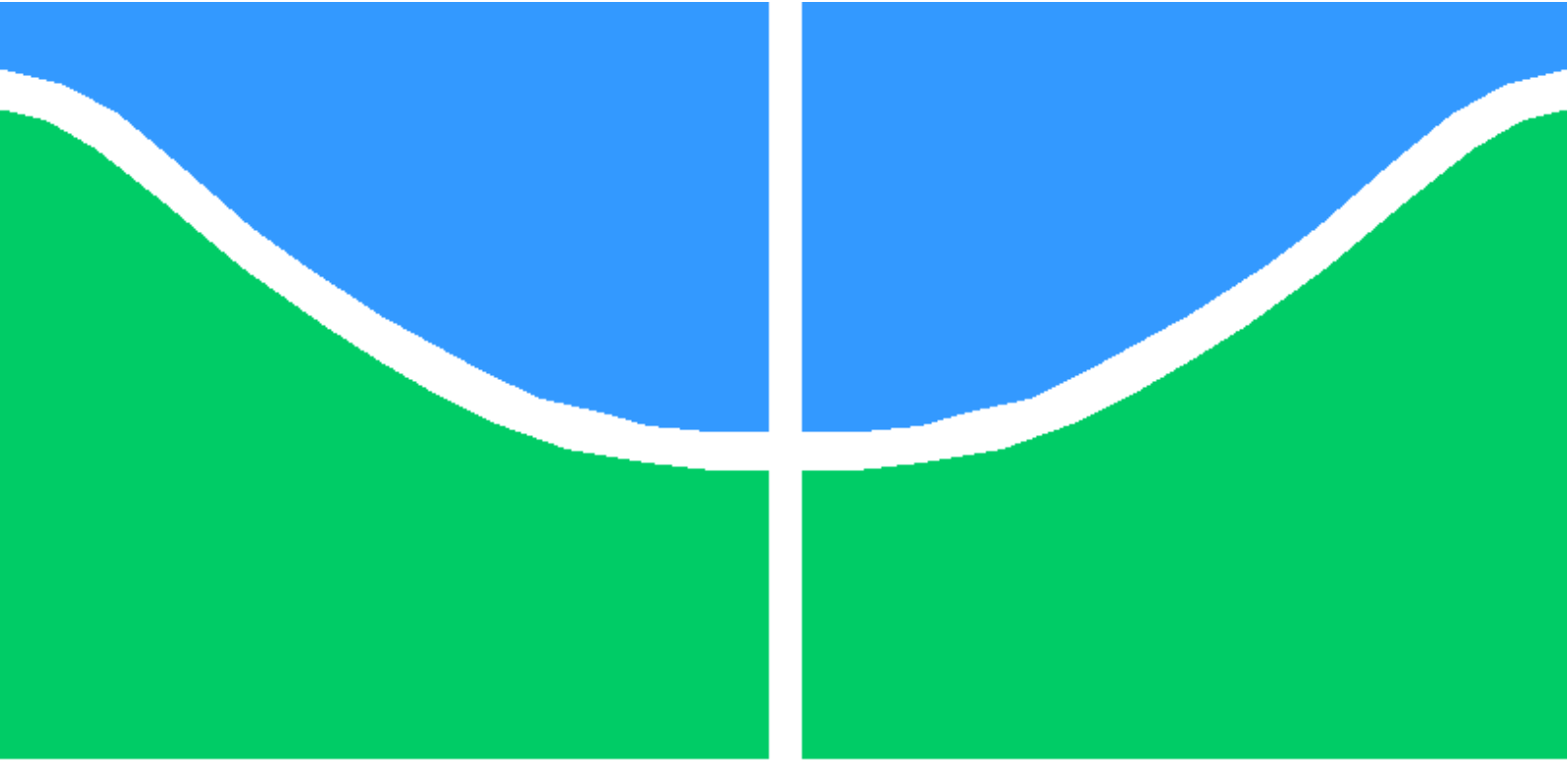


PROJETO FINAL DE GRADUAÇÃO

Proposta prática de laboratório virtual para o gerenciamento de métricas de desempenho, eventos e informações de segurança em ambiente de alta disponibilidade

Larissa Pires de Holanda

Pedro Henrique Nogueira da Silva

Brasília, maio de 2022

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

PROJETO FINAL DE GRADUAÇÃO

Proposta prática de laboratório virtual para o gerenciamento de métricas de desempenho, eventos e informações de segurança em ambiente de alta disponibilidade

Larissa Pires de Holanda

Pedro Henrique Nogueira da Silva

Relatório submetido ao Departamento de Engenharia
Elétrica como requisito parcial para obtenção
do grau de Engenheiro de Redes de Comunicação

Banca Examinadora

Dr. Georges Daniel Amvame Nze, ENE/UnB _____
Orientador

Dr. Fábio Lúcio Lopes de Mendonça, ENE/UnB _____
Examinador interno

Ms. Diego Martins de Oliveira, IFB/Brasília _____
Examinador externo

Agradecimentos

É com muita alegria que agradeço a todos que estiveram do meu lado durante todos esses anos de graduação. Muito grata a Deus em primeiro lugar, a minha família e amigos. Obrigada ao corpo docente da Universidade de Brasília por todos os ensinamentos, em especial ao professor Georges Daniel por toda a orientação nos últimos anos. Agradeço em especial ao meu amigo Pedrinho, não só pela parceria nesse projeto, mas também por toda nossa jornada até aqui e pela amizade para além dos muros da universidade.

Larissa Pires de Holanda

Tomo este lugar para agradecer à todos que, de fato, me apoiaram nessa longa e difícil jornada que é a graduação. Agradeço especialmente à minha família direta, aqueles que comigo moravam e que de mim cuidavam. Agradeço então diretamente à minha mãe Maristela, meu pai Lailson, minha irmã Amanda, minha tia Socorro e minha madrinha Genezia. Estes me proporcionaram as oportunidades e as condições de cursar o Ensino Superior sem quaisquer dificuldades, sendo então um grande prazer poder transmitir este orgulho àqueles que, em mim, mais acreditaram.

Agradeço também à minha esposa Nicole, que foi capaz de me suportar até mesmo nos diversos momentos em que estava à beira de um colapso. Sem a sua ajuda, sem as mudanças que me fez passar e sem as aventuras que experimentamos juntos nos últimos três anos, não teria sido fácil assim. Para complementar, agradeço imensamente também aos meus bichinhos de estimação Lumi e Jennie, que não me deixaram sequer um segundo sozinho em frente ao computador, sempre me importunando. Do jeito bom, claro.

