eventos**
Jirsik e Celeda (2018, 2020)	Fluxos de rede.	Sim.	Análise de *streams* de dados com o *Apache Spark*.
Liu et al. (2019)	Tráfego de aplicações, alertas de IDSs e fluxos de rede.	Não.	Modelo de Controle de Consciência Cognitivo.
Han et al. (2019)	Arquivos, pacotes, *streams*, sessões, etc.	Não.	Correspondência incremental de subgrafos.
Yang et al. (2019)	Fluxos *Netflow*.	Sim.	Modelo de *deep learning* com RNN.
Chaudhari et al. (2020)	Fluxos *Netflow* e logs de equipamentos de rede.	Sim.	Filtros em visualizações do *Kibana*.

Cinque, Della Corte e Pecchia (2020)	Logs textuais não estruturados.	Não.	Filtragem e pontuação de eventos usando entropia logarítmica.
ElastiFlow (2020)	Fluxos de rede (*Netflow*, *sFlow* e IPFIX).	Sim.	Visualizações do *Kibana* com fluxos enriquecidos.

Fonte: elaborada pela própria autora.

Um aspecto que deve ser destacado sobre esses trabalhos são os diferentes tipos de dados de entrada utilizados. Apenas Jirsik e Celeda (2018, 2020), Yang et al. (2019) e *ElastiFlow* (2020) usam apenas fluxos de rede como entrada do modelo; todos os outros trabalhos empregam outros tipos de dados, o que exige um maior custo computacional para homogeneizar e processar as informações recebidas. A principal vantagem de se utilizar apenas fluxos é a otimização de processamento e a consequente redução de custo de *hardware* para executar o modelo, especialmente considerando o volume de dados gerado por uma rede do porte de um Sistema Autônomo.

Com relação à visualização, somente Jirsik e Celeda (2018, 2020), Yang et al. (2019), Chaudhari et al. (2020) e *ElastiFlow* (2020) criaram visualizações elaboradas e fáceis de serem utilizadas por um administrador de rede, com o auxílio do *Kibana*. Liu et al. (2019), Han et al. (2019) e Cinque, Della Corte e Pecchia (2020) ou utilizaram gráficos mais abstratos, cujas interpretação e aplicação são mais complexas, ou não criaram nenhum tipo de visualização para os dados analisados.

Outra discussão importante trata dos dois modelos que são mais semelhantes ao proposto neste trabalho. O primeiro, *Stream4Flow*, proposto por Jirsik e Celeda (2018, 2020), permitia a visualização de características básicas de tráfego da rede inteira, além de particularidades dos hospedeiros da rede, por meio da interface do *Kibana*. Ele também permitia a detecção de alguns eventos de segurança, como ataques de *Scan*, *Bruteforce* SSH e DDoS. No entanto, o projeto deixou de ser mantido em agosto de 2020, e muitas bibliotecas utilizadas pelo sistema se tornaram depreciadas, de forma que o sistema não pode mais ser instalado e utilizado.

O segundo, *ElastiFlow* (2020), era originalmente um projeto *Open Source*, mas desde agosto de 2020 se tornou uma solução comercial, necessitando de uma chave de licença para ser utilizado. Essa nova solução tem apenas os *dashboards* em formato *Open Source*, e a versão *Community* (gratuita) do sistema possui limitações

de desempenho e armazenamento consideráveis, sendo recomendada apenas para ambientes de laboratório e para testes com volume reduzido de fluxos. Além disso, como as análises feitas pelo sistema são (e sempre foram) executadas diretamente nas visualizações do *Kibana*, é altamente complexo criar e editar visualizações, uma vez que é necessário analisar o código de visualizações já existentes para esse propósito.

Diante deste cenário, evidencia-se a contribuição do presente trabalho, dado que este visa o desenvolvimento de um modelo de Consciência Situacional em Cibersegurança (CSA) escalável baseado na *Elastic Stack* para uma rede de grande porte (Sistema Autônomo da UNESP) que seja capaz de criar visualizações contínuas para informações gerais de tráfego de rede e para os eventos de segurança detectados pelo sistema, utilizando uma estrutura de consulta flexível para realizar essa detecção. Além disso, o modelo proposto não exige alto custo computacional para a análise e visualização de dados porque utiliza apenas fluxos *Netflow* como entrada, que são arquivos que ocupam menos espaço e têm uma menor complexidade, e os quais, pela natureza das informações armazenadas por eles, também apresentam a vantagem de não infringir a privacidade dos usuários da rede.

CAPÍTULO 4 - Metodologia

4.1 Considerações Iniciais

Neste capítulo é apresentada a metodologia utilizada pelo modelo proposto de Consciência Situacional em Cibersegurança (CSA) para Sistemas Autônomos utilizando *Netflow*. Na seção 4.2 é apresentada uma visão geral do modelo e suas cinco etapas; na seção 4.3 discute-se a coleta e indexação dos fluxos utilizados como entrada do sistema; a seção 4.4 descreve a etapa de detecção de eventos de segurança; o processo de enriquecimento dos eventos detectados é apresentado na seção 4.5; a seção 4.6 detalha a etapa de visualização dos dados recebidos e gerados pelo modelo; o método de análise desses dados é explicado na seção 4.7; a seção 4.8 trata da persistência dos fluxos recebidos e dos eventos detectados; e, por fim, as seções 4.9 e 4.10 descrevem os procedimentos necessários para realizar a manutenção e a automatização da instalação da *Elastic Stack*.

4.2 Visão geral

Para que um sistema de Consciência Situacional em Cibersegurança cumpra seu objetivo, deve-se, a princípio, ter um conjunto de dados de entrada que serão analisados. Neste trabalho, os dados de entrada são de apenas um tipo, e consistem em fluxos *Netflow* V9 da rede de grande porte analisada, que se trata do Sistema Autônomo da UNESP (ASN 53166). Essa rede possui dois *ranges* /16 IPv4

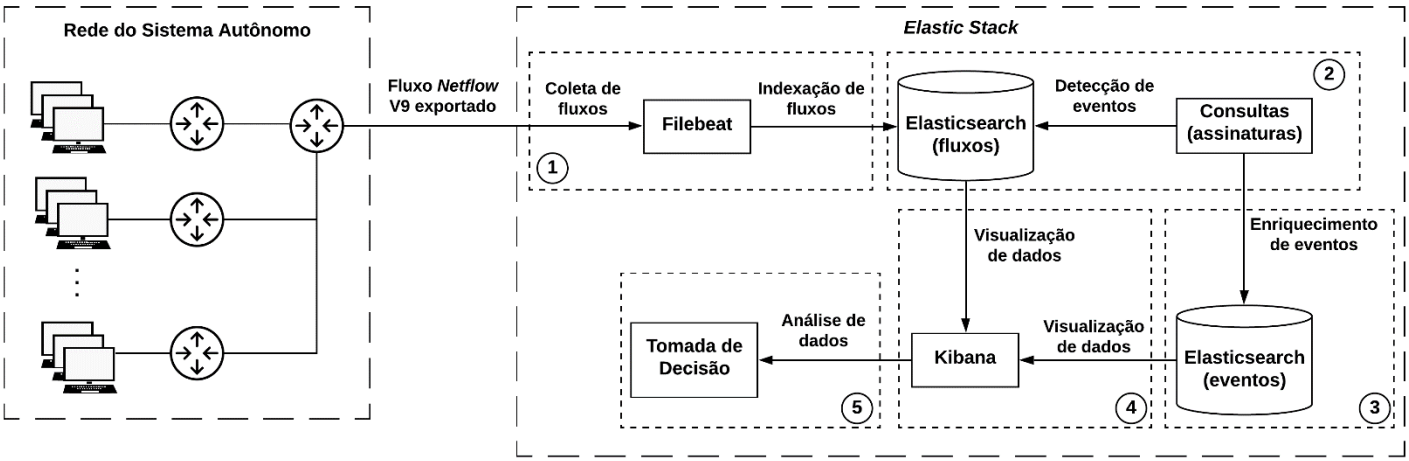
disponíveis, totalizando 131.068 endereços IPv4 possíveis, com uma estimativa de cerca de 70 mil hospedeiros fixamente ativos. Também possui um *range* /32 IPv6 disponível, totalizando mais de 79 octilhões de endereços IPv6 disponíveis.

Esses fluxos são exportados pelo roteador de borda da reitoria da UNESP diretamente para uma máquina da rede interna do Laboratório ACME! de Pesquisa em Cibersegurança de São José do Rio Preto, onde este projeto foi desenvolvido. Os dados exportados pela reitoria da Universidade constituem-se de fluxos *Netflow* V9 diários de todos os 24 *campi* da UNESP e, dessa forma, estão de acordo com os dados de entrada necessários para cumprir a proposta deste trabalho, pois correspondem aos fluxos diários do Sistema Autônomo da UNESP.

O modelo proposto foi implementado usando a *Elastic Stack* para ingerir, processar, enriquecer, armazenar, visualizar e analisar os dados de fluxos. Então, este modelo é composto de cinco etapas: coleta e indexação de fluxos (etapa 1), detecção de eventos (etapa 2), enriquecimento dos eventos detectados (etapa 3), visualização de dados (etapa 4) e análise de dados (etapa 5), como pode ser visto na Figura 2. Cada uma dessas etapas será descrita nas próximas seções.

É importante ressaltar que essas cinco etapas propostas se relacionam diretamente com o primeiro e o segundo estágios de *Cyber Situational Awareness* descritos na seção 2.4: a coleta e indexação de fluxos (etapa 1) e a detecção dos eventos (etapa 2) correspondem ao estágio de Percepção, e o enriquecimento dos eventos (etapa 3), visualização de dados (etapa 4) e análise de dados (etapa 5) equivalem ao estágio de Compreensão. O terceiro estágio da CSA, Projeção, não é abordado neste trabalho devido a sua complexidade intrínseca, como foi apontado na seção 3.2, mas uma perspectiva futura de implementação desse estágio no modelo é tratada na seção 6.4.

Figura 2: Arquitetura do modelo de CSA proposto.



Fonte: elaborada pela própria autora.

4.3 Coleta e indexação de fluxos *Netflow*

Os fluxos utilizados como entrada do sistema são exportados a partir de um roteador de borda pelo qual passa todo o tráfego gerado pelas 24 instituições de ensino da Universidade. Esse tráfego é recebido continuamente pela máquina do Laboratório ACME! onde a *Elastic Stack* está instalada, de forma que cerca de 70 milhões de fluxos *Netflow* são recebidos e armazenados por dia (o que equivale a 50 GB de espaço em disco ocupados diariamente). A taxa de ingestão de fluxos é de cerca de 800 fluxos por segundo, ou 48 mil fluxos por minuto, totalizando aproximadamente 2.880.000 fluxos recebidos em uma hora.

Para este trabalho, a *Elastic Stack* versão 7.8 foi instalada em uma máquina virtual com Ubuntu Server 18.04 (CANONICAL LTD., 2021), com 16 GB de RAM e um HDD com 400 GB de armazenamento disponíveis. O servidor de virtualização utilizado é equipado com um processador Intel Xeon E312xx, com *clock* de 2 GHz e 6 núcleos. Um segundo *node Elasticsearch* foi instalado em outra máquina virtual, com 16 GB de RAM e 200 GB de armazenamento em disco, em outro servidor de virtualização com as mesmas especificações de *hardware* do servidor que hospeda a máquina virtual executando a *Elastic Stack*. Este segundo *node* foi criado para auxiliar no desempenho de consulta e na replicação de dados para o *Elasticsearch*.

Com um armazenamento diário de cerca de 70 milhões de fluxos, o espaço em disco disponível para os *nodes* permite guardar no máximo quatro dias de fluxos recebidos.

Na *Elastic Stack*, o *Filebeat* é o responsável pela coleta de fluxos, tratando-os por meio de seu módulo de *Netflow* (ELASTIC, 2021e). Esse módulo mapeia os campos dos fluxos, executa um enriquecimento inicial, adicionando informações de geolocalização em cada fluxo, e então os envia para serem indexados pelo *Elasticsearch*. Os campos do *template* de mapeamento utilizados pelo módulo de *Netflow* do *Filebeat* são os campos padrão habilitados após sua instalação, com apenas uma modificação para criar campos unificados para os endereços de IPv4 e IPv6, os quais, por padrão, são armazenados em campos separados. Essas modificações foram feitas ao se notar a necessidade de se ter um campo único de endereços IP para facilitar algumas consultas.

4.4 Detecção de eventos de segurança

Uma vez que os fluxos são coletados e indexados, é possível consultá-los para detectar eventos de segurança. A etapa de detecção de eventos é realizada por meio de assinaturas, programadas como consultas ao *Elasticsearch* utilizando a biblioteca Elasticsearch DSL (ELASTICSEARCH DSL, 2021). Trata-se de uma biblioteca construída sobre o cliente em baixo nível oficial para o *Elasticsearch* na linguagem Python 3.8.5 (PYTHON SOFTWARE FOUNDATION, 2021), o Python Elasticsearch Client (ELASTICSEARCH, 2021). O objetivo dessa biblioteca é facilitar a escrita e manipulação de consultas no *Elasticsearch*.

Dessa forma, as assinaturas são escritas em Python 3.8.5, e executadas em outra máquina virtual Ubuntu Server 18.04 na mesma rede da *Elastic Stack*, com 4 GB de RAM. A cada 5 minutos, as assinaturas conectam-se ao *Elasticsearch* e fazem suas consultas para detectar algum dos tipos de eventos de segurança.

É relevante ressaltar que, no *Elasticsearch*, as consultas possuem duas partes principais: a filtragem, que reduz o número de resultados utilizando termos de busca e fazendo operações lógicas entre os termos (E, OU, NÃO), e as agregações, que agrupam os resultados das filtragens segundo algum parâmetro em comum (que pode ser o endereço IP de origem, destino, a porta de destino, ou o tempo, por exemplo).

Quando se faz uma consulta com agregações, é possível visualizar na resposta os *buckets* (ou agrupamentos) resultantes, com os valores sucessivos das chaves utilizadas para as agregações daquele *bucket*. Assim, quando um *bucket* é retornado pela assinatura, tem-se em sua estrutura todas as informações do evento identificado.

Como foi explanado na seção 2.2.2, o protocolo *Netflow* V9 possui 104 campos disponíveis para serem utilizados; no entanto, apenas um subconjunto deles é utilizado pelas assinaturas de detecção definidas. Esses campos, cujos nomes correspondem aos nomes mapeados pelo módulo *Netflow* do *Filebeat*, estão listados na Tabela 2.

Tabela 2: Lista de campos do *Netflow* V9 utilizados pelas assinaturas.

Nome do Campo	**Descrição**
flow_direction	Direção do fluxo (fluxo de entrada = 1, fluxo de saída = 0).
transport	Protocolo de camada de transporte utilizado.
duration	Duração do fluxo em nanosegundos.
timestamp	Data e hora em que o fluxo foi exportado.
tcp_control_bits	Cumulativo das *flags* TCP vistas no fluxo (valor inteiro).
destination_ip	Endereço IPv4 ou IPv6 de destino do fluxo.
source_ip	Endereço IPv4 ou IPv6 de origem do fluxo.
destination_transport_port	Número de porta de destino do fluxo.
source_transport_port	Número de porta de origem do fluxo.
bytes	Total de *bytes* transferidos em ambas as direções.
packets	Total de pacotes transferidos em ambas as direções.

Fonte: elaborada pela própria autora.

Outra observação relevante é que todas as assinaturas fazem sua consulta nos fluxos indexados dos penúltimos 5 minutos. Então, o período passado para o *Elasticsearch* em cada consulta é entre “now-10m” e “now-5m”, sendo “now” uma palavra reservada para se referir ao dia e horário quando a consulta é executada. Isso se deve ao fato de que, em função de limitações de *hardware* do roteador de borda que exporta os fluxos, estes sofrem um atraso para serem exportados para a *Elastic Stack*, de modo que os fluxos de eventos que ocorreram no tempo t são exportados

continuamente durante a janela $(t, t + 5m)$. Assim, esses fluxos só estarão totalmente indexados pelo *Elasticsearch* a partir do tempo $t + 5m$, o que justifica o fato da consulta realizada pelas assinaturas considerar os penúltimos 5 minutos em cada execução. Se o período consultado por elas fosse os últimos 5 minutos, os fluxos dos eventos que ocorreram nesse período ainda não teriam terminado de ser exportados, e consequentemente eles não seriam detectados.