Agradeço também à minha grande amiga e parceira Larissa, que aceitou trabalhar nessa proposta juntos. Apesar de ter sido um ano bem difícil para nós, acredito fielmente que nossos esforços serão recompensados. Agradeço também aos meus demais poucos amigos e todas as risadas que demos juntos sobre as boas e más surpresas da vida. Não preciso mencionar nomes, vocês sabem quem são.

Por fim, agradeço também ao nosso professor e orientador Georges. Ele, além de ótimo professor, é extremamente preocupado com a evolução da educação de seus alunos e, consequentemente, de mim, da Larissa e de nossos colegas. Agradeço por todo o suporte e paciência que nos forneceu nesse percurso final.

Gostaria de deixar claro que escrevo estes agradecimentos extremamente cansado, mas afirmo com convicção que me orgulho do que desenvolvi e de quem me tornei após toda essa jornada. Não me arrependo de nada e prometo me dedicar ainda mais para as próximas etapas que virão!

Pedro Henrique Nogueira da Silva

RESUMO

Diante do progressivo crescimento da utilização e importância da comunicação via Internet, a quantidade de dados transmitidos têm escalado exponencialmente e, entre isso, os requisitos de privacidade e segurança da informação na transferência destes dados se tornaram tópicos frequentes de preocupação, estudo e aprimoramento. Este trabalho consiste na apresentação de uma proposta de laboratório virtual emulado que implementa, de forma prática, uma topologia de rede corporativa altamente disponível que possibilite a análise do estado da arte em técnicas e ferramentas de gerenciamento de métricas de desempenho, eventos e informações de segurança. Serão descritas e analisadas potenciais vulnerabilidades em aplicações e sistemas implementados através de múltiplos cenários de ataque, propondo mitigações e melhorias cabíveis a cada caso estudado. Espera-se que este trabalho possa servir como base de estudo e pesquisa para profissionais da área.

ABSTRACT

Faced with the progressive growth in the use and importance of communication via the Internet, the amount of transmitted data has been increasing exponentially, whilst privacy and information security requirements in the transfer of this data have become frequent topics of concern, study and improvements. This work consists in the presentation of a proposal for an emulated virtual laboratory that implements, in a practical way, a highly available corporate network topology that allows a state-of-the-art analysis of performance metrics, event and security information management tools and techniques. Potential vulnerabilities in implemented applications and systems will be described and analyzed through multiple attack scenarios, proposing the appropriate mitigations and improvements for each case studied. It is expected that this work will be able to serve as a foundation for study and research from professionals of this field.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ACL	*Access Control List*
ANN	*Artificial Neural Network*
ANPD	Autoridade Nacional de Proteção de Dados Pessoais
API	*Application Programming Interface*
ARP	*Address Resolution Protocol*
AS	*Autonomous System*
ASBR	*Autonomous System Boundary Router*
ASN	*Autonomous System Number*
BGP	*Border Gateway Protocol*
CARP	*Common Address Redundancy Protocol*
CPU	*Central Processing Unit*
DAI	*Dynamic Arp Inspection*
DDoS	*Distributed Denial of Service*
DHCP	*Dynamic Host Configuration Protocol*
DMZ	*Demilitarized Zone*
DNAT	*Destination Network Address Translation*
DNS	*Domain Name System*
DoS	*Denial of Service*
EDR	*Endpoint Detection and Response*
FRR	*Free Range Routing*
FTP	*File Transfer Protocol*
HIDS	*Host Intrusion Detection System*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HyperText Transfer Protocol Secure*
ICMP	*Internet Control Message Protocol*
IDS	*Intrusion Detection System*
IEEE	*Institute of Electrical and Electronics Engineers*
IP	*Internet Protocol*
IPS	*Intrusion Prevention System*
ISP	*Internet Service Provider*
IoT	*Internet of Things*
KNN	*K-Nearest Neighbor*

KVM	*Kernel-based Virtual Machine*
LAN	*Local Area Networks*
LGPD	Lei Geral de Proteção de Dados Pessoais
MAC	*Media Access Control*
NAT	*Network Address Translation*
NFV	*Network Functions Virtualization*
NIDS	*Network Intrusion Detection System*
NIPS	*Network Intrusion Prevention System*
NOC	*Network Operations Center*
NSOC	*Network Security Operation Center*
ONF	*Open Networking Foundation*
OOB	*Out-of-Band*
OSPF	*Open Shortest Path First*
SDN	*Software Defined Networking*
SIC	*Security Intelligence Center*
SIEM	*Security Information and Event Management*
SLA	*Service Level Agreement*
SNAT	*Source Network Address Translation*
SOC	*Security Operations Center*
SSH	*Secure Socket Shell*
SSL	*Secure Sockets Layer*
SVM	*Support Vector Machine*
TCP	*Transmission Control Protocol*
TIP	*Threat Intelligence Platform*
TLS	*Transport Layer Security*
UDP	*User Datagram Protocol*
vCPU	*Virtual CPU*
VLAN	*Virtual Local Area Network*
VM	*Virtual Machine*
VPN	*Virtual Private Network*
VRRP	*Virtual Router Redundancy Protocol*
XSS	*Cross-Site Scripting*
WAN	*Wide Area Network*

Capítulo 1

Introdução

1.1 Contextualização

A Era da Informação manifestou um fato de extrema importância: dados possuem um valor inquestionável. Para cada pessoa conectada digitalmente no mundo, são transmitidas e computadas quantidades imensuráveis de dados a todo momento, sendo que o acesso à essas informações contém um valioso potencial de poder social e econômico.

Cada vez mais, grandes organizações investem recursos para o processo de obtenção de dados com diversas premissas, seja, por exemplo, para criar uma experiência cada vez mais pessoal e interativa com seus clientes. Quando se aborda os dados em um cenário voltado para segurança da informação, principalmente os dados relacionados aos usuários, os esforços se tornam ainda mais essenciais. O fato é que proteger os dados coletados é uma das principais e mais complexas prioridades e responsabilidades de qualquer organização, independente de seu tamanho.

Referindo-se a segurança no contexto de redes de computadores, é fundamental compreender que sua importância contempla desde prevenir vazamento de dados e evitar grandes prejuízos à organização e aos usuários, até mesmo garantir a proteção da rede contra invasões e ataques que prejudiquem o desempenho da mesma. Dada a quantidade de esforço realizado nessa área de atuação nas últimas décadas, é natural que existam profissionais dedicados e especializados atuando nas empresas, responsáveis por gerenciar o estado de eventos em tempo real, definir e implementar boas práticas, desenvolver e testar cenários de exploração de vulnerabilidades, aprimorar ferramentas e arquiteturas, entre outros aspectos voltados a tornar a fundação de segurança cada vez mais robusta.