Dessa forma, um evento de segurança que ocorreu às 10h terá seus fluxos exportados até as 10h05, e esses são os horários que ficam registrados nos documentos indexados pelo *Elasticsearch*. Esse evento só será detectado quando a assinatura correspondente consultar os documentos do período entre as 10h e as 10h05, o que só ocorrerá quando essa assinatura for executada às 10h10, porque ela sempre consulta os penúltimos 5 minutos. Portanto, em consequência do atraso da exportação dos fluxos, há um atraso de 10 minutos na detecção dos eventos de segurança na rede.

A escolha do período observado pelas assinaturas ser de 5 minutos se deve justamente ao fato de essa ser a menor janela de tempo que se deve aguardar para que os fluxos de um evento estejam disponíveis para serem consultados. Um tempo menor do que esse (3 minutos, por exemplo), também ocasionaria a não detecção de alguns eventos; já um tempo maior do que esse (janelas de 10 minutos, por exemplo), além de atrasar ainda mais a detecção dos eventos, também reduziria o desempenho das consultas, visto que seria necessário que o *Elasticsearch* considerasse um número maior de documentos para fazer a busca, o que impacta diretamente em um aumento do tempo necessário para que ele obtenha os resultados e do custo de processamento para tal.

Há duas categorias de eventos detectados nesta etapa: os ataques apresentados na seção 2.3, e comunicações de hospedeiros internos com endereços IP envolvidos em atividades maliciosas, obtidos a partir dos *feeds* de *Cyber Threat Intelligence*, apresentados na seção 2.5.1.

4.4.1 Detecção de ataques

Os campos e parâmetros utilizados pelas assinaturas de detecção de ataques foram inicialmente baseados no que foi estabelecido por Corrêa (2009), ex-membro do Laboratório ACME!, em seu trabalho sobre detecção de eventos de rede baseada em fluxos. Além disso, para a construção das assinaturas, foram consultados trabalhos reconhecidos na caracterização dos tipos de ataques (LEE; ROEDEL; SILENOK, 2003; PINKAS; SANDER, 2002; SCHUBA et al., 1997), e foram feitos ataques controlados a hospedeiros na rede do Sistema Autônomo, para que fosse possível visualizar esses eventos em ação, e verificar a atuação das assinaturas com fluxos sabidamente maliciosos.

Para esses ataques controlados, foram selecionados dois alvos aleatoriamente em cada uma das 24 instituições da Universidade, com serviços diferentes em execução em suas portas. Os ataques de *Scan* de Hospedeiro foram executados utilizando o *Nmap* (NMAP, 2021); os de *Bruteforce* foram feitos usando a ferramenta *Hydra* (HAUSER, 2021); e os de DoS/DDoS foram executados utilizando o *Hping* (SANFILIPPO, 2006). Essas ferramentas foram escolhidas por serem ferramentas *Open Source* amplamente utilizadas para realizar esses tipos de ataques, e que podem ser instaladas em *Virtual Private Servers*, a partir dos quais os ataques foram realizados.

Os ataques foram executados a partir de máquinas de três serviços de VPS diferentes, ao longo de cinco dias consecutivos, em momentos aleatórios em cada dia. Os ataques foram diversificados por meio de sua duração, que variou entre 60s, 120s, 180s e 240s, e também utilizando parâmetros diferentes para o *Nmap*, e dicionários de tamanhos diferentes para o *Hydra*. Dessa forma, foi possível simular diferentes tipos de ataques, em momentos variados do dia, vindos de origens heterogêneas. Por meio das observações efetuadas analisando o comportamento desses ataques, foi feito um refinamento nos parâmetros numéricos utilizados pelas assinaturas descritas nesta seção.

A seguir, são descritas as assinaturas estruturadas para os ataques de *Scan* de Hospedeiro, *Bruteforce*, DoS e DDoS, bem como suas táticas e técnicas correspondentes no *framework* MITRE ATT&CK.

a) Detecção de ataques de *Scan* de Hospedeiro

A assinatura de *Scan* de Hospedeiro busca encontrar um grande número de fluxos de um mesmo endereço IP de origem para um mesmo endereço IP de destino, tendo como alvo pelo menos 50 portas diferentes daquele destino, em um período de 5 minutos. Os fluxos devem ter apenas a *flag* SYN ativa, uma vez que buscam somente abrir a conexão com as portas dos hospedeiros de destino. Como o *scanner* tenta abrir conexões para serviços que não existem e o servidor responderá finalizando a conexão, a duração dos fluxos é zero.

Esta assinatura corresponde à técnica T1046 (Escaneamento de Serviço de Rede) da tática de Descoberta do MITRE ATT&CK (MITRE, 2021d).

A construção da assinatura com a estrutura da Elasticsearch DSL é a seguinte:

- Filtros:
    1. Filtrar fluxos que estejam direcionados para dentro da rede (*flow_direction* = 1);
    2. E que usem o protocolo TCP (*transport* = tcp);
    3. E que tenham duração (*duration*) igual a zero;
    4. E que tenham apenas a *flag* SYN ativa (*tcp_control_bits* = 2);
    5. E que estejam no período (*timestamp*) entre now-10m e now-5m.

- Agregações:
    1. Agrupar por IPs de destino (*destination_ip*);
    2. Subagrupar por IPs de origem (*source_ip*);
    3. Subagrupar por períodos de tempo (*timestamp*) de 5 minutos (esta agregação é feita para que o *timestamp* seja retornado na consulta e possa ser acessado);
    4. Subagrupar contabilizando o número de portas de destino (*destination_transport_port*) diferentes; apenas as agregações com 50 portas diferentes ou mais são retornadas.

As informações retornadas pela assinatura que identificam um evento de *Scan* de Hospedeiro são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de destino;
- Endereço IP de origem;
- Número de portas de destino diferentes;
- Número de fluxos do evento.

O algoritmo 1 descreve a assinatura de detecção de ataques de *Scan* de Hospedeiros.

Algoritmo 1 – Detecção de ataques de *Scan* de Hospedeiros

Entrada *IPcluster, usuário, senha*

1: iniciarConexãoServidor(*IPcluster, usuário, senha*)

2: *filtros* ← definirFiltros(*flow_direction = 1,*
transport = TCP,
duration = 0s,
timestamp = (now – 5m, now – 10m),
tcp_control_bits = 2)

3: *consulta* ← busca(*filtros*)

4: *consulta*.agregações(*destination_ip,*
source_ip,
destination_transport_port,
timestamp_interval = 5m)

5: *consulta*.métricas(*número_portas_diferentes* ≥ 50)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

b) Detecção de ataques de *Bruteforce*

A assinatura de *Bruteforce* busca encontrar pelo menos 15 fluxos de um mesmo endereço IP de origem para um mesmo endereço IP de destino, tendo como alvo uma mesma porta, em um período de 5 minutos. Como as tentativas de autenticação consistem em enviar diferentes combinações de usuário e senha, os fluxos possuem um padrão quantidade de *bytes*, pacotes e duração. Os fluxos devem

ter a *flag* SYN ativa, uma vez que buscam abrir a conexão com as portas dos hospedeiros de destino.

Esta assinatura corresponde à técnica T1187 (Autenticação Forçada) da tática de Acesso a Credenciais do MITRE ATT&CK (MITRE, 2021b).

A construção da assinatura com a estrutura da Elasticsearch DSL é a seguinte:

- Filtros:
    1. Filtrar fluxos que estejam direcionados para dentro da rede (*flow_direction* = 1);
    2. E que usem o protocolo TCP (t*ransport* = tcp);
    3. E que tenham no mínimo a *flag* SYN ativa (*tcp_control_bits* mod 4 >= 2);
    4. E que estejam no período (*timestamp*) entre now-10m e now-5m.

- Agregações:
    1. Agrupar por IPs de destino (*destination_ip*);
    2. Subagrupar por IPs de origem (*source_ip*);
    3. Subagrupar por porta de destino (*destination_transport_port*);
    4. Subagrupar contabilizando pelo menos 15 fluxos por períodos de tempo (*timestamp*) de 5 minutos;
    5. Subagrupar calculando a média de *bytes*, pacotes (*packets*) e duração (*duration*) dos fluxos. São retornadas apenas as agregações com:
        - média de *bytes* >= 50 e média de *bytes* <= 2000;
        - E média de pacotes >= 1 e média de pacotes <= 16;
        - E média de duração >= 0,1 segundo e média de duração <= 90 segundos.

As informações retornadas pela assinatura que identificam um evento de *Bruteforce* são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de destino;
- Endereço IP de origem;
- Número da porta de destino;

- Número de fluxos do evento;
- Média de pacotes dos fluxos do evento;
- Média de *bytes* dos fluxos do evento;
- Média de duração dos fluxos do evento.

O algoritmo 2 descreve a assinatura de detecção de ataques de *Bruteforce*.

Algoritmo 2 – Detecção de ataques de *Bruteforce*

Entrada *IPcluster, usuário, senha*

1: iniciarConexãoServidor(*IPcluster, usuário, senha*)

2: *filtros* ← definirFiltros(*flow_direction = 1,*
transport = TCP,
duration = 0s,
timestamp = (now – 5m, now – 10m),
tcp control bits % 4 ≥ 2)

3: *consulta* ← busca(*filtros*)

4: *consulta*.agregações(*destination_ip,*
source_ip,
destination_transport_port,
timestamp_interval = 5m)

5: *consulta*.númeroMínimoDeDocumentos(15)

6: *consulta*.métricas(*média_bytes = entre*(50, 2000),
média_pacotes = entre(1, 16),
média_duração = entre(0.1s, 90s))

7: *resposta* ← *consulta*.executar()

Saída *resposta*

c) Detecção de ataques de DoS/DDoS

Esta assinatura busca encontrar um grupo de fluxos para um mesmo endereço IP de destino e que tenham valores acima do normal para o número de fluxos e para as razões de média de *bytes* por média de duração, média de *bytes* por média de pacotes, e média de duração por média de pacotes, no período dos penúltimos 5

minutos. Esses valores acima do normal se devem à natureza do ataque, uma vez que o objetivo é inundar o alvo de forma que ele se torne incapaz de responder a requisições legítimas.

Para obter esses valores acima do normal, primeiro é feita uma consulta nos 5 minutos antes aos penúltimos 5 minutos para obter os valores de média de *bytes* por fluxo, média de pacotes por fluxo e média de duração por fluxo. Então, são calculadas as razões: média de *bytes* por média de duração, média de *bytes* por média de pacotes, e média de duração por média de pacotes. Isso fornece três razões que são consideradas os valores normais, antes do ataque começar.

Em seguida, são consultados os penúltimos 5 minutos, e são obtidas as mesmas três razões para cada IP de destino na rede interna, e esses valores são comparados com os valores normais. Se o número de fluxos for maior ou igual a 5.000 fluxos; e a razão da média de *bytes* pela média de duração for menor ou igual ao valor normal dessa razão dividido por 50; e a razão da média de *bytes* pela média de pacotes for menor ou igual ao valor normal dessa razão dividido por 10; e a razão da média de duração pela média de pacotes for maior ou igual ao valor normal dessa razão multiplicado por 2; então aquele IP de destino sofreu um ataque de DoS ou DDoS nos penúltimos 5 minutos. Essas proporções entre os valores das razões foram determinadas de forma experimental, observando-se os padrões e comportamentos dos ataques reais realizados.

O próximo passo da assinatura é determinar se trata-se de um ataque de DoS ou de DDoS, e isso é possível analisando o número de endereços IP de origem diferentes que enviaram pelo menos 1.000 fluxos para cada hospedeiro atacado, para a mesma porta de destino, nos penúltimos 5 minutos. Se há apenas um endereço IP de origem, então trata-se de um ataque de DoS. Caso haja mais de um endereço IP de origem, então trata-se de um ataque de DDoS.

Esta assinatura corresponde à técnica T1499.002 (Negação de serviço: exaustão de serviço) da tática de Impacto do MITRE ATT&CK (MITRE, 2021a).

As informações retornadas pela assinatura que identificam um evento de DoS/DDoS são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de destino;
- Endereço(s) IP de origem;

- Número da porta de destino;
- Número de fluxos do evento;
- Soma de *bytes* dos fluxos do evento;
- Soma de pacotes dos fluxos do evento.

O algoritmo 3 descreve a assinatura de detecção de ataques de DoS/DDoS.

Algoritmo 3 – Detecção de ataques de DoS/DDoS

Entrada *IPcluster, usuário, senha*

1: iniciarConexãoServidor(*IPcluster, usuário, senha*)

2: *filtros1* ← definirFiltros(*flow_direction = 1,*
transport = TCP,
timestamp = (now – 10m, now – 15m))

3: *consulta1* ← busca(*filtros1*)

4: *consulta1*.métricas(*média_bytes,*
média_pacotes,
média_duração)

5: *resposta1* ← *consulta1*.executar()

6: *filtros2* ← definirFiltros(*flow_direction = 1,*
transport = TCP,
timestamp = (now – 5m, now – 10m))

7: *consulta2* ← busca(*filtros2*)

8: *consulta2*.agregações(*destination_ip*)

9: *consulta2*.métricas(*média_bytes,*
média_pacotes,
média_duração,
número_fluxos)

10: *filtros3* ← definirFiltros(*flow_direction = 1,*
transport = TCP,
timestamp = (now – 5m, now – 10m))

11: *consulta3* ← busca(*filtros3*)

12: *consulta3*.agregações(*destination_ip,*
source_ip,
destination_transport_port)

13: *consulta3*.métricas(*soma_bytes*,
soma_pacotes,
número_fluxos)

14: *valores_ fluxos* ← *consulta2*.executar()

15: *dados_fluxos* ← *consulta3*.executar()

16: *valores_ fluxos.bytes_duração* ← *valores_ fluxos.média_bytes* / *valores_ fluxos.média_duração*

17: *valores_ fluxos.bytes_pacotes* ← *valores_ fluxos.média_bytes* / *valores_ fluxos.média_pacotes*

18: *valores_ fluxos.duração_pacotes* ← *valores_ fluxos.média_duração* / *valores_ fluxos.média_pacotes*

19: *média_bytes_normal* ← *resposta1.média_bytes*

20: *média_duração_normal* ← *resposta1.média_duração*

21: *média_pacotes_normal* ← *resposta1.média_pacotes*

22: *razão_bytes_duração_normal* ← *média_bytes_normal* / *média_duração_normal*

23: *razão_bytes_pacotes_normal* ← *média_bytes_normal* / *média_pacotes_normal*

24: *razão_duração_pacotes_normal* ← *média_duração_normal* / *média_pacotes_normal*

25: *lista_eventos* ← *valores_ fluxos*.onde((*valores_ fluxos.número_fluxos* ≥ 5000) AND (*valores_ fluxos.bytes_duração* ≤ *razão_bytes_duração_normal* / 50) AND (*valores_ fluxos.bytes_pacotes* ≤ *razão_bytes_pacotes_normal* / 10) AND (*valores_ fluxos.duração_pacotes* ≥ *razão_duração_pacotes_normal* * 2))

26: *fluxos_eventos* ← *dados_fluxos*.onde(*dados_fluxos.destination_ip* = *lista_eventos.destination_ip*)

27: *fluxos_eventos* ← *fluxos_eventos*.onde(*fluxos_eventos.número_fluxos* ≥ 1000)

28: *fluxos_eventos* ← *fluxos_eventos*.agruparPor(*destination_ip*, *destination_transport_port*)

29: *lista_dos* ← selecionar *fluxos_eventos* em que há único *destination_ip* AND *source_ip*

30: *lista_ddos* ← selecionar *fluxos_eventos* em que um *destination_ip* tem mais de um *source_ip*

Saída *lista_dos*, *lista_ddos*

4.4.2 Detecção de comunicações com hospedeiros maliciosos

Esta categoria trata dos eventos identificados a partir de dados de *feeds* de *Cyber Threat Intelligence*. Os *feeds* utilizados nesta etapa são *Open Source*, disponibilizados gratuitamente por empresas e organizações de cibersegurança, e são atualizados pelo menos uma vez por semana. Então, para manter esses *feeds* sempre atualizados para as assinaturas os analisarem, um *script* é executado duas vezes por dia para baixar a versão atual dos *feeds* nos endereços em que são disponibilizados.