Uma arquitetura de segurança implementada no mundo real envolve múltiplas camadas de ação, incluindo: conscientização constante de funcionários sobre prevenção de ataques; implementação de ferramentas de detecção, prevenção e gerenciamento de eventos; resposta ativa a vulnerabilidades; monitoramento de infraestrutura e serviços; entre outros. Essas camadas geralmente envolvem a ação de diversos profissionais de múltiplas áreas de tecnologia dentro da mesma empresa, o que pode tornar o processo burocrático e complexo.

Gerenciar uma infraestrutura de rede de maneira eficiente e segura sempre foi e, provavelmente, continuará sendo uma tarefa repleta de desafios. Os fatores de riscos são grandes e, com novas tecnologias emergindo a cada dia, para o "bem"e para o "mal", as possibilidades aumentam exponencialmente. Além da manutenção de uma boa fundação de segurança, a tomada de decisões de forma rápida e assertiva é essencial, afinal, qualquer vulnerabilidade explorada pode causar prejuízos inimagináveis à empresa e a seus usuários. Propor estudos e destacar esforços nessa área de atuação são necessidades de extrema valia para o contexto do mundo digital conectado, tendendo a se intensificar cada vez mais com a evolução da tecnologia e das possibilidades provenientes da mesma.

1.2 Motivação

A rápida evolução dos processos tecnológicos atuais e as consequências decorrentes disso são motivos de inquietação de muitas pessoas, até mesmo de quem não está diretamente interligado com estudos de tecnologia, por exemplo. O estado da atenção e da preocupação com a privacidade e com a segurança de dados está em sua máxima global, e essa inquietação tem gerado demanda de soluções cada vez mais complexas e imediatistas. Por um lado, este estado gera oportunidades de contratação e atuação de mais profissionais na área, porém é nítida a preferência crescente por profissionais mais experientes, criativos e capacitados.

Engenheiros de redes de computadores ou profissionais de áreas relacionadas estão constantemente propensos a encarar novos e árduos desafios na resolução de problemas encontrados no decorrer de sua carreira profissional, principalmente com o inesgotável crescimento da demanda na área. Contudo, através de comentários recorrentes destes profissionais em estágio de formação, percebe-se que há muitas lacunas de experiência a serem ocupadas para que os mesmos desempenhem suas futuras atividades com perícia e competência. Além disso, há uma grande falta na literatura de implementações práticas que contemplem todo o escopo de uma organização real. Normalmente, é possível identificar apenas partes, ou etapas, de uma topologia, de forma que se torna difícil e cansativa a agregação de conhecimento.

Além de dominar os conhecimentos e técnicas necessárias para criar e manter uma infraestrutura que possibilite a alta disponibilidade dos serviços, saber como garantir a integridade e a privacidade dos dados sensíveis gerenciados pelas aplicações, tal como definir estratégias e boas políticas de segurança, são conhecimentos de muito valor no mercado de trabalho. Há uma alta demanda por essa experiência, dado que estão decretados diversos cenários jurídicos que responsabilizam profissionais e organizações pela proteção e segurança de dados pessoais, como, por exemplo, a LGPD (Lei Geral de Proteção de Dados Pessoais) (GOVERNO FEDERAL, 2020). Portanto, é vital que estudantes e profissionais interessados tenham acesso aos ambientes necessários para praticar e desenvolver a implementação prática de seus conhecimentos teóricos.

Através deste trabalho, propõe-se a detalhada implementação prática em ambiente emulado de uma topologia de rede corporativa, que representa uma organização que realiza a gerência de um *campus*. Esta proposta envolve a implantação de protocolos, aplicações e ferramentas em um

ambiente de alta disponibilidade, visando a análise e monitoramento de *logs*, tráfegos, entre outros pontos. Dessa forma, disponibiliza-se um ambiente *sandbox* para exploração, testes e pesquisas relacionadas, proporcionando uma base para o estudo e crescimento de profissionais da área.

Este estudo se propõe a ser um meio de orientação e incentivo para que profissionais de áreas relacionadas sejam capazes de se aperfeiçoar e contribuir no caminho do seu próprio desenvolvimento e, consequentemente, na evolução da produção e do conhecimento científico em um contexto geral.

1.3 Objetivos

Pesquisar, analisar, testar e implementar protocolos, técnicas, sistemas, ferramentas e boas práticas amplamente aplicadas no mercado corporativo para definir a arquitetura, configuração e gerenciamento de uma topologia de rede corporativa. O foco é estabelecido na implementação de uma organização genérica que realize o gerenciamento de um *campus*, capaz de configurar e monitorar métricas de desempenho, eventos e informações de segurança em um ambiente de alta disponibilidade. Similar à implementação de um NSOC, é realizada a integração das áreas da organização, mantendo visibilidade total da topologia enquanto executada a detecção e análise de ataques aos sistemas e serviços, descrevendo e implementando formas de mitigação quando possível.

Como fruto da motivação do trabalho, foram selecionadas apenas soluções *open-source* e gratuitas para implementação, cumprindo os objetivos específicos da proposta:

- Software para emulação da topologia proposta.
- Roteador para implementação de protocolos BGP, NAT, entre outros.
- Switch com suporte ao protocolo OpenFlow para implementação da arquitetura SDN.
- Sistemas operacionais estáveis para criação de servidores e *desktops*.
- Sistema composto por softwares de exploração de vulnerabilidades e ataques.
- Ferramenta de monitoramento de métricas de desempenho dos ativos da rede através de visualização gráfica. Possibilitar a geração de alertas em ocorrências de comportamentos indesejados.
- Ferramenta para controle de inventário e estado de ativos da rede.
- Sistema SIEM que possibilite a coleta, formatação, agregação e enriquecimento de *logs* dos demais sistemas, além do gerenciamento de *hosts* e análise de vulnerabilidades dos mesmos.
- Sistema de *firewall*. Possibilitar a definição de regras e agregação de módulos IDS (*Intrusion Detection System*), servidor DHCP, servidor Syslog entre outros.

- Controladora SDN capaz de gerenciar a rede interna em configuração redundante de alta disponibilidade.
- Aplicações WEB e DNS.
- Sistema de *proxy* reverso.

Além dos objetivos descritos, está definida a implementação de configurações de segurança viáveis e necessárias em cada sistema, visando a definição de boas práticas e visibilidade aos ataques realizados. Para assegurar a alta disponibilidade do ambiente, também propõe-se realizar configurações relacionadas nos componentes disponíveis.

É importante definir também a qual escopo este presente estudo não se propõe a cumprir, incluindo:

- Desenvolvimento e hospedagem de aplicações com bancos de dados integrados.
- Implementação e análise de ataques complexos e diversos.
- Análise e correção de vulnerabilidades em todos os sistemas propostos.

1.4 Justificativa

É indiscutível a importância e a complexidade do gerenciamento e otimização que a infraestrutura de qualquer organização deve seguir para evitar distúrbios de desempenho e exploração de vulnerabilidades de segurança. Apesar disso, é adequado ressaltar que nenhuma ferramenta ou técnica implementada é suficiente para solucionar todos os possíveis problemas por completo. Sendo assim, seria ingênuo imaginar que não existem métodos desconhecidos, ou em desenvolvimento, para explorar e burlar vulnerabilidades nas tecnologias implementadas.

Dessa forma, a proposta deste trabalho visa realizar a validação das possíveis melhores técnicas e tecnologias passíveis de implementação para gerenciamento de eventos, informações de segurança e desempenho de dispositivos, permitindo controle de ocorrências e a resposta ativa, em alguns casos até mesmo de forma automatizada. É definido o que é de fato uma solução satisfatória e quais são as etapas e requisitos necessários para alcançar o resultado mais condizentes com este objetivo. Para isso, é necessário pesquisar e testar sucessivos recursos para especificação da solução de melhor eficácia, incluindo formas de escaneamento de vulnerabilidades, métodos de ataque e invasão, condensadores e analisadores de *logs*, entre outros.

Esta produção propõe-se a ser uma significante contribuição para o processo de estudo de novos profissionais das áreas relacionadas. Além dos resultados disponibilizados nesta produção, estará disponível diversos códigos, arquivos e referências para auxílio na reprodução das atividades e dos processos implementados.

1.5 Metodologia

O resumo da metodologia implementada neste trabalho está definida na Figura 1.1. Dada a extensão da implementação não foi possível descrever todos os tópicos em detalhe, mas estes já foram detalhados na descrição dos objetivos. No decorrer do desenvolvimento do projeto foram propostas melhorias e adaptações em processos aplicáveis como, por exemplo, na arquitetura e *design* da topologia, e essas decisões também estarão descritas neste trabalho.

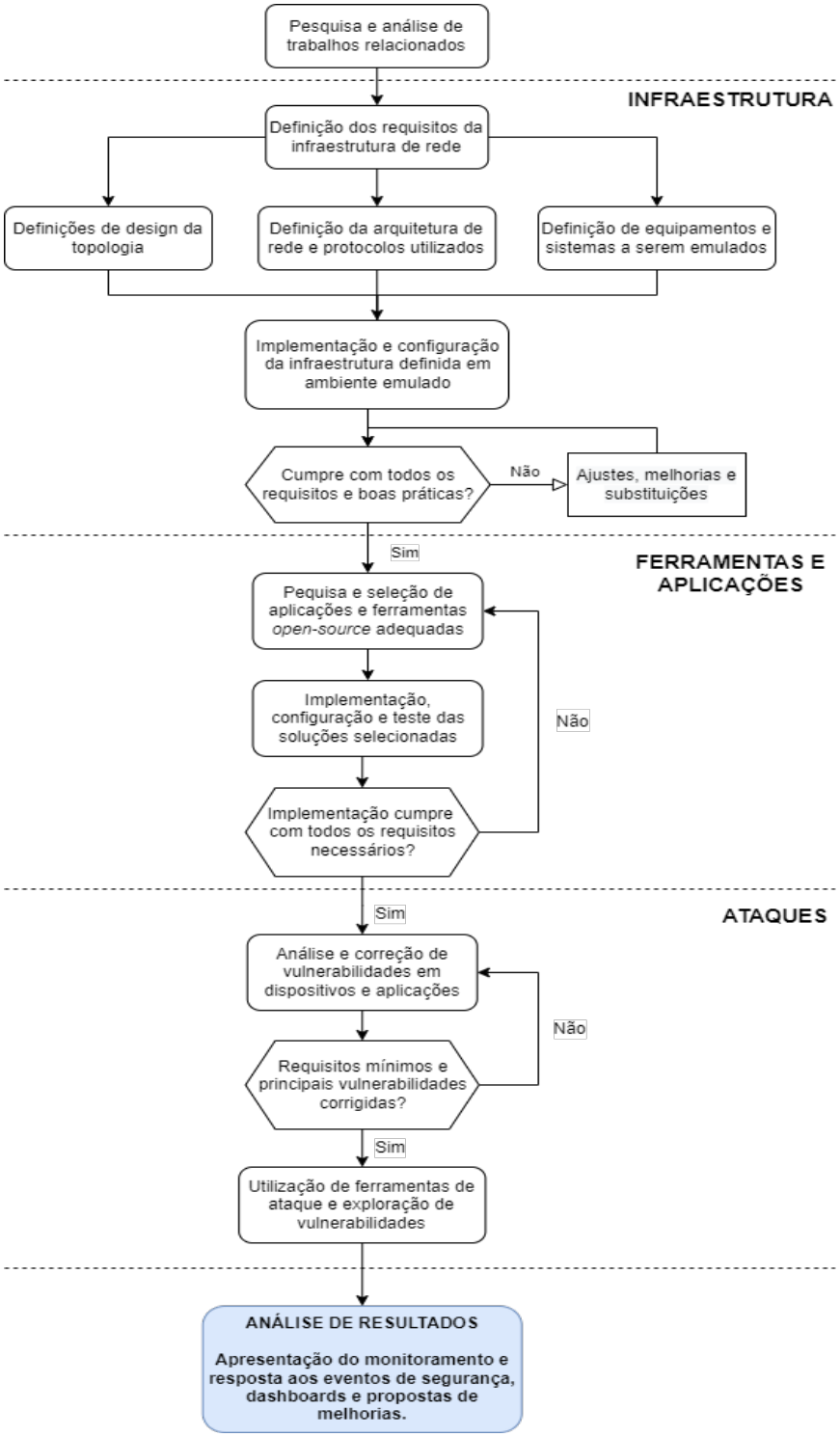


Figura 1.1: Metodologia descrita em diagrama de fluxo. Fonte: autores.

Capítulo 2

Fundamentação Teórica

2.1 Introdução

A pandemia do coronavírus provocou uma rápida migração de informações e processos para o meio digital, que ocorreu em um ritmo inimaginável até o momento. Diante disso, foi-se necessário adaptar a infraestrutura e o método de trabalho da maioria das empresas ao redor do mundo, principalmente empresas de pequeno e médio porte, ou seja, ainda em processo de expansão e modernização.