O processo geral de detecção consiste em parsear cada *feed* para gerar uma lista de hospedeiros externos maliciosos, e então os endereços IP dessa lista são buscados nos fluxos indexados a cada 5 minutos. Fluxos encontrados contendo algum dos IPs maliciosos listados são retornados como eventos de comunicação maliciosa.

Há cinco tipos de eventos detectados nesta etapa: comunicações de hospedeiros externos comprometidos com hospedeiros da rede interna, comunicações de endereços da rede interna com mineradores externos, comunicações com hospedeiros externos envolvidos em atividades de Comando e Controle, comunicações com endereços externos listados em uma *blacklist*, e comunicações com hospedeiros externos envolvidos em atividades de *malware*. As assinaturas utilizadas para identificar cada um desses eventos são descritas a seguir. Como essas assinaturas detectam apenas comunicações potencialmente maliciosas, não há táticas e técnicas do *framework* MITRE ATT&CK associadas a elas.

a) Detecção de comunicações advindas de hospedeiros comprometidos

Para identificar esses eventos, foi utilizado o *feed* “emerging-compromised.rules” (PROOFPOINT INC., 2021b), disponibilizado pela Proofpoint. Este *feed* possui uma lista de hospedeiros hostis ou comprometidos.

O processo de detecção consiste em obter a lista dos endereços IP comprometidos e seus números de porta do arquivo do *feed*, e então buscar por fluxos nos penúltimos 5 minutos em que o IP de origem e a porta de origem

correspondam a algum endereço e porta contidos nessa lista. Portanto, apenas fluxos de entrada são pesquisados.

As informações retornadas por essa assinatura que identificam um evento de comunicação advinda de um hospedeiro comprometido são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de origem;
- Endereço IP de destino;
- Número da porta de origem;
- Número da porta de destino;
- Protocolo da camada de transporte.

O algoritmo 4 descreve a assinatura de detecção de comunicações advindas de hospedeiros comprometidos.

Algoritmo 4 – Detecção de comunicações advindas de hospedeiros externos comprometidos

Entrada *IPcluster, usuário, senha, feed_compromised*

1: *lista_ips_portas* ← lista(*feed_compromised*)

2: iniciarConexãoServidor(*IPcluster, usuário, senha*)

3: *filtros* ← definirFiltros(*flow_direction = 1,*
transport = TCP OR *UDP,*
source_ip = lista_ips_portas.ips,
source_transport_port = lista_ips_portas.portas,
timestamp = (now – 5m, now – 10m))

4: *consulta* ← busca(*filtros*)

5: *consulta*.agregações(*source_ip,*
destination_ip,
source_transport_port,
destination_transport_port,
timestamp_interval = 5m)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

b) Detecção de comunicações com mineradores

Para identificar esses eventos, foi utilizado o *feed* Threatlist Miner (INTERNET STORM CENTER, 2021), disponibilizado pelo Internet Storm Center da SANS. Este *feed* possui uma lista de endereços IP envolvidos em *pools* de mineração de criptomoedas.

O processo de detecção consiste em obter a lista dos endereços IP mineradores do arquivo do *feed*, e então buscar por fluxos nos penúltimos 5 minutos em que o IP de destino corresponda a algum endereço contido nessa lista. Portanto, apenas fluxos de saída são pesquisados.

As informações retornadas por essa assinatura que identificam um evento de comunicação com um hospedeiro minerador são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de origem;
- Endereço IP de destino;
- Número da porta de origem;
- Número da porta de destino;
- Protocolo da camada de transporte.

O algoritmo 5 descreve a assinatura de detecção de comunicações com hospedeiros mineradores.

Algoritmo 5 – Detecção de comunicações com hospedeiros externos mineradores

Entrada *IPcluster, usuário, senha, feed_threatlist_miner*

1: *lista_ips* ← lista(*feed_threatlist_miner*)

2: iniciarConexãoServidor(*IPcluster, usuário, senha*)

3: *filtros* ← definirFiltros(*flow_direction = 0,*
transport = TCP OR *UDP,*
destination_ip = lista_ips,
timestamp = (now – 5m, now – 10m))

4: *consulta* ← busca(*filtros*)

5: *consulta*.agregações(*destination_ip,*

source_ip,
destination_transport_port
source_transport_port,
timestamp_interval = 5m)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

---

c) Detecção de comunicações com hospedeiros de Comando e Controle

Para identificar esses eventos, foram utilizados dois *feeds*: o *feed* "emerging-botcc.rules" (PROOFPOINT INC., 2021a), disponibilizado pela Proofpoint, e o *feed* de Comando e Controle da Threatview (THREATVIEW, 2021). Ambos possuem listas de endereços IP envolvidos em atividades de Comando e Controle[29], mas o primeiro tem também números de porta de destino para as comunicações.

O processo de detecção consiste em obter uma lista com os endereços IP e portas dos arquivos de *feed*, e então buscar por fluxos nos penúltimos 5 minutos em que o IP e porta de destino correspondam a algum endereço e porta contidos nessa lista. Portanto, apenas fluxos de saída são pesquisados.

As informações retornadas por essa assinatura que identificam um evento de comunicação com um hospedeiro de Comando e Controle são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de origem;
- Endereço IP de destino;
- Número da porta de origem;
- Número da porta de destino;
- Protocolo da camada de transporte.

O algoritmo 6 descreve a assinatura de detecção de comunicações com hospedeiros envolvidos em atividades de Comando e Controle.

---

[29] Técnicas usadas por atacantes para se comunicarem com sistemas sob seu controle em uma rede comprometida.

Algoritmo 6 – Detecção de comunicações com hospedeiros externos de Comando e Controle

Entrada *IPcluster, usuário, senha, feed_botcc, feed_threatview*

1: *lista_ips_portas* ← lista(*feed_botcc, feed_threatview*)

2: iniciarConexãoServidor(*IPcluster, usuário, senha*)

3: *filtros* ← definirFiltros(*flow_direction = 0,*
transport = TCP OR *UDP,*
destination_ip = lista_ips_portas.ips,
destination_transport_port = lista_ips_portas.portas,
timestamp = (now – 5m, now – 10m))

4: *consulta* ← busca(*filtros*)

5: *consulta*.agregações(*destination_ip,*
source_ip,
destination_transport_port
source_transport_port,
timestamp_interval = 5m)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

d) Detecção de comunicações com hospedeiros em uma *blacklist*

Para identificar esses eventos, foi utilizado o *feed* de *blacklist* publicado pela AbuseIPDB (ABUSEIPDB, 2021). Este *feed* possui uma lista de endereços IP envolvidos em atividades maliciosas, como envio de *spam*, tentativas de fraudes, ataques de DDoS, entre outros, e é desenvolvido de forma colaborativa. É relevante ressaltar que este *feed* é obtido por meio de uma consulta à API da AbuseIPDB, e são requisitados apenas os 400 IPs com pontuação de confiança de 90% ou mais. Essa pontuação é atribuída pela AbuseIPDB para reduzir o impacto que relatos incorretos podem causar na sua base de dados.

O processo de detecção consiste em obter uma lista dos endereços IP do arquivo do *feed*, e então buscar por fluxos nos penúltimos 5 minutos em que o IP de destino corresponda a algum endereço contido nessa lista. Portanto, apenas fluxos de saída são pesquisados.

As informações retornadas por essa assinatura que identificam um evento de comunicação com um hospedeiro em uma *blacklist* são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de origem;
- Endereço IP de destino;
- Número da porta de origem;
- Número da porta de destino;
- Protocolo da camada de transporte.

O algoritmo 7 descreve a assinatura de detecção de comunicações com hospedeiros em uma *blacklist*.

Algoritmo 7 – Detecção de comunicações com hospedeiros externos em uma *blacklist*

Entrada *IPcluster, usuário, senha, feed_blacklist*

1: *lista_ips* ← lista(*feed_blacklist*)

2: iniciarConexãoServidor(*IPcluster, usuário, senha*)

3: *filtros* ← definirFiltros(*flow_direction = 0,*
transport = TCP OR *UDP,*
destination_ip = lista_ips,
timestamp = (now – 5m, now – 10m))

4: *consulta* ← busca(*filtros*)

5: *consulta*.agregações(*destination_ip,*
source_ip,
destination_transport_port
source_transport_port,
timestamp_interval = 5m)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

e) Detecção de comunicações com *Malwares*

Para identificar esses eventos, foi utilizado o *feed* MalSilo (MALSILO, 2021), que consta na lista de *feeds* padrão disponibilizados pelo MISP (MISP PROJECT, 2021). Este *feed* possui uma lista de endereços IP envolvidos em atividades de *malware*, a porta de destino, e o nome do *malware* utilizado.

O processo de detecção consiste em obter uma lista dos endereços IP e portas do arquivo do *feed*, e então buscar por fluxos nos penúltimos 5 minutos em que o IP e a porta de destino correspondam a algum endereço contido nessa lista. Portanto, apenas fluxos de saída são pesquisados.

As informações retornadas por essa assinatura que identificam um evento de comunicação com um hospedeiro envolvido em atividade de *malware* são:

- *Timestamp* (dia e hora) do evento;
- Endereço IP de origem;
- Endereço IP de destino;
- Número da porta de origem;
- Número da porta de destino;
- Nome do *malware* utilizado;
- Protocolo da camada de transporte.

O algoritmo 8 descreve a assinatura de detecção de comunicações com hospedeiros envolvidos em atividades de *malware*.

Algoritmo 8 – Detecção de comunicações com hospedeiros externos envolvidos em atividades de *malware*

Entrada *IPcluster, usuário, senha, feed_malsilo*

1: *lista_ips_portas* ← lista(*feed_malsilo*)

2: iniciarConexãoServidor(*IPcluster, usuário, senha*)

3: *filtros* ← definirFiltros(*flow_direction = 0,*
transport = TCP OR *UDP,*
destination_ip = lista_ips_portas.ips,
destination_transport_port = lista_ips_portas.portas,
timestamp = (now – 5m, now – 10m))

4: *consulta* ← busca(*filtros*)

5: *consulta*.agregações(*destination_ip*,
source_ip,
destination_transport_port
source_transport_port,
timestamp_interval = *5m*)

6: *resposta* ← *consulta*.executar()

Saída *resposta*

4.5 Enriquecimento dos eventos detectados

Após a execução das assinaturas de detecção dos eventos maliciosos, tem-se, para cada categoria de eventos, um conjunto de endereços IP de origem e destino envolvidos nos eventos detectados (além de outras informações específicas para cada tipo de evento). Antes de realizar a inserção desses eventos em seus respectivos indexes no *Elasticsearch*, esta etapa adiciona mais algumas informações a eles.

Como foi mencionado na seção 4.3, o *Filebeat* executa um enriquecimento inicial dos fluxos recebidos, adicionando algumas informações de geolocalização a eles, como nome da organização de origem/destino e código do país de origem/destino. Assim, a etapa inicial do enriquecimento consiste na realização de uma nova consulta ao *Elasticsearch*.

Originalmente, considerou-se obter essas informações por meio da API do *Shodan* (SHODAN, 2021), que permite que os usuários façam consultas ilimitadas para obter esses dados, desde que não sejam utilizados filtros. No entanto, como muitos hospedeiros externos não possuem dados inseridos nesse motor de busca, e como as informações de geolocalização já estão presentes no index de fluxos original, o método escolhido foi fazer uma nova consulta ao *Elasticsearch*.

Essa consulta de enriquecimento busca pelo endereço IP externo listado em cada evento de segurança detectado para obter seu Número de Sistema Autônomo (ASN), nome da organização, e código do país. Os campos utilizados nessa busca estão listados na Tabela 3.

Tabela 3: Campos utilizados para a consulta de enriquecimento dos eventos.

Nome do Campo	**Descrição**
source_as_number	Número de Sistema Autônomo de origem do fluxo.
destination_as_number	Número de Sistema Autônomo de destino do fluxo.
source_country_iso_code	Código do país de origem do fluxo.
destination_country_iso_code	Código do país de destino do fluxo.
source_organization_name	Nome da organização de origem do fluxo.
destination_organization_name	Nome da organização de destino do fluxo.

Fonte: elaborada pela própria autora.

O algoritmo 9 descreve a consulta de enriquecimento dos endereços externos dos eventos. Esse algoritmo considera que alguma das assinaturas apresentadas na seção 4.4 foi executada, e portanto recebe como entrada uma lista de IPs externos envolvidos nos eventos detectados.

Algoritmo 9 – Consulta para enriquecimento dos IPs externos

Entrada *IPcluster, usuário, senha, lista_ips_externos*

1: iniciarConexãoServidor(*IPcluster, usuário, senha*)

2: *ips_externos_enriquecidos* ← *lista_ips_externos*

3: **Para cada** *ip* em *lista_ips_externos* **faça**

4: *filtros* ← definirFiltros(*flow_direction = 0, destination_ip = ip)*

5: *consulta* ← obterPrimeiroDocumento(*filtros*)

6: *resposta* ← *consulta*.executar()

7: *ips_externos_enriquecidos[ip].as_number* ← *resposta.destination_as_number*

8: *ips_externos_enriquecidos[ip].country_iso_code* ← *resposta.destination_country_iso_code*

9: *ips_externos_enriquecidos[ip].organization_name* ← *resposta.destination_organization_name*

Saída *ips_externos_enriquecidos*

O algoritmo 9 é usado quando o IP externo é o destino do evento detectado. Nos casos em que o IP externo é a origem do evento, basta alterar o valor de *flow_direction* para 1, e os campos de *destination* para *source*.

Após a obtenção do Número de Sistema Autônomo, nome da organização e código do país para o IP externo, a próxima tarefa na etapa de enriquecimento consiste em descobrir o *hostname*[30] para cada endereço IP de origem e destino envolvidos nos eventos detectados. Os *hostnames* são informações relevantes para ajudar um administrador de rede a identificar se o evento detectado é de fato malicioso.

Para essa tarefa, listas dos IPs internos e externos envolvidos nos eventos detectados são usadas como entrada para o algoritmo 10, que realiza uma chamada para uma função que faz uma consulta DNS reversa para obter os *hostnames* dos endereços IP.

Algoritmo 10 – Consulta para obtenção de *hostnames*

Entrada *lista_ips_externos, lista_ips_internos*

1: *ips_hostnames* ← *lista_ips_externos, lista_ips_internos*

2: **Para cada** *ip* em *ips_hostnames* **faça**

3: *ips_hostnames[ip].hostname* ← obterHostnamePeloEndereço(*ip*)

Saída *ips_hostnames*

Assim, ao final da etapa de enriquecimento de eventos, as seguintes informações são adicionadas a cada evento de segurança detectado:

- Número de Sistema Autônomo do hospedeiro externo;
- Código do país do hospedeiro externo;
- Nome da organização do hospedeiro externo;
- *Hostname* do hospedeiro externo;
- *Hostname* do hospedeiro interno.

[30] Nome de domínio que identifica um servidor na *Internet*, construído por meio da combinação do nome local do hospedeiro com o nome de domínio pai.

Esses eventos enriquecidos são então inseridos nos indexes correspondentes do *Elasticsearch*, de forma que cada tipo de evento apresentado na seção 4.4 tenha seu próprio index para ser usado na próxima etapa do modelo. Assim, além do index que armazena os fluxos recebidos, o *Elasticsearch* possui também oito indexes de eventos de segurança.

4.6 Visualização de dados

Esta etapa consiste em construir visualizações no *Kibana* utilizando os fluxos *Netflow* recebidos e os indexes de eventos detectados enriquecidos criados na seção 4.5. Visualizações (ELASTIC, 2021a) são gráficos, tabelas, métricas, buscas e mapas criados para facilitar o entendimento dos dados recebidos e gerados pelo modelo.

O *Kibana* permite que o usuário escolha o tipo de visualização que deseja criar e o index cujos documentos serão usados pela visualização. Além disso, é possível realizar buscas em cada visualização por meio da *Kibana Query Language*, que é uma sintaxe desenvolvida para filtrar os documentos indexados pelo *Elasticsearch*.