A ascensão do trabalho remoto e virtual desencadeou a demanda por mudanças na infraestrutura das redes. Torna-se evidente, por exemplo, como as pessoas começaram a exigir mais dos seus provedores de Internet (ISP's) em suas residências, como as empresas precisaram disponibilizar novos dispositivos para trabalho remoto (*notebooks*) e também providenciar a criação de servidores VPN para acesso seguro às redes internas. Em conjunto a essas rápidas transições em escala mundial, uma grande discussão quando se fala no gerenciamento de rede se intensificou: incidentes de segurança da informação.

Em um momento em que há trabalhadores, estudantes e outros profissionais em ambiente presencial, remoto e híbrido simultaneamente, é exposto um grande número de brechas de segurança e é responsabilidade das organizações prevenir e tratar destes casos. Para isso, é necessário cada vez mais a implementação de ferramentas e técnicas robustas para detecção de vulnerabilidades, prevenção e análise de incidentes de segurança, tanto na infraestrutura principal da organização quanto nos dispositivos fornecidos aos usuários, garantindo observabilidade e gerenciamento sobre todo o sistema e seus hospedeiros. Além disso, torna-se cada vez mais importante o treinamento e disseminação do conhecimento técnico para garantir que todos cumpram requisitos básicos de segurança, dado que, em muitos casos, a ocorrência de um incidente de segurança, e até mesmo o desvio dele, está nas mãos do usuário final que tem disponível o acesso à rede e aos sistemas da organização em questão.

Nesta seção serão expostos estudos e trabalhos correlacionados aos tópicos que serviram de base para alcançar os objetivos propostos deste projeto. Ressalta-se o foco especial na relação dos temas selecionados à segurança da informação, desenvolvendo como cada tópico se envolve nesta

temática.

2.2 NSOC (*Network Security Operations Center*)

Para aprimorar o rendimento e garantir ações mais eficientes, muitas organizações de médio e grande porte se estruturam em uma distribuição em equipes dedicadas, com especialistas focados em áreas específicas. Para gerência da infraestrutura e segurança de redes corporativas é essencial a existência de um NSOC (*Network Security Operations Center*).

O domínio definido como NSOC é uma integração de um setor de NOC, *Network Operations Center*, e um setor de SOC, *Security Operations Center*. Estas áreas têm suas responsabilidades comumente confundidas entre si, dada que a interdependência entre ambas é óbvia e necessária. Apesar disso, em suas respectivas caracterizações, suas funções são bem definidas.

De acordo com Awati (2021), o NOC é um setor que possui visibilidade completa sobre os ativos da rede da infraestrutura e torna capaz o gerenciamento e controle dos mesmos. Essa condição possibilita a gestão de performance e qualidade da rede e suas aplicações; manutenção e análise dos *firewalls*, servidores, domínios, bancos de dados, política de grupo para aplicações, entre outros; controle da política de SLA (*service level agreement*); instalação e manutenção de *software*; realização do ciclo de vida de *backup*; e também serve como primeira linha de defesa em relação à segurança da informação, podendo identificar, isolar e responder à tais eventos de forma adequada, sendo esta resposta um encaminhamento para a área de SOC, por exemplo.

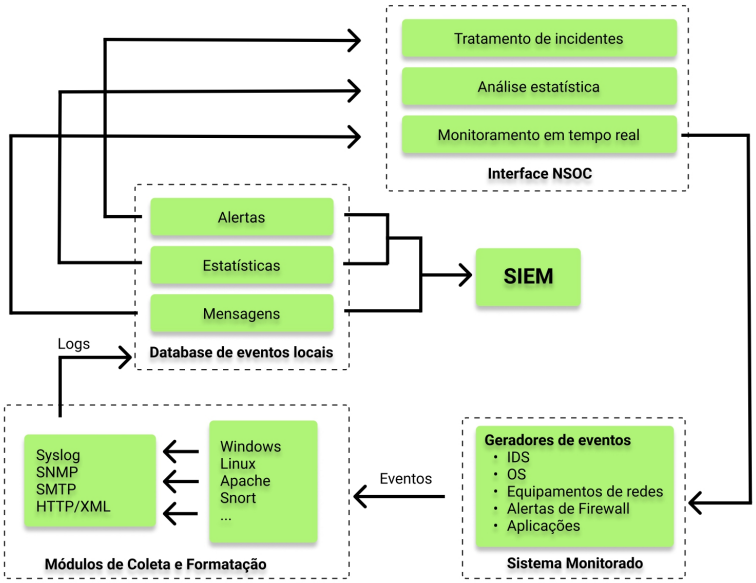


Figura 2.1: Diagrama reduzido do fluxo de operação de um NSOC, baseada no fluxo proposto em (BIDOU, 2005, Fig. 2).

Um SOC, através das definições de McAfee (2021), por sua vez, realiza o gerenciamento de eventos de segurança e é responsável pela prevenção, detecção, análise e resposta à estes eventos.

Muitos destes processos são automatizados através de um sistema SIEM (*Security Information and Event Management*) e cabe aos especialistas presentes no SOC a reação e a posterior investigação dos eventos ocorridos através dos dados automaticamente correlacionados pelo sistema. Para uma melhor definição, um SIEM é composto de diversos módulos, incluindo partes dos sistemas responsáveis por: escaneamento de aplicações e bancos de dados; sistemas de detecção de intrusão (IDS) e sistemas de prevenção de intrusão (IPS); detecção e remediação contínua de *endpoints* (EDR); plataformas de inteligência contra ameaças (TIP); coleta e análise de *logs*;entre outros importantes módulos. Vale ressaltar que uma evolução progressiva de um SOC é definida como SIC (*Security Intelligence Center*), responsável por um foco maior no estudo de ameaças globalmente conhecidas e na realização de atualizações e manutenções preventivas dos sistemas. Contudo, para este trabalho, tal definição será incluída na definição de SOC e posteriormente na definição de NSOC, de forma a simplificar as nomenclaturas. (MCAFEE, 2021)

Anomalias e comportamentos indesejados são eventos comuns e fazem parte do cotidiano dos analistas. Devido a isso, é essencial a presença de uma equipe técnica qualificada presente em suas funções 24 horas por 7 dias da semana preparados para controle e remediação destes casos (AWATI, 2021; MCAFEE, 2021). Novamente, reitera-se a interdependência entre ambas as áreas, sendo o NOC uma base para acesso do SOC à todos os dispositivos da infraestrutura e o SOC uma entidade garantidora do controle de incidentes de segurança e certificadora de que tais eventos não causarão prejuízo à performance da rede. Uma visualização reduzida da função dessa integração em NSOC pode ser visualizada na Figura 2.1.