Para este modelo, foram criadas 187 visualizações, para mostrar dados de tráfego gerais (métricas de fluxos, *bytes* e endereços, número de países, cidades e Sistemas Autônomos nos fluxos, dados sobre pares de IPs trocando fluxos), e também para listar e obter detalhes sobre os eventos detectados (tabelas de eventos de cada tipo, endereços externos e internos envolvidos, entre outras).

4.7 Análise de dados

Uma das funções mais relevantes do *Kibana* é a criação de *dashboards*, que permitem organizar um conjunto de visualizações de forma que traga mais clareza para os dados analisados, e permita tomar decisões que não seriam possíveis sem essa visão multifacetada. A organização das visualizações descritas na seção 4.6 em *dashboards* é o que define esta etapa do modelo.

Para este trabalho, foram criados 20 *dashboards*, sendo que cada um deles tem uma temática específica. É relevante ressaltar que 11 desses *dashboards*

mostram informações sobre os fluxos do Sistema Autônomo inteiro, e nove deles mostram dados apenas do conjunto de endereços do *campus* de São José do Rio Preto, instituição onde essa pesquisa foi desenvolvida.

Os 11 *dashboards* criados para tratar de todos os fluxos do AS são:

- *Overview;*
- Geolocalização;
- HTTP/HTTPS;
- *Scan*;
- *Bruteforce;*
- DoS-DDoS;
- *Compromised;*
- *Miners;*
- C2;
- *Blacklisted;*
- *Malware.*

Cada um deles mostra informações gerais de tráfego, ou de algum evento de segurança apresentado na seção 4.4.

Os nove *dashboards* criados para tratar dos fluxos específicos do *campus* de São José do Rio Preto são:

- *Overview*;
- *Scan Attacks*;
- *Bruteforce Attacks*;
- DoS-DDoS *Attacks;*
- *Compromised;*
- *Miners;*
- C2;
- *Blacklisted;*
- *Malware.*

Esses nove *dashboards* correspondem aos *dashboards* que foram criados para mostrar informações dos fluxos totais do AS, com exceção dos de Geolocalização e

HTTP/HTTPS, pois algumas visualizações destes dois foram incorporadas ao *dashboard* de *Overview* para reduzir o número total de *dashboards* para a instituição.

Detalhes dos componentes de cada *dashboard*, bem como as observações que foram feitas a partir deles, estão descritos no capítulo 5.

4.8 Persistência dos fluxos recebidos e eventos detectados

Como foi apontado na seção 4.3, com os recursos de armazenamento disponíveis atualmente, o sistema pode guardar apenas os fluxos recebidos nos últimos quatro dias, pois são armazenados cerca de 70 milhões de fluxos por dia. Por outro lado, como os eventos de segurança detectados são documentos bastante simples, e os indexes que armazenam esses eventos recebem, cada um, no máximo 200 mil novos documentos por dia, a limitação de armazenamento que se aplica aos fluxos recebidos do roteador de borda não se aplica aos eventos de segurança identificados.

Então, o espaço ocupado pelos indexes de eventos de segurança é consideravelmente inferior ao ocupado pelos fluxos (são da ordem de centenas de *megabytes*), o que permite facilmente armazenar todos os eventos detectados no Sistema Autônomo inteiro no último ano. Isso permite fazer uma análise dos eventos de segurança detectados em períodos de tempo arbitrários, que podem variar de semanas a meses, o que não pode ser feito com os fluxos recebidos, que possuem uma limitação histórica de apenas quatro dias.

4.9 Manutenção da *Elastic Stack*

Para que o espaço em disco disponível para o armazenamento dos fluxos recebidos pudesse ser reciclado, criou-se um *script* de manutenção com o objetivo de apagar o(s) index(es) mais antigo(s) diariamente, de forma que apenas os quatro mais recentes permaneçam.

Essa manutenção é de suma importância, uma vez que, caso 90% ou mais do disco esteja ocupado, o *Elasticsearch* entra em estado de apenas leitura, e novos fluxos recebidos passam a ser descartados.

4.10 Automatização da instalação da *Elastic Stack* via *Ansible*

O fato da *Elastic Stack* receber atualizações continuamente faz com que seja necessário reinstalá-la com frequência para que se possa usufruir dos novos recursos disponibilizados, bem como para verificar a compatibilidade dos módulos e das bibliotecas utilizados pelas assinaturas com as novas versões da pilha. No entanto, como a pilha possui muitos elementos que precisam ser instalados e configurados para que o conjunto funcione adequadamente, o processo de instalação pode ser bastante trabalhoso, principalmente em novos ambientes.

Para superar essa dificuldade, foi desenvolvido um *playbook* em *Ansible* (RED HAT, 2021) que realiza o processo de instalação automaticamente, após serem configurados alguns parâmetros do ambiente. O *Ansible* é uma ferramenta *Open Source* para provisionamento, instalação e gerenciamento de configurações de *softwares*. Ele funciona com arquivos que descrevem quais ações devem ser executadas em uma máquina, que são chamadas de *roles*. Um conjunto de *roles* dá origem a um *playbook*.

Assim, após configurar o *range* de IPv4 e IPv6 da rede interna, a interface de rede utilizada pela máquina, e mais algumas configurações específicas da *Stack*, basta executar o *playbook* para que todos os elementos da pilha sejam instalados e configurados para se comunicarem. Ao final do processo, tem-se a pilha pronta para receber os fluxos *Netflow*, faltando apenas copiar os executáveis das assinaturas e o *script* de manutenção, e importar os *dashboards*. Esse processo foi testado em uma máquina virtual reserva na rede do laboratório, e a instalação e configuração da *Elastic Stack* versão 7.13 (a mais recente naquele momento) foi feita com sucesso.

CAPÍTULO 5 - Resultados

5.1 Considerações Iniciais

Neste capítulo são apresentados os resultados obtidos com a metodologia proposta neste trabalho. Na seção 5.2, são apresentados os *dashboards* construídos na etapa de análise de dados; nas seções 5.3, 5.4 e 5.5, descrevem-se os resultados observados por meio desses *dashboards* produzidos; na seção 5.6, é feita uma avaliação sobre os períodos de tempo necessários para executar as assinaturas; ao final, na seção 5.7, são feitas considerações sobre os resultados obtidos.

Os resultados descritos neste capítulo foram obtidos considerando um período de sete dias de observações do sistema, tendo como entrada os fluxos exportados pelo roteador de borda do Sistema Autônomo da UNESP. Assim, trata-se de resultados obtidos em um contexto totalmente real. Como há uma limitação de quatro dias para o armazenamento dos fluxos recebidos (mas não dos eventos detectados, como foi discutido na seção 4.8), os valores de tráfego apresentados na seção 5.3 são médias do que foi observado diariamente sobre os fluxos recebidos ao longo desses sete dias.

5.2 *Dashboards* construídos

Nesta seção, descreve-se a composição dos 11 *dashboards* criados para mostrar informações dos fluxos totais do Sistema Autônomo. Os nove *dashboards*

que foram estruturados para apresentar os dados dos fluxos específicos do *campus* de São José do Rio Preto são apenas particularizações dos primeiros, e portanto são compostos das mesmas visualizações.

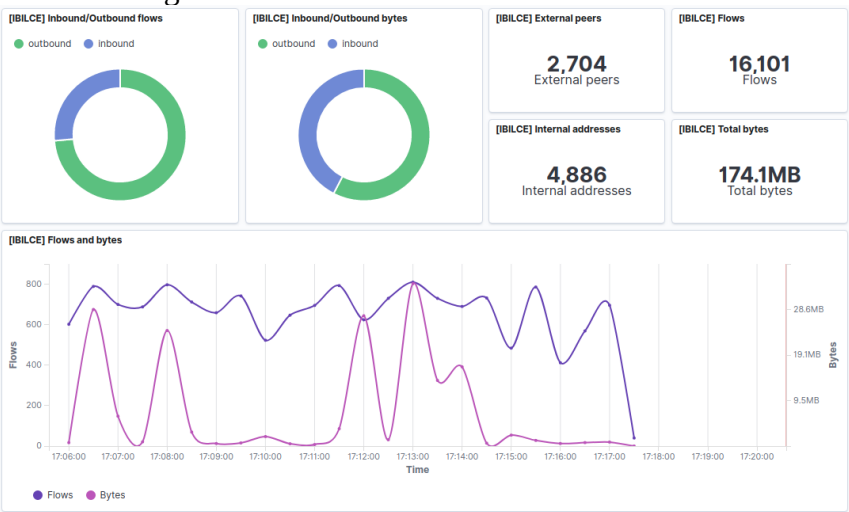
5.2.1 *Dashboard* de *Overview*

Este *dashboard* mostra informações de tráfego gerais para a rede, nos últimos 15 minutos (esse período de tempo é configurável para todos os *dashboards*). Algumas das visualizações nesse *dashboard* incluem:

- Tabela mostrando os endereços que mais trocaram fluxos, *bytes* e pacotes;
- Gráfico de setores comparando *bytes* e fluxos de entrada e saída na rede, e comparando a utilização de versões de protocolos da camada de rede e da camada de transporte;
- Métricas contabilizando o número de hospedeiros externos e internos, e a soma de fluxos e *bytes* trocados;
- Gráfico de linha mostrando a evolução do número de fluxos e *bytes* no período.

Na Figura 3 é apresentado um trecho do *dashboard* de *Overview*.

Figura 3: Trecho do *dashboard* de *Overview*.



Fonte: elaborada pela própria autora.

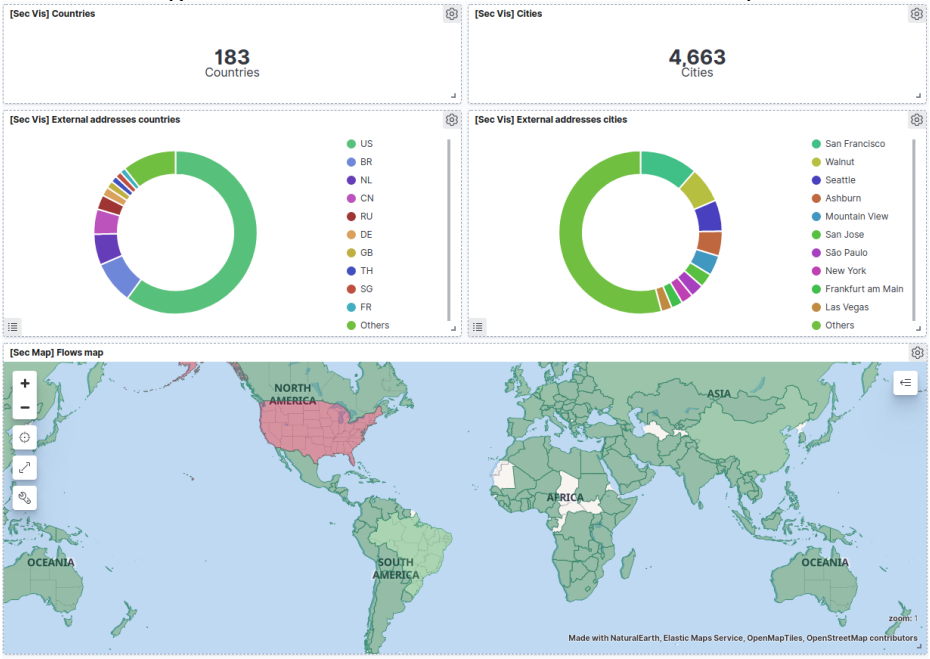
5.2.2 *Dashboard* de Geolocalização

Mostra visualizações relativas aos dados de geolocalização do tráfego da rede nos últimos 15 minutos. Algumas de suas visualizações incluem:

- Métricas contabilizando o número de países, cidades e ASs nos fluxos;
- Gráficos de setores para os países, cidades e ASs que mais estiveram presentes nos fluxos;
- Mapas indicando os países que mais se comunicaram com a rede analisada, considerando fluxos e *bytes*.

Na Figura 4 é apresentado um trecho do *dashboard* de Geolocalização.

Figura 4: Trecho do *dashboard* de Geolocalização.



Fonte: elaborada pela própria autora.

5.2.3 *Dashboard* de HTTP/HTTPS

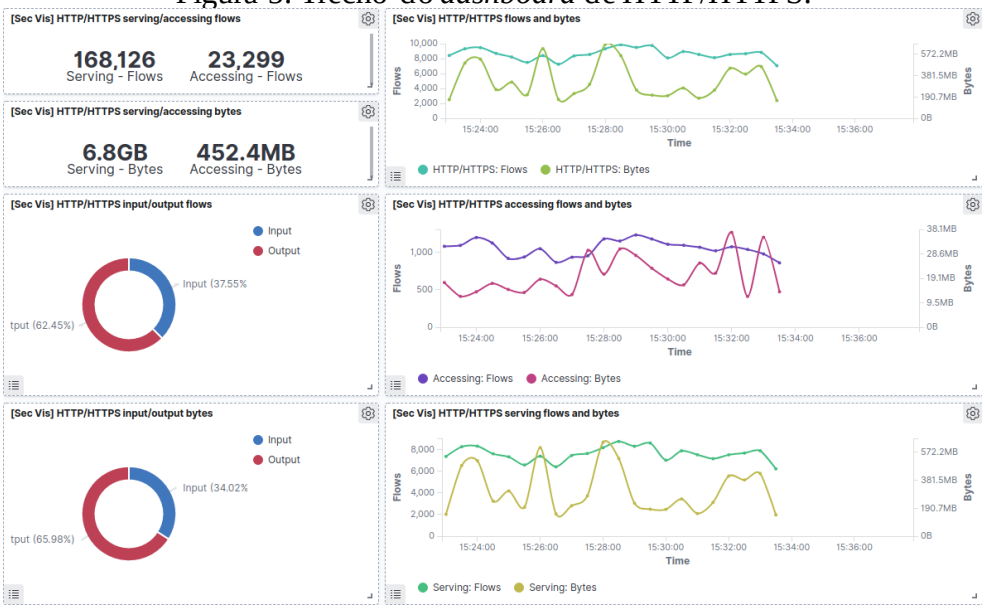
Trata-se de um *dashboard* com dados de fluxos que utilizaram esses protocolos, nos últimos 15 minutos. Foi criado considerando que o tráfego

HTTP/HTTPS é um dos mais volumosos em uma rede, uma vez que a maioria dos serviços comuns utilizados pelos usuários roda sobre HTTP ou HTTPS. Algumas de suas visualizações incluem:

- Métricas contabilizando o número de fluxos e *bytes* servindo ou acessando serviços HTTP/HTTPS;
- Gráficos de setores comparando o volume de fluxos e *bytes* servindo ou acessando serviços HTTP/HTTPS;
- Gráficos de linhas mostrando a evolução do volume de fluxos e *bytes* totais, servindo e acessando esses serviços no período;
- Tabelas mostrando os endereços que mais trocaram fluxos, *bytes* e pacotes usando esses protocolos;
- Mapa mostrando a localização dos países que mais trocaram *bytes* usando esses protocolos.

Na Figura 5 é apresentado um trecho do *dashboard* de HTTP/HTTPS.

Figura 5: Trecho do *dashboard* de HTTP/HTTPS.



Fonte: elaborada pela própria autora.

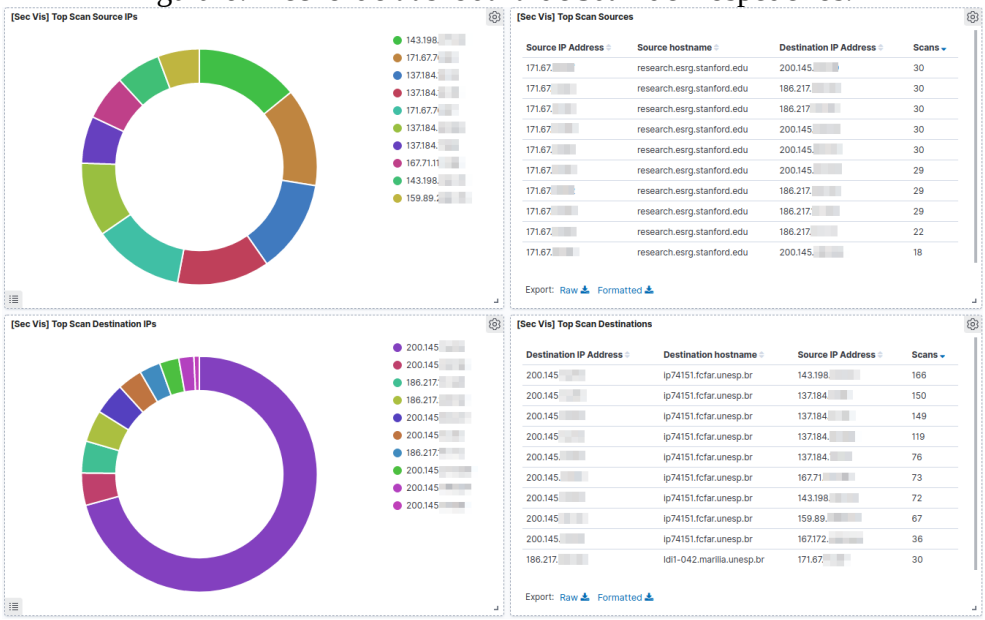
5.2.4 *Dashboard* de eventos de *Scan* de Hospedeiros

Este *dashboard* mostra informações sobre os eventos de *Scan* de Hospedeiros detectados na última semana por meio da assinatura descrita pelo algoritmo 1. Algumas de suas visualizações incluem:

- Lista dos eventos detectados, ordenados dos mais recentes para os mais antigos;
- Tabela com os pares de IPs com mais eventos detectados no período;
- Tabelas e gráficos de setores com os destinos e origens que mais apareceram nos eventos;
- Métricas contabilizando o número de eventos detectados no período, o número de origens e destinos diferentes envolvidos, média de portas escaneadas, e número máximo e mínimo de portas escaneadas;
- Gráficos de setores com os países e organizações dos IPs externos que mais executaram ataques de *scan*;
- Tabelas com os *hostnames* internos e externos que mais se envolveram em eventos.

Na Figura 6 é apresentado um trecho do *dashboard* de *Scan* de Hospedeiros.

Figura 6: Trecho do *dashboard* de *Scan* de Hospedeiros.



Fonte: elaborada pela própria autora.

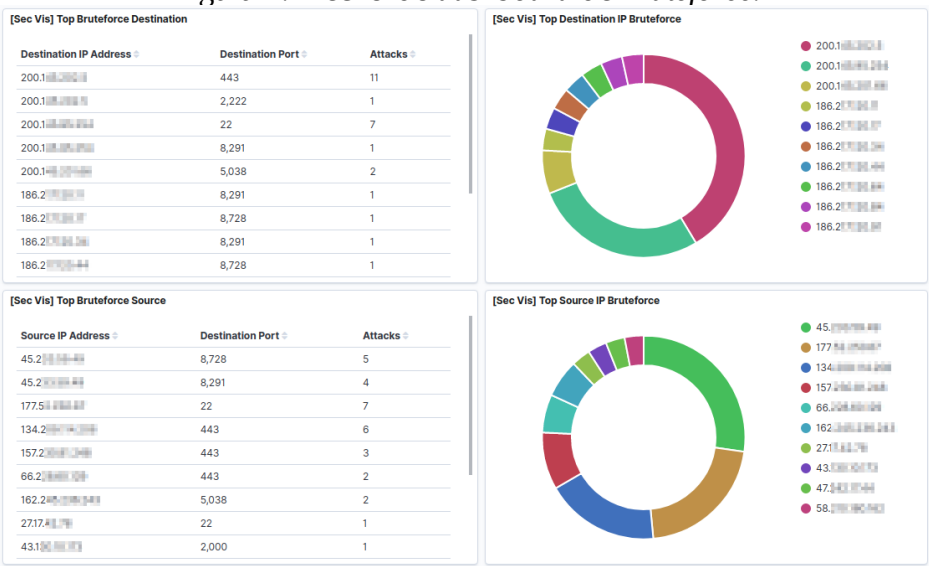
5.2.5 *Dashboard* de eventos de *Bruteforce*

Mostra informações sobre os eventos de *Bruteforce* detectados nas últimas 24 horas por meio da assinatura apresentada no algoritmo 2. Algumas de suas visualizações incluem:

- Lista dos eventos detectados, ordenados dos mais recentes para os mais antigos;
- Tabela com os pares de IPs com mais eventos detectados no período;
- Tabelas e gráficos de setores com os destinos e origens que mais apareceram nos eventos;
- Métricas contabilizando o número de eventos detectados no período, e o número de origens e destinos diferentes envolvidos;
- Gráficos de setores com os países e organizações dos IPs externos que mais executaram ataques de *bruteforce*;
- Tabela com os *hostnames* internos e externos que mais se envolveram em eventos;
- Gráfico de barras com a média de tentativas por ataque nas portas de destino mais atacadas.

Na Figura 7 é apresentado um trecho do *dashboard* de *Bruteforce*.

Figura 7: Trecho do *dashboard* de *Bruteforce*.



Fonte: elaborada pela própria autora.

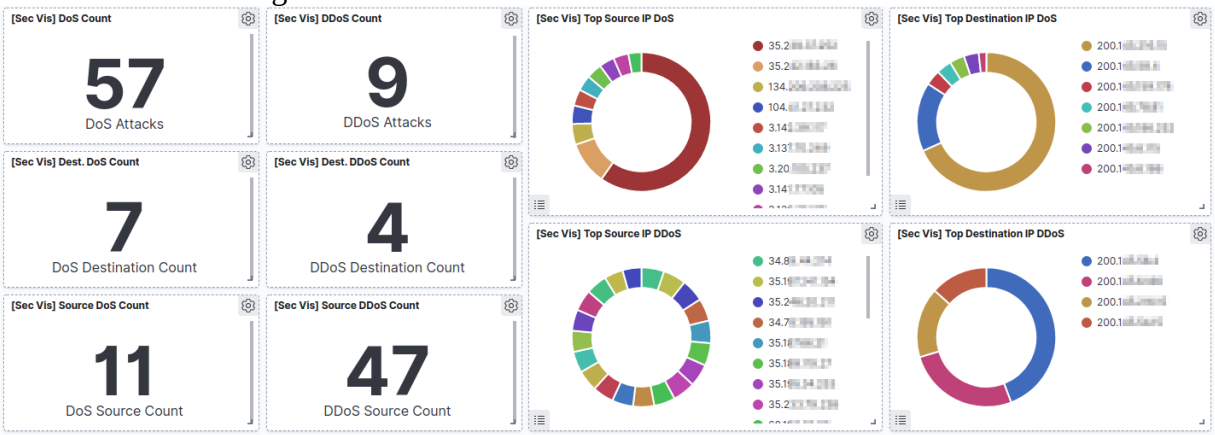
5.2.6 *Dashboard* de DoS/DDoS

Este *dashboard* possui informações sobre os eventos de DoS/DDoS detectados no último ano utilizando a assinatura descrita pelo algoritmo 3. Algumas de suas visualizações incluem:

- Lista dos eventos de DoS e de DDoS detectados, ordenados dos mais recentes para os mais antigos;
- Métricas contabilizando o número de eventos detectados no período, e o número de origens e destinos diferentes envolvidos;
- Gráficos de setores com os destinos e origens mais envolvidos nos ataques de DoS e de DDoS detectados, e com as portas mais atacadas.

Na Figura 8 é apresentado um trecho do *dashboard* de DoS/DDoS.

Figura 8: Trecho do *dashboard* de DoS/DDoS.



Fonte: elaborada pela própria autora.

5.2.7 *Dashboards* de *feeds* de *Cyber Threat Intelligence*

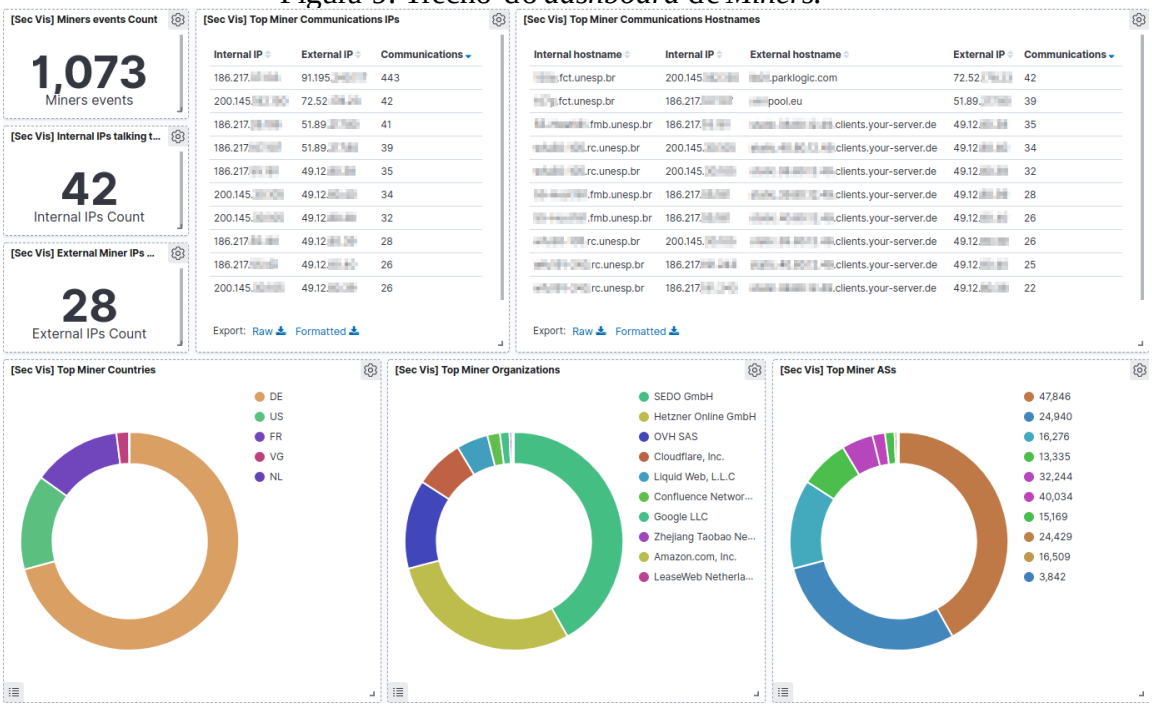
Os *dashboards* que mostram os eventos de comunicações advindas de hospedeiros comprometidos, comunicações com mineradores, comunicações com hospedeiros de Comando e Controle, comunicações com hospedeiros em uma *blacklist* e comunicações com *malwares* possuem a mesma estrutura, utilizando os mesmos tipos de visualizações. Todos eles apresentam informações sobre os eventos

detectados nas últimas 24 horas por meio das assinaturas descritas pelos algoritmos 4 a 8. Algumas de suas visualizações incluem:

- Lista dos eventos detectados, ordenados dos mais recentes para os mais antigos;
- Tabela com os pares de IPs com mais eventos detectados no período;
- Tabelas e gráficos de setores com os destinos e origens que mais apareceram nos eventos;
- Métricas contabilizando o número de eventos detectados no período, e o número de origens e destinos diferentes envolvidos;
- Gráficos de setores com os países e organizações dos IPs externos que mais participaram dos eventos detectados;
- Tabela com os *hostnames* internos e externos que mais se envolveram em eventos.

Na Figura 9 é apresentado um trecho do *dashboard* de *Miners*.

Figura 9: Trecho do *dashboard* de *Miners*.



Fonte: elaborada pela própria autora.

5.3 Resultados apresentados pelos *dashboards* de tráfego

Os *dashboards* de tráfego (*Overview*, Geolocalização e HTTP/HTTPS), apesar de apresentarem informações mais gerais e que não têm interpretações tão diretas quanto os *dashboards* de eventos, mostraram-se de grande valia no período analisado. Eles são os únicos que permitem uma visualização rápida da situação atual da rede, ajudando a identificar anomalias que talvez não se classifiquem dentro de nenhum dos tipos de eventos com assinaturas definidas.

É por meio desses *dashboards* que é possível detectar, por exemplo, picos de transferências entre hospedeiros, pares de IPs com número de fluxos ou *bytes* trocados muito acima do normal, ou até mesmo o uso de protocolos de camada de transporte incomuns. Essas informações são extremamente relevantes quando se quer descobrir causas de problemas de desempenho da rede ou de comportamentos incomuns.

Outro ponto para o qual esses *dashboards* contribuem é a determinação do perfil da rede analisada. É de excepcional relevância que um administrador saiba qual é o perfil de funcionamento normal da rede para que ele seja capaz de notar situações anômalas, que podem indicar tanto eventos maliciosos quanto falhas de serviços e erros de configuração.

Alguns dos dados que permitem traçar esse perfil são: o número de hospedeiros internos normalmente ativos, número de fluxos e volume de *bytes* trocados em um dia, organizações que mais trocam tráfego com a rede, entre outros. Todas essas informações são disponibilizadas pelos *dashboards* de tráfego mencionados.

No período de sete dias analisado, foi possível identificar que, diariamente, o AS da UNESP possui cerca de 130 mil endereços IPv4 internos e 400 endereços IPv6 internos ativos, conversando com cerca de 1,7 milhão de hospedeiros externos diferentes. Além disso, são trocados cerca de 70 milhões de fluxos, totalizando mais de 700 GB de tráfego. Participam das comunicações diárias mais de 170 países, quase 4.000 cidades diferentes, e mais de 3.000 ASs, sendo que o país que, sozinho, hospeda mais de 50% dos IPs externos que participam dessas comunicações é os Estados Unidos.

5.4 Avaliação dos eventos de segurança detectados

Esta seção trata dos resultados apresentados especificamente pelas assinaturas de detecção de eventos de segurança, e foram obtidos por meio da observação de seus *dashboards*. A princípio, são feitas algumas considerações sobre os eventos de forma generalizada, e em seguida analisam-se os tipos de eventos que apresentaram os resultados mais expressivos no período examinado.

5.4.1 Observações gerais

Durante os sete dias analisados, pôde-se observar uma diferença considerável no número de eventos detectados pelos oito tipos de assinaturas. Na Tabela 4 é feita uma comparação desses valores.

Tabela 4: Número de eventos detectados pelas assinaturas ao longo de sete dias.

Assinatura	**Número de eventos detectados**
Scan de Hospedeiro	506
Bruteforce	8.132
DoS/DDoS	0
Comunicação advinda de hospedeiros comprometidos	896.008
Comunicação com mineradores	13.178
Comunicação com hospedeiros de Comando e Controle	36
Comunicação com hospedeiros em *blacklist*	206.648
Comunicação com *malwares*	4

Fonte: elaborada pela própria autora.

A assinatura de *Scan* de Hospedeiros detectou um número baixo de eventos quando comparada a outras que tiveram detecções na casa dos milhares de eventos. Como se trata de um evento de segurança que faz apenas um reconhecimento dos

alvos, e portanto não traz resultados diretos e imediatos para o atacante, é compreensível que haja menos eventos desse tipo.

A assinatura de *Bruteforce*, por outro lado, detectou mais de 8 mil eventos no período. Por se tratar de um ataque simples de ser executado, que pode ser facilmente programado usando ferramentas *Open Source*, e considerando que as portas alvo teoricamente foram descobertas por meio dos ataques de *Scan*, é concebível que este seja um ataque com alta incidência de eventos.

Não foram identificados ataques de DoS/DDoS no período, o que pode se dever ao aumento nas proteções que os provedores de VPS têm implementado para barrar esse tipo de ataque sendo executado a partir de uma máquina em sua nuvem. Como máquinas em VPSs são uma opção bastante fácil para lançar ataques de negativa de serviço (este inclusive foi o método empregado para testar a assinatura de detecção, como descrito na seção 4.4.1), os provedores têm conseguido detectar e bloquear esse tipo de evento em poucas horas, de modo que não tem sido mais viável proceder desta forma. Quanto aos ataques de DDoS, eles apresentam ainda outra dificuldade: utilizam várias máquinas ao mesmo tempo, então seu custo é maior. Dessa forma, o fato de não terem sido detectados ataques dessas categorias no período não é surpreendente.

O número exorbitante de eventos de comunicações advindas de hospedeiros comprometidos (896.008) é preocupante, mas deve-se considerar o sentido dos fluxos analisados por essa assinatura: são fluxos de entrada, e não necessariamente significam que os hospedeiros da rede interna sendo contatados estão comprometidos, uma vez que esses hospedeiros internos possuem IPs reais públicos que podem receber comunicações a qualquer momento, a não ser que estejam bem protegidos por um *firewall*. No entanto, uma análise mais aprofundada sobre essas comunicações é relevante.