A organização SANS *Institute* realiza diversas pesquisas e entrevistas voltadas ao estudo de SOCs e outras áreas da segurança da informação. Em 2019, a SANS realizou uma pesquisa (CROWLEY; PESCATORE, 2019) relacionada às mais comuns e melhores práticas implementadas em um SOC e, através dos resultados dessas entrevistas direcionadas à profissionais que atuam nesse setor, concluiu-se que a grande maioria das organizações não possuem um NOC dedicado ou não são estabelecidas interações com frequência; será iniciada ou continuada a migração de SOCs centralizados na organização física para serviços disponíveis em *cloud*; as funções de testes de penetração, estudo de ameaças e investigação forense digital, entre outros processos mais investigativos e relacionados à análise, geralmente são terceirizados, mantendo apenas os processos mais diretos como funções do SOC; insatisfação com a performance de ferramentas de gerenciamento de inventário, prevenção de perda de dados, soluções de detecção com *machine learning* e tecnologias para engano de atacantes, entre outras; satisfação com ferramentas de VPN, SIEM, *firewall*, IDS/IPS, EDR, proteção contra DoS/DDoS, entre outras.

Estas respostas são de grande interesse na definição de escopo durante a implementação de novos SOCs ou para aprimoramento de SOCs já existentes. A análise negativa para a relação entre NOC e SOC também relata um grande problema nas organizações, pois essa falta de comunicação impede a disseminação de informação e o consequente prejuízo na entrega de soluções eficientes e rápidas. Neste trabalho, houve um esforço para garantir a correlação entre ambos setores e, por isso, utiliza-se a nomenclatura de NSOC, uma área que integra ambas responsabilidades de forma mais proveitosa possível.

Uma nova pesquisa realizada pela SANS em 2020 (PESCATORE; FILKINS, 2020), iniciada dias antes da definição do estado de pandemia global, trouxe novos paradigmas e resultados em relação à 2019, dado o estado global completamente distinto. Nesta pesquisa, concluiu-se que a busca de profissionais com experiência em operações de segurança, principalmente segurança em *cloud*, cresceu significativamente, ressaltando também que houve aumento na busca por provedores de serviços externos para testes de penetração e investigação de eventos. Neste cenário e progressivamente mais com a modernização dos processos, a atuação em segurança da informação será mais requisitada e se tornará um desafio cada vez mais complexo. Em razão disso, muitas organizações recorrem à alguma forma de auxílio externo, como infraestrutura em *cloud* ou outras formas de provedores de serviços.

2.3 Ataques e Incidentes de Segurança

A democratização ao acesso à tecnologia e virtualização de processos diários trazem excepcionais benefícios e possibilitam acessibilidade global à Internet e a todo conhecimento ali disseminado. Contudo, novas formas de aproveitamento de vulnerabilidades para realização de atividades ilícitas também são constantemente desenvolvidas e distribuídas em uma velocidade cada vez maior. Através da conexão global pela rede de computadores, qualquer dispositivo conectado está suscetível a ser um alvo de ataques, por isso é necessário que sejam feitos testes, análises e atualizações constantes nos sistemas de forma a garantir a segurança frente das vulnerabilidades, mesmo que desconhecidas.

Para que um sistema seja classificado como "seguro", é necessário que se realize, além de toda a implementação de métodos de identificação, análise e resposta à incidentes de segurança, a validação efetiva contra ataques. A MITRE, um dos centros de pesquisa e desenvolvimento fundadas pelo governo dos Estados Unidos, desenvolveu uma ferramenta chamada MITRE ATT&CK (*Adversarial Tactics, Techniques, and Common Knowledge*) (MITRE, 2021), que disponibiliza uma enorme base de dados de táticas e técnicas utilizadas por atacantes para reconhecimento, acesso e execução de operações desautorizadas e ilícitas. Esta ferramenta oferece uma matriz que correlaciona tais métodos com formas de detecção e mitigação e, além de definir uma taxonomia comum para a comunidade, facilita com que analistas de segurança (ou até mesmo equipes internas de investigação e de teste de penetração) validem a topologia implementada contra as vulnerabilidades mais conhecidas e utilizadas globalmente. Na Figura 2.2, pode ser visualizado um exemplo de percurso nas três primeiras camadas do *framework* do ATT&CK.

De acordo com Georgiadou, Mouzakitis e Askounis (2021), apenas no mês de março de 2020 houve um aumento de 400% nos casos de fraudes cibernéticas, resultando em impactos como destruição de dados, perda de produtividade, roubo de propriedade intelectual, perdas financeiras, exposição de dados sensíveis, entre outros. Complementa-se à este resultado a afirmação de que as organizações estão demasiadamente concentradas em ferramentas e sistemas de detecção e defesa de ameaças e acabam deixando de lado o estudo e investigação de ameaças, ataques e vulnerabilidades, sendo estes essenciais para a prevenção e proatividade de resolução de incidentes de segurança.

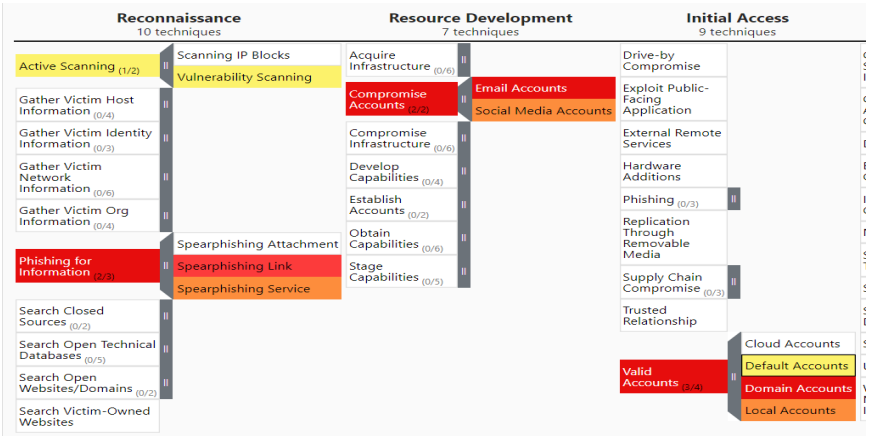


Figura 2.2: Exemplo de percurso apenas pelas três primeiras fases do processo de invasão, elencando vulnerabilidades através de uma ferramenta disponibilizada por (MITRE, 2021).

Ainda em Georgiadou, Mouzakitis e Askounis (2021), foi realizada uma pesquisa cujos resultados indicam que a maioria das empresas de média e grande porte entrevistadas utilizam da ferramenta ATT&CK para proteção contra ameaças, análise de vulnerabilidades nas soluções implementadas, implementação de políticas de segurança e modelagem de ameaças para identificação e estudo.

O aumento expressivo de incidentes de segurança é preocupante, principalmente em casos repentinos, muitas vezes modificando estruturas dentro das organizações e até em meio à sociedade, dependendo da magnitude da ocorrência. Em Khan, Brohi e Zaman (2020), verificou-se as principais formas de ataques cibernéticos realizados durante a pandemia de Covid-19, incluindo: DDoS, *malware*, *ransomware*, domínios e sites maliciosos, *phising*, entre outros. Apesar destas técnicas parecerem extensivamente aplicadas e conhecidas, é fácil para que em um momento de descuido ou por uma ação de um usuário leigo, seja aberta uma brecha de acesso, por exemplo, via *phising*, pelo fornecimento de credenciais por um *link* forjado, semelhante ao da organização. Mesmo em casos simples como este, a injeção de um *ransomware* ou outra forma de *malware* na rede podem causar inimagináveis prejuízos financeiros e jurídicos. Dessa forma, deve-se atentar que a segurança não é uma responsabilidade apenas da área de SOC, e sim da organização como um todo. Assim como descrito em Georgiadou, Mouzakitis e Askounis (2021), a organização deve promover e assegurar mecanismos de *backup*, controle de acesso, análise de risco, programas de treinamento e capacitação, testes de penetração, entre outros. No caso dos funcionários, é necessário apresentar certificação de treinamento e resultados, conhecimento sobre políticas e processos, testes de habilidades de segurança, deixando claro que todos são responsáveis pela manutenção da qualidade do estado de segurança da informação.

Análises da literatura atual (HUMAYUN et al., 2020) também apresentam os tópicos mais estudados pelas maiores organizações de desenvolvimento científico relacionados à área, como o IEEE, e estes incluem com maior frequência os tópicos: DoS, *malware*, *phishing*, *SQL injection*, *sross-site scripting*, entre outros. Comparando este resultado com os tipos de ataques mais ocorridos durante os últimos anos, nota-se que há pouca variedade de ataques em estudo e disso pode surgir a indagação de como tais formas de ataques continuam eficientes. A falta de capacitação das pessoas em relação à segurança da informação e a evolução constante das técnicas ilícitas de

ataques podem explicar esse fenômeno. Novamente em Humayun et al. (2020), é descrito que as técnicas mais utilizadas para mitigação de ataques são: *IDS*, *firewalls*, técnicas de *anti-phishing*, detecção de anomalias, *software anti-malware*, *sandboxing*, entre outros. Mais uma vez, as técnicas de mitigação também não são inéditas e refletem, em sua maioria, configurações básicas, desde a mais simples infraestrutura até a mais complexa destas, como um *datacenter*. Contudo, ressalta-se que, muitas vezes, a presença de vulnerabilidades se dá nas minúcias e nos pontos mais desprezados, fazendo com que essa dedicação frente ao estudo e análise constante das soluções implementadas diferencie, por exemplo, um *firewall* desprotegido devido à má configuração de outro que abrange até mesmo regras preventivas de ataques nunca recebidos.

2.4 Gerenciamento de Sistemas e Geração de Eventos

Topologias, infraestruturas e diferentes sistemas estão constantemente em expansão, através de novos dispositivos e tecnologias, aumentando a complexidade da rede como um todo. De maneira equivalente, o gerenciamento e a monitoração dessas redes deve acompanhar o avanço, de maneira a não comprometer os sistemas. A boa gestão de uma rede abarca tanto o gerenciamento da segurança da informação quanto da geração de eventos, funções estas que podem inclusive ser acopladas em um único sistema (MAGANE ENGINE, 2021) (ROSENCRANCE, 2020).

De acordo com Bidou (2005), eventos são compreendidos como uma mudança de estado em determinado componente e podem ser gerados de acordo com uma operação específica no sistema, nas aplicações ou ao longo da rede, como no caso dos sensores, por exemplo. Também é possível gerar eventos como uma reação a um estímulo externo da rede, como um *ping* por exemplo. Eventos são importantes fontes de dados no sistema, desde *logs* a fluxos de informação. A monitoração de eventos retém sua relevância tanto em um cenário de condições excepcionais como no dia a dia da organização. Isso significa verifica sistematicamente os serviços e seus componentes, registrar e reportar mudanças de estado identificadas como eventos. Assim, é possível determinar respostas apropriadas para eventos de infraestrutura, de aplicação, segurança, inclusive falhas de segurança (BARANES, 2019).

Usualmente, eventos podem ser informativos, alertas ou exceções. Eventos informativos apenas indicam mudanças que não requerem ações específicas por parte da gerência. Quando se fala de alertas, eventualmente alguma operação deve ser necessária para não comprometer o funcionamento dos sistemas. Mas em se tratando de exceções, é necessária uma operação imediata, com intuito de identificar e resolver o incidente (BARANES, 2019). Ao gerenciar um sistema, é inevitável ter que lidar com falhas. Para isso, é recomendável que a equipe responsável tenha um sistema e um plano de ação para essas situações, englobando a detecção de eventos, isolamento e notificação, assim como a eventual resolução (MAGANE ENGINE, 2021).

Pode-se conceituar dois modelos de monitoração de uma rede: ativa ou passiva (MAGANE ENGINE, 2021). Uma monitoração ativa remete à detecção proativa dos sistemas através da marcação de limiares de operação, como é o caso do *ping* ICMP, por exemplo. Já uma monitoração passiva sugere a escuta pelo evento, como no caso do *Syslog*. Após detectar os eventos, no caso de

uma vulnerabilidade, é essencial isolar o problema em si, isso é, identificar entre os eventos coletados o que pode ser considerado o potencial incidente, notificar os administradores e estabelecer as ações a serem tomadas de acordo com o ocorrido (MAGANE ENGINE, 2021).

É válido considerar a importância do emprego de uma boa ferramenta de gerência de eventos. Com o uso do gerenciador, é possível aprimorar a detecção de eventos e o tempo de resposta, de maneira até mesmo a evitar que alertas se tornem grandes fragilidades na rede. Também é viável estabelecer fluxos de execução de ações baseado no perfil de evento identificado. Isso quer dizer que, de acordo com o tipo de evento, o sistema aciona automaticamente quais ações são necessárias, como notificar administradores, realocar o problema para diferentes equipes, comutar dispositivos, entre outros, o que garante acurácia no manuseamento de um evento. A criação de relatórios de desempenho também é parte fundamental do gerenciamento de uma arquitetura, principalmente de larga escala. Através de relatórios formulados, podem ser estabelecidas zonas de fragilidade na topologia em questão (BARANES, 2019).