Com relação às comunicações com mineradores e com hospedeiros em *blacklist*, ambas tiveram altos números de eventos (13.178 e 206.648, respectivamente), e ambas analisam fluxos de saída, o que implica que os hospedeiros internos estão iniciando a comunicação ativamente. Trata-se de uma situação alarmante, e idealmente os hospedeiros internos que mais realizaram essas comunicações devem ser investigados para determinar o que está causando esses eventos.

Apesar de não terem sido detectadas muitas comunicações com hospedeiros de Comando e Controle (apenas 36), as que foram observadas são um forte indicativo de comprometimento, e devem ser analisadas o mais rápido possível, uma vez que esse tipo de atividade quase sempre implica a ocorrência de exfiltração de dados. O baixo número de eventos detectados provavelmente se deve ao fato de se tratar de um tipo de ocorrência bastante específica, que normalmente segue um ataque bem-sucedido, e portanto tende a ser mais incomum.

As comunicações com *malwares* também são eventos bastante particulares, uma vez que têm como destino um hospedeiro e uma porta específicos, e o próprio *feed* possui poucos endereços listados (sempre são em torno de 90, sendo que os outros *feeds* possuem no mínimo 400 endereços diferentes). Assim, o fato de apenas quatro eventos terem sido detectados é compreensível, mas, por sua natureza altamente perigosa, devem ser investigados com extrema urgência.

5.4.2 Avaliação dos resultados de *Scan* de Hospedeiros

Os 506 eventos de *Scan* detectados envolveram 77 destinos diferentes na rede interna, e 18 origens externas. Dentre essas origens, houve dois endereços de IP que foram responsáveis por 29,03% e 19,56% dos ataques, o que é um forte indicativo de que tais endereços muito provavelmente devem ser bloqueados por regras de *firewall*. Apesar desses dois endereços não apresentarem *hostnames* associados, foi possível identificar alguns endereços externos cujos *hostnames* são subdomínios do *Shodan*, o que implica que correspondem a processos de reconhecimento de alvos.

Além disso, dentre os alvos dos *Scans*, houve um IP em particular que recebeu 75,86% dos ataques no período. Esse endereço, que possui o *hostname* “ip74151.fcfar.unesp.br”, muito provavelmente possui algum ou alguns serviços conhecidamente vulneráveis, e deve ser investigado com urgência. É relevante destacar que esse hospedeiro está no *campus* da UNESP de Araraquara.

Também foi possível observar uma certa constância no momento do dia em que os ataques desse tipo são realizados. Em quatro dos sete dias analisados notou-se um pico de ataques entre as 18h e as 21h. Como os ataques de *Scan* normalmente antecedem outros tipos de ataques mais elaborados, é relevante que um

administrador saiba que essas atividades muito provavelmente ocorrerão nesses horários.

5.4.3 Avaliação dos resultados de *Bruteforce*

Os 8.132 ataques observados no período tiveram 509 origens e 835 destinos internos diferentes. Dessas origens, uma informação chamou a atenção: 31,82% dos ataques vieram do AS da Universidade Federal da Bahia, e 25,38% deles vieram do AS da Associação Rede Nacional de Ensino e Pesquisa. Esse é um forte indicativo de que há um ou mais hospedeiros comprometidos nesse ASs, o que é uma informação interessante considerando que se trata de instituições educacionais.

A porta de destino mais atacada foi a porta 22, recebendo 45,31% dos ataques, seguida da porta 8080, recebendo 12,67% dos ataques, da porta 443, com 8,87% dos ataques, e da porta 80, com 7,4%. Nota-se, então, uma tendência já conhecida dos eventos de *Bruteforce* de se concentrarem nos serviços de SSH e em aplicações *web*, que costumam apresentar mais problemas de credenciais fracas. Outra porta que curiosamente também se destacou foi a 8291, recebendo 6,15% dos ataques, que se trata de uma porta utilizada pela aplicação *Winbox*, serviço do *MikroTik* que possui uma vulnerabilidade conhecida que foi divulgada em 2018 (MIKROTIK, 2018), e até hoje é bastante explorada. É relevante ressaltar que, apesar de existirem vários serviços que usam processos de autenticação, para fins de análise deste cenário em particular foram listados anteriormente apenas os protocolos que mais receberam ataques.

Dentre as origens dos ataques, dois endereços se destacaram por serem responsáveis por 34,98% e 27,87% dos eventos, e muito provavelmente devem ser bloqueados por regras de *firewall*. O IP com maior taxa de ataques de fato pertence ao AS da Universidade Federal da Bahia, e o outro possui o *hostname* “fwpalmas.ifto.edu.br”, que corresponde a um subdomínio do Instituto Federal do Tocantins, que está no AS da Associação Rede Nacional de Ensino e Pesquisa.

Com relação aos alvos, também há dois que se destacaram: o que recebeu 60,59% dos ataques possui o *hostname* “224e.fct.unesp.br”, está localizado no *campus* da UNESP de Presidente Prudente, e recebeu ataques apenas na porta 22; e o

que recebeu 16,93% dos ataques possui o *hostname* “ibb99-131.ibb.unesp.br”, está localizado no *campus* de Botucatu, e recebeu ataques apenas na porta 8080. Claramente deve ser feita uma investigação sobre esses dois hospedeiros mais visados, para que se determine o motivo dessas ocorrências.

Ao contrário dos ataques de *Scan*, que ocorrem em horários específicos do dia, foi possível observar que os de *Bruteforce* ficam constantes ao longo do dia, sendo detectados cerca de 100 ataques a cada 3 horas.

5.4.4 Avaliação das comunicações advindas de hospedeiros comprometidos

As 896.008 comunicações detectadas vieram de 905 endereços externos diferentes, e foram direcionadas para 126.812 endereços internos distintos. A organização responsável por 72,33% dessas comunicações é uma fornecedora de serviço de VPS, o que é compreensível.

Não houve endereços internos que foram alvo de uma porcentagem maior das comunicações: todos os 10 endereços mais contatados receberam cerca de 10% das comunicações. O endereço mais contatado recebeu 11,25% dos fluxos, e corresponde a uma máquina na rede do *campus* de São José do Rio Preto, com o *hostname* “swdzb1.gerencia.ibilce.unesp.br”.

Como essa assinatura detecta comunicações de entrada na rede, ela é bastante relevante para apontar hospedeiros externos que devem ser bloqueados por regras de *firewall*, uma vez que as comunicações identificadas foram iniciadas por eles. Assim, o endereço comprometido de origem que mais efetuou comunicações, sendo responsável por 21,57% delas, deve muito provavelmente ser bloqueado. Infelizmente esse hospedeiro não possui um *hostname* associado, mas é possível saber que ele está no AS do fornecedor de serviço de VPS mencionado anteriormente.

Também foi possível observar que, assim como os ataques de *Bruteforce*, essas comunicações se mantiveram constantes ao longo dos dias analisados, havendo cerca de 7.000 comunicações por hora.

5.4.5 Avaliação das comunicações com mineradores

As 13.178 comunicações com mineradores identificadas no período foram executadas por 126 hospedeiros internos diferentes, para 71 endereços externos. A maior parte dos hospedeiros de destino está em ASs de provedores de serviços de VPS, mas o curioso é que 59,6% desses destinos estão hospedados na Alemanha.

Com relação aos hospedeiros de origem, houve um que efetuou 45,26% das comunicações, o que é um forte indicativo de que está comprometido. Esse hospedeiro possui o *hostname* "usr-186-217-37-133.v4.eduroam.fclar.unesp.br" e está localizado no *campus* de Araraquara. Todas as comunicações realizadas por ele tiveram como destino um endereço específico, que não possui *hostname* associado, mas obviamente também foi o IP de destino que mais recebeu comunicações no período (41,66%).

As comunicações com mineradores também possuem a característica de permanecerem constantes ao longo do dia, sendo detectados cerca de 50 eventos por hora.

5.4.6 Avaliação das comunicações com hospedeiros de Comando e Controle

As 36 comunicações detectadas foram efetuadas por seis hospedeiros da rede interna, para 25 endereços externos diferentes. Destes seis endereços, um foi responsável por efetuar 52,78% das comunicações, o que é um firme indício de que este hospedeiro está comprometido e muito provavelmente está exfiltrando informações. Como ele não possui um *hostname* associado, não é possível determinar o *campus* onde ele está localizado.

Dos endereços externos contatados, um deles, com o *hostname* "server.artisandice.com" associado, se destacou por ter recebido a maior parte das comunicações no período (28,57%).

5.4.7 Avaliação das comunicações com hospedeiros em uma *blacklist*

Os 206.648 eventos de comunicações com hospedeiros em uma *blacklist* identificados foram executados por 10.944 endereços internos, para 3.190 hospedeiros externos diferentes, sendo que as 10 organizações que mais hospedam esses endereços externos são de fornecedores de VPS.

Assim como foi observado com as comunicações advindas de hospedeiros comprometidos, não houve endereços internos que foram responsáveis por executar uma porcentagem maior das comunicações: os 10 endereços que mais se comunicaram realizaram cerca de 10% delas. Dos endereços de destino, o que mais foi contatado recebeu 18,75% dessas comunicações, e possui o *hostname* "recyber.net".

Os eventos de comunicações com hospedeiros em uma *blacklist* também ocorreram de forma contínua ao longo do dia, com cerca de 1.000 eventos por hora.

5.4.8 Avaliação das comunicações com *Malwares*

As quatro comunicações com hospedeiros envolvidos em atividades de *malware* foram feitas por dois endereços internos, com destino a dois alvos externos distintos. Dos endereços internos, um efetuou 75% das comunicações, que corresponde a três dos quatro eventos detectados, e foi possível identificar seu *hostname* como "host105023.fcav.unesp.br", o que significa que uma máquina muito provavelmente infectada está no *campus* de Jaboticabal. O hospedeiro externo que foi contatado por essa máquina, e portanto recebeu a maior parte das comunicações maliciosas (75%), não possui *hostname* associado. O hospedeiro interno que realizou apenas uma comunicação possui o *hostname* associado "wfu199207.fcav.unesp.br", e também está no *campus* de Jaboticabal.

O nome do *malware* presente em três dessas comunicações é desconhecido, pois consta no *feed* como "unknown", mas a outra comunicação detectada registrou o nome do *malware* como "downloader", que se refere a uma família de *malwares* que são capazes de baixar outros programas maliciosos na vítima.

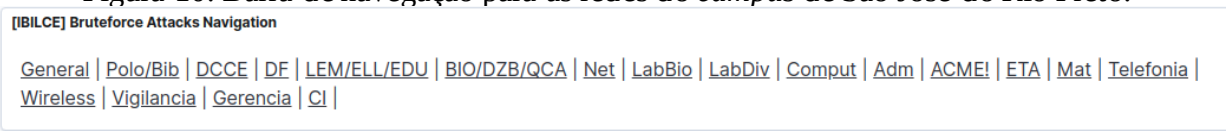
5.5 Resultados para o *campus* de São José do Rio Preto

Para o *campus* de São José do Rio Preto, foi disponibilizada uma lista dos endereços das redes atribuídas a ele, bem como o nome associado a cada uma dessas redes. Isso permitiu criar um conjunto de nove *dashboards* específicos para esse *campus*, como foi mencionado na seção 4.7.

O *dashboard* de *Overview* apresenta informações gerais de tráfego de rede. No período de sete dias analisado, foi possível identificar que, diariamente, o *campus* de São José do Rio Preto possui cerca de 7 mil endereços internos ativos, conversando com cerca de 80 mil hospedeiros externos diferentes. Além disso, são trocados cerca de 3 milhões de fluxos, totalizando aproximadamente 16 GB de tráfego. Participam das comunicações diárias mais de 190 países, cerca de 8.500 cidades diferentes, e mais de 6 mil ASs, sendo que o país que, sozinho, hospeda em torno de 50% dos IPs externos que participam dessas comunicações é os Estados Unidos.

Nos oito *dashboards* de eventos de segurança, além de ser possível observar os eventos detectados apenas no *campus* de São José do Rio Preto de forma geral, é possível também enxergar os eventos específicos de cada uma das 18 redes da instituição. Isso foi possibilitado por meio de *links* pré-programados nos *dashboards*, que dão acesso rápido a filtragens pelas redes específicas na barra de busca. Na Figura 10 é apresentada uma das barras de navegação criadas nos *dashboards* para permitir essa filtragem por redes.

Figura 10: Barra de navegação para as redes do *campus* de São José do Rio Preto.



Fonte: elaborada pela própria autora.

Nessa barra de navegação, o *link* “General” faz uma filtragem por todas as redes do *campus*, mostrando então os eventos totais que ocorreram na instituição. Os outros *links* apontam para filtros das 18 redes do *campus*, permitindo visualizar os eventos para cada uma delas. Dessa forma, caso o administrador visualize um evento de segurança observando o “General”, basta que acesse sucessivamente os *links* para

que seja possível descobrir em qual das redes do instituto o evento foi detectado, facilitando assim sua mitigação.

Isso demonstra que, caso sejam fornecidas as informações das redes específicas dos outros *campi* do Sistema Autônomo, é possível disponibilizar *dashboards* particulares para cada instituição, permitindo que os respectivos administradores de rede vejam informações apenas do *campus* onde atuam, e tratem dos eventos de segurança detectados com maior celeridade. Esse controle de acesso é facilmente implementado pelo *X-Pack* (ELASTIC, 2018), extensão da *Elastic Stack* que se tornou *Open Source* desde 2018, e que permite limitar o acesso de usuários da pilha a conjuntos de *dashboards* específicos.

Na Tabela 5 são mostrados os números de eventos de segurança de cada tipo detectados nas redes do *campus* de São José do Rio Preto, bem como a proporção à qual esses números correspondem com relação ao total de eventos detectados no Sistema Autônomo, nos sete dias analisados.

Tabela 5: Eventos detectados no *campus* de São José do Rio Preto em sete dias.

Assinatura	**Número de eventos**	**Proporção do total**
Scan de Hospedeiro	21	4,15%
Bruteforce	276	3,39%
DoS/DDoS	0	-
Comunicação advinda de hospedeiros comprometidos	61.713	6,89%
Comunicação com mineradores	128	0,97%
Comunicação com hospedeiros de Comando e Controle	0	0%
Comunicação com hospedeiros em *blacklist*	15.448	7,48%
Comunicação com *malwares*	0	0%

Fonte: elaborada pela própria autora.

Nota-se que o *campus* de São José do Rio Preto é responsável por proporções bem baixas do total de eventos de segurança detectados no Sistema Autônomo da

UNESP, sendo todas elas abaixo de 8%. Esse cálculo só foi possível devido às informações fornecidas sobre essa instituição, e, então, conhecendo os endereços das redes dos outros *campi* da Universidade, seria possível também determinar quais instituições possuem maiores incidências de tipos específicos de eventos.

Dos 61.713 eventos de comunicações advindas de hospedeiros comprometidos, a maior parte deles, 12.022 (19,48%), ocorreu nas redes *Wireless* do instituto, que é uma informação previsível, visto que são nessas redes em que se conectam dispositivos móveis, bastante visados neste tipo de evento. Não foi possível detectar algum endereço interno que tenha recebido uma proporção maior das comunicações (todos receberam cerca de 10% delas), mas foi possível determinar um endereço externo responsável por realizar 23,71% delas, que é um indicativo de que esse hospedeiro deve ser bloqueado por regras de *firewall*.

Dos 15.448 eventos de comunicações com endereços em uma *blacklist*, 7.599 (49,19%) deles partiram de hospedeiros na rede da Gerência. Nessa rede, também não foi possível identificar um endereço que efetuou mais comunicações, ficando todos responsáveis por cerca de 10% delas. Quanto aos destinos, o endereço mais contatado recebeu 19,33% das comunicações, mas infelizmente não possui um *hostname* associado.

Com relação aos ataques de *Scan*, não houve uma rede que se destacou por ter recebido um volume maior de ataques; os eventos ficaram bem distribuídos entre as redes do instituto. Os ataques de *Bruteforce*, por outro lado, concentraram-se na rede da Gerência, que recebeu 92,02% deles. Apesar de não ter havido um hospedeiro interno dessa rede que recebeu uma porcentagem superior de ataques, foi possível notar que eles se concentraram nas portas 8728 (51,18%) e 8291 (47,24%), que são portas utilizadas pelo serviço *Winbox*, que é uma aplicação conhecidamente vulnerável do *MikroTik*, como já foi mencionado na seção 5.4.3. Este é um alerta de que se deve procurar por dispositivos *MikroTik* vulneráveis nessa rede, pois essa pode ser a causa desse número considerável de eventos. Além disso, pôde-se perceber que há dois hospedeiros externos que foram responsáveis por realizar 33,99% e 29,25% desses ataques, respectivamente. Um deles possui o *hostname* “fixed-189-203-137-137.totalplay.net” associado.

Os eventos de comunicações com mineradores se concentraram igualmente em duas redes do instituto: a dos Laboratórios de Biologia e a dos Laboratórios de

Computação, ambas tendo executado 46,88% das comunicações. Na rede dos Laboratórios de Biologia, houve apenas um hospedeiro que executou todas as comunicações, e portanto deve se tratar de uma máquina comprometida. Seu *hostname* associado é "biti05.labbio.ibilce.unesp.br", e ela se comunicou com mais frequência com dois endereços externos (35% e 33,33% das comunicações foram destinadas a eles), que possuem os *hostnames* "static.40.80.12.49.clients.your-server.de" e "static.38.80.12.49.clients.your-server.de".

Na rede dos Laboratórios de Computação, também houve apenas um hospedeiro que foi responsável por todas as comunicações para endereços de mineradores, com o *hostname* "lim04.comput.ibilce.unesp.br". Este, no entanto, contatou um destino com mais frequência: um endereço com o *hostname* "xmrpool.eu" associado, que recebeu 71,19% das comunicações.

5.6 Avaliação das execuções das assinaturas de eventos

Outro resultado relevante diz respeito ao tempo de execução das assinaturas. Como elas são executadas paralelamente a cada 5 minutos, idealmente sua execução não deve ultrapassar esse período de tempo para evitar a inserção de eventos repetidos no *Elasticsearch*, uma vez que haveria choque no *timestamp* dos eventos detectados.

Então, foi feita uma listagem do tempo necessário para cada execução de cada uma das oito assinaturas utilizadas ao longo dos sete dias analisados, e calculou-se a média desses tempos de execução. Os tempos médios de execução em segundos por tipo de assinatura são apresentados na Tabela 6.

Tabela 6: Tempo médio de execução das assinaturas, ao longo de sete dias.

Assinatura	**Tempo médio de execução (s)**
Scan de Hospedeiro	7,34
Bruteforce	14,67
DoS/DDoS	10,07

Comunicação advinda de hospedeiros comprometidos	92,19
Comunicação com mineradores	13,45
Comunicação com hospedeiros de Comando e Controle	9,18
Comunicação com hospedeiros em *blacklist*	55,43
Comunicação com *malwares*	154,28

Fonte: elaborada pela própria autora.

É possível notar que as assinaturas de *Scan* de Hospedeiros, *Bruteforce*, DoS/DDoS, e de comunicações com mineradores e com hospedeiros de Comando e Controle possuem tempos médios de execução próximos de 10 segundos, o que faz sentido considerando que apresentaram números de eventos no máximo na casa das dezenas de milhares. Deve-se considerar que a etapa de enriquecimento presente em todas as assinaturas faz consultas no *Elasticsearch* e consultas DNS reversas para obter as informações, e que um número menor de resultados da etapa de detecção da assinatura impacta diretamente na redução do tempo de execução da etapa de enriquecimento, pois haverá menos endereços para serem consultados.

Assim, é esperado que as assinaturas de *Scan* e de comunicação com hospedeiros de Comando e Controle apresentem tempos médios de execução menores, uma vez que detectaram apenas 506 e 36 eventos, respectivamente, e que as assinaturas de comunicações advindas de hospedeiros comprometidos e de comunicações com hospedeiros em *blacklist* possuam tempos médios de execução maiores, considerando que detectaram 896.008 e 206.648 eventos, respectivamente.

A princípio, o fato da assinatura de comunicações com *malwares* ter apresentado o maior tempo médio de execução (154,28 segundos), apesar de ter detectado apenas quatro eventos no período, pode parecer incoerente. No entanto, ao analisar seu funcionamento, descrito pelo algoritmo 8 apresentado na seção 4.4.2, é possível notar que essa assinatura é uma das únicas que faz consultas ao *Elasticsearch* usando como filtros tanto endereços IP quanto portas de destino específicas. Apesar das assinaturas de detecção de comunicações advindas de hospedeiros comprometidos (algoritmo 4) e de detecção de comunicações com

hospedeiros de Comando e Controle (algoritmo 6) também utilizarem a porta de destino (ou origem) como parte de seus filtros, analisando os arquivos de *feed* utilizados por elas pode-se perceber que em nenhuma das linhas há especificação da porta de destino, apesar dessa possibilidade existir, e por isso ter sido considerada nas assinaturas. Em todas as versões dos arquivos de *feed* empregados por essas duas assinaturas até o momento, nunca ocorreu de alguma das portas de destino (ou origem) serem diferentes do valor "any", que significa que a porta não é especificada, e portanto as consultas consideram apenas os endereços IP como filtros.

Por outro lado, para o *feed* utilizado pela assinatura de comunicações com *malwares*, a porta de destino sempre é especificada em todas as linhas do *feed*, e esse filtro é então sempre considerado em todas as consultas. Isso faz com que a consulta ao *Elasticsearch* seja mais específica, e, consequentemente, faz com que ela consuma mais tempo para ser executada.

Mesmo nesse caso da assinatura de detecção de comunicações com *malwares*, o tempo médio de execução foi de 2 minutos e 34 segundos, bem abaixo dos 5 minutos que haviam sido considerados como limite superior de execução para evitar a inserção de documentos duplicados.

5.7 Considerações sobre os resultados obtidos

A análise dos resultados obtidos deixou clara a relevância do modelo proposto. Por meio dele, foi possível entender o comportamento do tráfego da rede do Sistema Autônomo, e entender também os perfis dos eventos de segurança que ocorreram nessa rede. As visualizações construídas e sua organização em *dashboards* permitiram chegar a algumas conclusões que muito provavelmente não seriam possíveis sem esses recursos, como endereços externos que devem ser bloqueados por regras de *firewall*, endereços internos que parecem estar comprometidos e devem ser investigados, e, em alguns casos, em qual instituição esse hospedeiro está.

Além disso, o estudo das redes específicas do *campus* de São José do Rio Preto demonstrou o potencial de granularidade que o modelo pode proporcionar. Por meio de uma simples filtragem pelas redes do instituto, foi possível identificar as redes que são mais atacadas, potenciais motivos pelos quais esses ataques podem

estar ocorrendo, e também hospedeiros de redes específicas que estão asseguradamente comprometidos. Todos esses fatores auxiliam de forma valiosa na mitigação de eventos de segurança ocorrendo na rede de uma instituição desse porte.

CAPÍTULO 6 - Conclusões

6.1 Considerações Iniciais

Este capítulo tem como objetivo apresentar as conclusões do trabalho realizado. Na seção 6.2, são apresentadas as conclusões gerais; na seção 6.3, são expostas as dificuldades que foram encontradas durante a elaboração desta pesquisa; por fim, na seção 6.4, propõe-se trabalhos futuros que podem ser desenvolvidos.

6.2 Conclusões gerais

O aumento no número de ataques em redes de grande porte evidencia a importância de se conseguir obter informações sobre essas redes e entendê-las para que seja possível defender esses ambientes adequadamente. Modelos de Consciência Situacional em Cibersegurança têm sido sugeridos com este propósito, mas o alto custo computacional resultante do uso de vários tipos de entradas de dados, principalmente no contexto de *Big Data*, dificulta seu uso em ambientes reais.

Neste trabalho, propôs-se um modelo de Consciência Situacional em Cibersegurança para Sistemas Autônomos que utiliza apenas fluxos de rede *Netflow* V9 como entrada. O sistema foi capaz de processar o volume de fluxos exportado pelo roteador de borda do Sistema Autônomo da UNESP, e de fornecer uma visão holística do tráfego recebido e dos eventos de segurança ocorrendo na organização, permitindo detectar esses eventos com um atraso mínimo, informando o momento

em que aconteceram e detalhes dos hospedeiros afetados, e auxiliando, assim, um administrador de rede a tomar as medidas necessárias para mitigar esses eventos com celeridade. Além disso, o fato de serem empregados apenas fluxos de rede para esse propósito respeita a privacidade dos usuários, e contribui para a redução do custo de funcionamento do modelo.

Os resultados demonstraram que os objetivos deste trabalho foram alcançados, pois o sistema proposto foi capaz de coletar fluxos, detectar os eventos de segurança, enriquecê-los e visualizá-los de forma que foi possível compreender o estado da rede do Sistema Autônomo e das potenciais ameaças em andamento nesse ambiente, correspondendo aos dois primeiros estágios de *Cyber Situational Awareness*. Esse processo foi efetuado por meio de um modelo de consultas flexível, que permite a detecção de ataques e o cruzamento das informações presentes nos fluxos com *feeds* de *Threat Intelligence*, e também por meio de visualizações que possibilitam analisar os dados enriquecidos e tirar conclusões significativas a partir deles. A versatilidade da *Elastic Stack* foi a base que propiciou o desenvolvimento de uma arquitetura ao mesmo tempo simples e escalável para o modelo proposto.

Os resultados obtidos especificamente para o *campus* de São José do Rio Preto demonstraram o potencial de granularidade que o modelo apresenta. É possível, ao mesmo tempo, visualizar informações de tráfego e de eventos de segurança para o AS inteiro, e aumentar a especificidade das visualizações até que se consiga analisar um evento ocorrendo em uma rede de uma instituição em particular.

Dessa forma, nota-se que os resultados obtidos pela abordagem proposta se mostraram de grande valia, uma vez que trabalhos semelhantes, como Jirsik e Celeda (2018, 2020), Yang et al. (2019) e *ElastiFlow* (2020), apesar de usarem apenas fluxos de rede como entrada, não proporcionam o mesmo nível de abstração do conjunto de redes analisadas que é apresentado pelo modelo sugerido. Além disso, os trabalhos correlatos não possibilitam uma flexibilidade para a detecção de novos tipos de eventos de segurança, o que o modelo apresentado viabiliza por meio da lógica empregada nas assinaturas.

Assim, considerando-se que a aplicabilidade dos aspectos do sistema proposto em um contexto real foi totalmente demonstrada, uma vez que os resultados apresentados se deram a partir de fluxos exportados durante uma semana de funcionamento normal da rede do Sistema Autônomo da UNESP, pode-se afirmar

que o trabalho apresenta uma contribuição relevante para a comunidade de pesquisa em cibersegurança.

6.3 Dificuldades encontradas

A principal dificuldade encontrada durante o desenvolvimento deste trabalho foi o tratamento do atraso que o roteador de borda apresenta para a exportação dos fluxos. Esse atraso impactou diretamente na lógica de funcionamento das assinaturas, uma vez que estas devem fazer suas consultas nos fluxos indexados dos penúltimos 5 minutos. Isso ocasiona um atraso de 10 minutos entre o momento da ocorrência do evento e sua detecção pela assinatura correspondente, como foi explicado na seção 4.4. Deve-se observar, no entanto, que essa detecção tardia é suplantada caso não haja atraso na exportação dos fluxos, uma vez que seria possível, nesse caso, consultar os últimos 5 minutos para realizar a detecção dos eventos.

Outra dificuldade diz respeito à falta de informações sobre os endereços IP das outras 23 instituições de ensino da Universidade, que impediram a obtenção de resultados mais específicos sobre vários eventos de segurança detectados. O fornecimento dos endereços das redes do *campus* de São José do Rio Preto permitiu que resultados bastante elucidativos sobre os eventos ocorrendo nessa instituição fossem relatados. Infelizmente, a falta de informações semelhantes sobre os outros *campi* impossibilitou que observações análogas fossem feitas sobre os eventos nas redes dessas unidades, além de limitar conclusões de interesse sobre o próprio Sistema Autônomo, como quais são as instituições que recebem maiores volumes de tráfego diariamente, e quais *campi* apresentam a maior incidência de eventos de segurança de cada tipo.

6.4 Trabalhos futuros

Para os trabalhos futuros, propõe-se criar novas assinaturas para detectar outros tipos de eventos de rede, como a presença de protocolos menos comuns que

podem indicar atividades de interesse e potencialmente maliciosas, como o IRC[31] e o ICMP.

A adição de novos *feeds* de *Cyber Threat Intelligence* nas assinaturas de detecção de eventos de segurança é também uma sugestão que permitiria expandir ainda mais os tipos de eventos identificados pelo modelo. Outra possibilidade seria fazer a análise de relatórios de campanhas de ameaças, divulgados por grupos e empresas de segurança, para criar novas assinaturas. Para isso, seria necessário estruturar um novo módulo que seja capaz de parsear esses relatórios e extrair informações de interesse, como os endereços de IP e portas envolvidos no evento, para então criar as novas assinaturas que serão incluídas no sistema.

Sugere-se também aperfeiçoar as assinaturas para otimizar seu tempo de execução. Duas maneiras de se fazer esse ajuste seriam identificando trechos dos códigos que podem ser paralelizados, e empregando *caches* locais para reduzir o número de consultas DNS repetidas realizadas na etapa de enriquecimento dos eventos detectados.

Por fim, propõe-se integrar o modelo proposto com outro trabalho que foi desenvolvido em paralelo com este no laboratório ACME!, que utiliza redes neurais para prever a ocorrência de alguns dos eventos de segurança mencionados na seção 4.4 (ataques de *Scan* de Hospedeiros, *Bruteforce*, comunicações advindas de hospedeiros comprometidos e comunicações com mineradores) nas próximas 24 horas. Uma integração com esse trabalho permitiria criar novos *dashboards* contendo informações sobre os eventos de segurança previstos, auxiliando ainda mais na sua mitigação, uma vez que o administrador de rede poderia criar medidas de proteção levando em consideração essas previsões. Essa integração possibilitaria que o modelo de Consciência Situacional em Cibersegurança sugerido executasse os três estágios de CSA apresentados na seção 2.4, uma vez que, atualmente, ele engloba apenas os dois primeiros.

[31] *Internet Relay Chat* é um protocolo de comunicação que era bastante utilizado por aplicações de bate-papo em tempo real.

REFERÊNCIAS

ABU, M. S.; SELAMAT, S. R.; ARIFFIN, A.; YUSOF, R. **Cyber threat intelligence – Issue and challenges.** Indonesian Journal of Electrical Engineering and Computer Science, [s. l.], v. 10, n. 1, p. 371–379, 2018.

ABUBAKAR, Y.; ADEYI, T. S.; AUTA, I. G. **Performance evaluation of NoSQL systems using YCSB in a resource austere environment.** International Journal of Applied Information Systems (IJAIS), [s.l.], p. 23–27, 2014.

ABUSEIPDB. **AbuseIPDB**, 2021. BLACKLIST Endpoint**.** Disponível em: <https://api.abuseipdb.com/api/v2/blacklist>. Acesso em: 17 ago. 2021.

ADENUSI, D.; ALESE, B. K.; KUBOYE, B. M.; THOMPSON, A. F. B. **Development of cyber situation awareness model.** International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), [s. l.], p. 1–11, 2015.

AFEK, Y.; BREMLER-BARR, A.; LANDAU FEIBISH, S. **Automated signature extraction for high volume attacks.** Proceedings of the ninth ACM/IEEE symposium on Architectures for networking and communications systems, [s. l.], p. 147-156, 2013.

BASS, T. **Intrusion Detection Systems & Multisensor Data Fusion: Creating Cyberspace Situational Awareness.** Communications of the ACM, [s. l.], v. 43, n. 4, p. 99–105, 2000.

BOUKHTOUTA, A.; MOKHOV, S. A.; LAKHDARI, N. E.; DEBBABI, M.; PAQUET, J. **Network malware classification comparison using DPI and flow packet headers.** Journal of Computer Virology and Hacking Techniques, [s. l.], v. 12, n. 2, p. 69-100, 2016.

BRASIL. Lei nº 12.965, de 23 de abril de 2014. **Estabelece princípios, garantias, direitos e deveres para o uso da Internet no Brasil**, Brasília, DF, abr. 2014.

CANONICAL LTD. **Ubuntu**, 2021. Ubuntu 18.04.5 LTS (Bionic Beaver). Disponível em: <https://releases.ubuntu.com/18.04.5/>. Acesso em: 17 ago. 2021.

CERT.BR. **CERT.br**, 2020. Estatísticas dos Incidentes Reportados ao CERT.br. Disponível em: <https://www.cert.br/stats/incidentes/>. Acesso em: 15 ago. 2021.

CHAUDHARI, S.; MAURYA, V. K.; SINGH, V.; TOMAR, S. S.; RAJAN, A.; RAWAT, A. **Real Time Logs and Traffic Monitoring, Analysis and Visualization setup for IT Security Enhancement.** Analysis and Visualization Setup for IT Security Enhancement, [s. l.], p. 1–11, 2020.

CINQUE, M.; DELLA CORTE, R.; PECCHIA, A. **Contextual filtering and prioritization of computer application logs for security situational awareness.** Future Generation Computer Systems, [s. l.], v. 111, p. 668–680, 2020.

CISCO. **Cisco**, 2011. NetFlow Version 9 Flow-Record Format. Disponível em: <https://www.cisco.com/en/US/technologies/tk648/tk362/technologies_white_paper 09186a00800a3db9.html>. Acesso em: 22 ago. 2021.

CISCO. **Cisco**, 2012. Introduction to Cisco IOS NetFlow - A Technical Overview. Disponível em: <https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html>. Acesso em: 22 ago. 2021.

CISCO. **Cisco**, 2020. Cisco Annual Internet Report (2018–2023). Disponível em: <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>. Acesso em: 15 ago. 2021.

CLAISE, B.; SADASIVAN, G.; VALLURI, V.; DJERNAES, M. **IETF**, 2004. RFC 3954: Cisco systems NetFlow services export version 9. Disponível em: <http://www. ietf. org/rfc/rfc3954.txt>. Acesso em: 09 dez. 2019.

CORRÊA, J. L. **Um modelo de detecção de eventos em redes baseado no rastreamento de fluxos.** [s. l.], 2009.

DE MELO E SILVA, A.; GONDIM, J. J. C.; DE OLIVEIRA ALBUQUERQUE, R.; VILLALBA, L. J. G. **A methodology to evaluate standards and platforms within cyber threat intelligence.** Future Internet, [s. l.], v. 12, n. 6, p. 1–23, 2020.

ELASTIC. **Elastic**, 2018. We opened X-Pack. Disponível em: <https://www.elastic.co/pt/what-is/open-x-pack>. Acesso em: 20 ago. 2021.

ELASTIC. **Elastic**, 2021. Dashboard. Disponível em: <https://www.elastic.co/guide/en/kibana/current/dashboard.html>. Acesso em: 18 ago. 2021.

ELASTIC. **Elastic**, 2021. Your window into the Elastic Stack. Disponível em: <https://www.elastic.co/kibana>. Acesso em: 16 ago. 2021.

ELASTIC. **Elastic**, 2021. Kibana Query Language. Disponível em: <https://www.elastic.co/guide/en/kibana/current/kuery-query.html>. Acesso em: 18 ago. 2021.

ELASTIC. **Elastic**, 2021. Lightweight shipper for logs. Disponível em: <https://www.elastic.co/pt/beats/filebeat>. Acesso em: 16 ago. 2021.

ELASTIC. **Elastic**, 2021. NefFlow module. Disponível em: <https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-module-netflow.html#filebeat-module-netflow>. Acesso em: 16 ago. 2021.

ELASTIC. **Elastic**, 2021. Open Source Search & Analytics. Disponível em: <https://www.elastic.co/pt/>. Acesso em: 15 ago. 2021.

ELASTICSEARCH. **Elasticsearch**, 2021. Python Elasticsearch Client. Disponível em: <https://elasticsearch-py.readthedocs.io/en/master/>. Acesso em: 16 ago. 2021.

ELASTICSEARCH DSL. **Elasticsearch DSL**, 2021. Elasticsearch DSL high-level library. Disponível em: <https://elasticsearch-dsl.readthedocs.io/en/latest/#>. Acesso em: 16 ago. 2021.

ELASTIFLOW. **ElastiFlow**, 2020. Network Performance and Security Analytics for Open Data Platforms. Disponível em: <https://www.elastiflow.com/>. Acesso em: 16 ago. 2021.

ELSEVIER. **ScienceDirect,** 2021. Search for peer-reviewed journal articles and book chapters (including open access content). Disponível em: <https://www.sciencedirect.com/>. Acesso em: 22 ago. 2021.

ENDSLEY, M. R. **Design and evaluation for situation awareness enhancement.** Proceedings of the Human Factors Society annual meeting, [s. l.], v. 32, n. 2, p. 97-101, 1988.

FRANKE, U.; BRYNIELSSON, J. **Cyber situational awareness - A systematic review of the literature.** Computers and Security, [s. l.], v. 46, p. 18–31, 2014.

FRIEDMAN, J; BOUCHARD, M. **Definitive Guide to Cyber Threat Intelligence:** using knowledge about adversaries to win the war against targeted attacks. Annapolis: Cyberedge Group, LLC, 2015.

GOLLMAN, D. **Computer Security.** 3. ed. West Sussex: John Wiley & Sons Ltd, 2011.

GOOGLE. **Google Acadêmico**, 2021. Google Acadêmico. Disponível em: <https://scholar.google.com.br/>. Acesso em: 22 ago. 2021.

HAN, W.; TIAN, Z.; HUANG, Z.; ZHONG, L.; JIA, Y. **System architecture and key technologies of network security situation awareness system YHSAS.** Computers, Materials and Continua, [s. l.], v. 59, n. 1, p. 167–180, 2019.

HAUSER, V. **GitHub**, 2021. Hydra. Disponível em: <https://github.com/vanhauser-thc/thc-hydra>. Acesso em: 17 ago. 2021.

HUSÁK, M.; JIRSÍK, T.; YANG, S. J. **SoK: Contemporary Issues and Challenges to Enable Cyber Situational Awareness for Network Security.** Proceedings of the 15th International Conference on Availability, Reliability and Security, [s. l.], p. 1–10, 2020.

IEEE. **IEEE Xplore**, 2021. Advancing Technology for Humanity. Disponível em: <https://ieeexplore.ieee.org/Xplore/home.jsp>. Acesso em: 22 ago. 2021.

INTERNET STORM CENTER. **SANS**, 2021. Threatlist Miner. Disponível em: <https://isc.sans.edu/api/threatlist/miner>. Acesso em: 17 ago. 2021.

JIRSIK, T.; CELEDA, P. **Toward real-time network-wide cyber situational awareness.** IEEE/IFIP Network Operations and Management Symposium: Cognitive Management in a Cyber World, NOMS, [s. l.], p. 1–7, 2018.

JIRSIK, T.; CELEDA, P. **Cyber Situation Awareness via IP Flow Monitoring.** NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, [s. l.], p. 1–6, 2020.

KINABLE, J. **Detection of network scan attacks using flow data.** 9th Twente Student Conference on IT, 23th June, [s. l.], 2008.

KREBS, B. Kaseya Left Customer Portal Vulnerable to 2015 Flaw in its Own Software. **KrebsonSecurity**, 2021. Disponível em: <https://krebsonsecurity.com/2021/07/kaseya-left-customer-portal-vulnerable-to-2015-flaw-in-its-own-software/>. Acesso em: 15 ago. 2021.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a Internet: uma abordagem top-down.** 6. ed. São Paulo: Pearson Education do Brasil, 2013.

LAKKARAJU, K.; YURCIK, W.; LEE, A. J. **NVisionIP: NetFlow visualizations of system state for security situational awareness.** VizSEC/DMSEC ’04: Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security, [s. l.], p. 65–72, 2004.

LEE, C. B.; ROEDEL, C.; SILENOK, E. **Detection and Characterization of Port Scan Attacks.** Department of Computer Science & Engineering, University of California, San Diego, 2003.

LI, B.; SPRINGER, J.; BEBIS, G.; GUNES, M. H. **A survey of network flow applications.** Journal of Network and Computer Applications, [s. l.], v. 36, n. 2, p. 567-581, 2013.

LI, Y.; HUANG, G.; WANG, C.; LI, Y. **Analysis framework of network security situational awareness and comparison of implementation methods.** EURASIP Journal on Wireless Communications and Networking, [s. l.], v. 2019, n. 1, p. 1–32, 2019.

LIU, X.; YU, J.; LV, W.; YU, D.; WANG, Y.; WU, Y. **Network security situation: From awareness to awareness-control.** Journal of Network and Computer Applications, [s. l.], v. 139, n. April, p. 15–30, 2019.

MALSILO. **MalSilo**, 2021. MalSilo IPv4 feed. Disponível em: <https://malsilo.gitlab.io/feeds/dumps/ip_list.txt>. Acesso em: 17 ago. 2021.

MIKROTIK. **MikroTik**, 2018. CVE-2018-14847 Winbox Vulnerability. Disponível em: <https://blog.mikrotik.com/security/winbox-vulnerability.html>. Acesso em: 19 ago. 2021.

MIRKOVIC, J.; REIHER, P. **A taxonomy of DDoS attack and DDoS defense mechanisms.** ACM SIGCOMM Computer Communication Review, [s. l.], v. 34, n. 2, p. 39-53, 2004.

MISP PROJECT. **MISP Project**, 2021. MISP - Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing. Disponível em: <https://www.misp-project.org/feeds/>. Acesso em: 17 ago. 2021.

MITRE. **MITRE ATT&CK**, 2021. Endpoint Denial of Service: Service Exhaustion Flood. Disponível em: <https://attack.mitre.org/techniques/T1499/002/>. Acesso em: 29 set. 2021.

MITRE. **MITRE ATT&CK**, 2021. Forced Authentication. Disponível em: <https://attack.mitre.org/techniques/T1187/>. Acesso em: 29 set. 2021.

MITRE. **MITRE ATT&CK**, 2021. MITRE ATT&CK. Disponível em: <https://attack.mitre.org/>. Acesso em: 29 set. 2021.

MITRE. **MITRE ATT&CK**, 2021. Network Service Scanning. Disponível em: <https://attack.mitre.org/techniques/T1046/>. Acesso em: 29 set. 2021.

NMAP. **Nmap**, 2021. Nmap: the network mapper. Disponível em: <https://nmap.org/>. Acesso em: 17 ago. 2021.

PINKAS, B.; SANDER, T. **Securing passwords against dictionary attacks.** Proceedings of the 9th ACM conference on Computer and communications security, [s. l.], p. 161-170, 2002.

POSTEL, J.; REYNOLDS, J. **IETF**, 1983. RFC 854: Telnet Protocol Specification. Disponível em: <https://tools.ietf.org/html/rfc854>. Acesso em: 16 ago. 2021.

PROOFPOINT INC. **Emerging Threats**, 2021. Emerging Threats Botnet Command and Control drop rules. Disponível em: <https://rules.emergingthreats.net/open/snort-2.9.0/rules/emerging-botcc.rules>. Acesso em: 17 ago. 2021.

PROOFPOINT INC. **Emerging Threats**, 2021. Rules to block known hostile or compromised hosts. Disponível em: <https://rules.emergingthreats.net/open/snort-2.9.0/rules/emerging-compromised.rules>. Acesso em: 17 ago. 2021.

PYTHON SOFTWARE FOUNDATION. **Python,** 2021. Python programming language. Disponível em: <https://www.python.org/>. Acesso em: 17 ago. 2021.

RED HAT. **Ansible**, 2021. Simple, agentless IT automation that anyone can use. Disponível em: <https://www.ansible.com/>. Acesso em: 18 ago. 2021.

SANFILIPPO, S. **Hping**, 2006. Hping - Active Network Security Tool. Disponível em: <http://www.hping.org/>. Acesso em: 17 ago. 2021.

SCHUBA, C. L.; KRSUL, I. V.; KUHN, M. G.; SPAFFORD, E. H.; SUNDARAM, A.; ZAMBONI, D. **Analysis of a denial of service attack on TCP.** Symposium on Security and Privacy, [s. l.], p. 208-223, 1997.

SHODAN. **Shodan**, 2021. Search Engine for the Internet of Everything. Disponível em: <https://www.shodan.io/>. Acesso em: 18 ago. 2021.

SPEROTTO, A.; SCHAFFRATH, G.; SADRE, R.; MORARIU, C.; PRAS, A.; STILLER, B. **An overview of IP flow-based intrusion detection.** IEEE Communications Surveys and Tutorials, [s. l.], v. 12, n. 3, p. 343–356, 2010.

STROM, B. E.; APPLEBAUM, A.; MILLER, D. P.; NICKELS, K. C.; PENNINGTON, A. G.; THOMAS, C. B. **MITRE ATT&CK: Design and Philosophy.** MITRE Product MP, p. 18–944, 2018.

THE APACHE SOFTWARE FOUNDATION. **Apache Kafka**, 2021. A distributed streaming platform. Disponível em: <https://kafka.apache.org/>. Acesso em: 16 ago. 2021.

THE APACHE SOFTWARE FOUNDATION. **Apache Lucene,** 2021. Welcome to Apache Lucene. Disponível em: <https://lucene.apache.org>. Acesso em: 16 ago. 2021.

THE APACHE SOFTWARE FOUNDATION. **Apache Spark**, 2021. Apache Spark™ is a unified analytics engine for large-scale data processing. Disponível em: <https://spark.apache.org/>. Acesso em: 16 ago. 2021.

THE APACHE SOFTWARE FOUNDATION. **Apache Spark**, 2021. GraphX is Apache Spark's API for graphs and graph-parallel computation. Disponível em: <https://spark.apache.org/graphx/>. Acesso em: 16 ago. 2021.

THREATVIEW. **Threatview**, 2021. Threat Intelligence Feeds Generated by Threatview[.]io Proactive c&c Hunter with domain and IP address. Disponível em:<https://threatview.io/Downloads/High-Confidence-CobaltstrikeC2_IP_feed.txt>. Acesso em: 17 ago. 2021.

TUNG, L. Ransomware: Meat firm JBS says it paid out $11m after attack. **ZDNet**, 2021. Disponível em: <https://zdnet.com/article/ransomware-meat-firm-jbs-says-it-paid-out-11m-after-attack/>. Acesso em: 15 ago. 2021.

UMER, M. F.; SHER, M.; BI, Y. **Flow-based intrusion detection: Techniques and challenges.** Computers & Security, [s. l.], v. 70, p. 238-254, 2017.

VAAS, L. Law Firm to the Fortune 500 Breached with Ransomware. **ThreatPost**, 2021. Disponível em: <https://threatpost.com/law-firm-fortune-500-breach-ransomware/167951/>. Acesso em: 15 ago. 2021.

VARGA, M.; WINKELHOLZ, C.; TRÄBER-BURDIN, S. **Cyber Situation Awareness.** Cyber Security Science and Engineering, [s. l.], p. 1–18, 2016.

VIEGAS, E.; SANTIN, A.; BESSANI, A.; NEVES, N. **BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks.** Future Generation Computer Systems, [s. l.], v. 93, p. 473–485, 2019.

YANG, C. T.; JIANG, W. J.; KRISTIANI, E.; CHAN, Y. W.; LIU, J. C. **The Implementation of a Network Log System Using RNN on Cyberattack Detection with Data Visualization.** International Conference on Frontier Computing, [s. l.], v. 551, p. 321–329, 2019.

YONG, Z.; XIAOBIN, T.; HONGSHENG, X. **A Novel Approach to Network Security Situation Awareness Based on Multi-Perspective Analysis.** International Conference on Computational Intelligence and Security (CIS 2007), [s. l.], n. June, p. 262–265, 2007.

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

TERMO DE REPRODUÇÃO XEROGRÁFICA

Autorizo a reprodução xerográfica do presente Trabalho de Conclusão, na íntegra ou em partes, para fins de pesquisa.

São José do Rio Preto, 29/09/2021

_____

Assinatura do autor