Logs de eventos indicam todos os acontecimentos registrados nos dispositivos e ativos da rede. Um servidor *Syslog* pode ser um grande diferencial na rede, de maneira a centralizar e uniformizar todos os *logs* obtidos ao longo da topologia, unificando, assim, a monitoração. É através das informações coletadas nos *logs* que a ferramenta de monitoração determina alarmes, disparando alertas aos administradores (MANAGE ENGINE, 2021).

Nesse contexto, compreende-se a importância de estabelecer uma abordagem de segurança conhecida como SIEM (*Security Information and Event Management*) (REBACK, 2020). Um gerenciamento baseado no conceito de SIEM corresponde em integrar tanto políticas de segurança da informação quanto segurança de eventos em um mesmo centro de gerência da rede. Ferramentas SIEM fornecem relatórios unificados dos eventos e incidentes voltados à questões de segurança, assim como alertas, quando identificadas vulnerabilidades no sistema. Através dessa centralização e normalização dos dados, há uma melhora significativa na identificação de problemas e no tempo de resposta. Dessa forma, é possível visualizar *logs*, tanto de infraestrutura, aplicações e rede, obtidos por todos os ativos do sistema, em uma única interface, o que facilita traçar o comportamento de invasores e aplicações afetadas, por exemplo. Além disso, a auditoria do sistema se torna muito mais simples (REBACK, 2020) (ROSENCRANCE, 2020).

Com o desenvolvimento de novas tecnologias, o volume de dados e a complexidade dos ataques cresce exponencialmente. Logo, essas ferramentas de segurança também precisam acompanhar as mudanças. Por exemplo, há uma forte necessidade de integrar os *logs* centralizados de um sistema com dados na nuvem, que garantem importantes métricas para a gerência. Outra forma de ilustrar a importância de buscar inovações é a integração de dispositivos IoT, que correspondem a um aumento significativo nos pontos ativos das redes, abrindo o perímetro de vulnerabilidades (REBACK, 2020). De forma sintetizada, é importante que, ao gerenciar uma rede, os administradores se preocupem em construir um modelo de confidencialidade, integridade e disponibilidade dos dados que circulam os ativos (FORCEPOINT, s.d.).

2.5 Coleta e Análise de Dados

A discussão a respeito do fluxo de dados de uma rede requer mencionar a importância da segurança desses dados. Garantir a segurança dos dados é uma obrigação legal de toda organização, pública ou privada. Esse debate abarca desde arquivos, bancos de dados até aplicações, isso é, todo o processo de obtenção dos dados, coleta e armazenamento, até eventual análise. As boas práticas se iniciam já no processo de coleta, pois é essencial que os dados coletados sejam relevantes e confiáveis para qualquer análise ou necessidade futura (HARRINGTON, 2021) (CISCO PRESS, 2015).

No contexto de monitoração de uma rede, os dados são provenientes de diferentes fontes, com múltiplos formatos, como *logs*, pacotes, fluxos. Aplicações compreendidas com "coletoras"de informações tratam seus dados de maneira independente. Portanto, para um gerenciamento mais assertivo da rede como um todo, é necessário garantir a correta formatação e identificação dos dados previamente ao seu armazenamento. Uma arquitetura escalável permite uma distribuição de coleta e armazenamento de dados por toda a rede, garantindo melhor performance e disponibilidade de dados, o que pode ser decisivo ao enfrentar um incidente de segurança (BIDOU, 2005).

A correta categorização dos dados pode ser um grande diferencial na detecção de ataques e vulnerabilidades da topologia, dado que cada tipo de dado remete a diferentes aplicações. Uma maneira eficiente de classificar os dados coletados é fundamentada pelos métodos de coleta. Além disso, é fundamental a compreensão de que tanto os dados de conteúdo quanto dados estatísticos coletados na rede são de extrema importância para uma análise e monitoração eficiente (JING; YAN; PEDRYCZ, 2019).

Pacotes correspondem a um conjunto de dados tanto de controle quanto de conteúdo. Pacotes podem ser gerados quando aplicações utilizam diferentes protocolos, como TCP, UDP e ICMP. Esses pacotes podem ser capturados na interface física, através de um *sniffer*. Existem diferentes aplicações capazes de realizar essa coleta de pacotes, como Wireshark, TCPdump, Snort, Nmap, cada qual com múltiplas ferramentas e métodos de coleta. No contexto de captura de pacotes, é muito relevante reter a informação de endereços de origem e destino. Através da análise desses endereços, é possível, por exemplo, identificar um ataque devido à alta concentração de pacotes com mesmo endereço IP de origem. O mesmo vale para informações como tamanho dos pacotes, tempo de vida, portas de origem e destino e até quantidade de pacotes, visto que um aumento drástico no número de pacotes pode ser um indicativo de um ataque de inundação DDoS (JING; YAN; PEDRYCZ, 2019).

Fluxo pode ser compreendido como uma corrente de pacotes com um ou mais atributos similares, as chamadas chaves de fluxo. De acordo com as necessidades do administrador, é possível identificar diferentes fluxos a partir de suas chaves. Os fluxos são de extrema importância para o monitoramento da rede, de aplicações, *hosts* e identificação de vulnerabilidades. De acordo com o método utilizado para coleta de fluxos, importantes informações estatísticas podem ser geradas para análise, como contador de fluxos, tipos, tamanho, direção e duração. Essas são importantes métricas para detectar um incidente de segurança, por exemplo, no caso do tamanho dos fluxos,

durante um ataque de DDoS, a extensão do fluxo do ataque é consideravelmente maior que a de fluxos legítimos. Uma análise relevante de ser realizada quando se especifica um ataque é que, em um cenário regular, o número de fluxos de requisição, através de algum protocolo ou porta específicos, equivale aos fluxos de resposta de mesma chave. Portanto, mudanças nessa relação podem indicar distúrbios na rede (JING; YAN; PEDRYCZ, 2019).

Já uma **conexão** equivale ao tráfego existente entre dois endereços IP, um interno e outro externo, dependendo da rede analisada. Só é possível considerar que uma conexão foi estabelecida após a troca de pelo menos dois pacotes. Portanto, é possível obter importantes estatísticas desse tráfego, como a quantidade de pacotes e fluxos na conexão, a duração da conexão entre os *hosts*, tipo de conexão. Mais uma vez, ao realizar uma coleta recorrente de informações sobre as conexões existentes na rede, é viável operar análises que evidenciam possíveis ataques (JING; YAN; PEDRYCZ, 2019).

Diferente das categorias anteriores, cujos dados eram coletados nos dispositivos da rede, os dados do hospedeiro são coletados diretamente no *host* local. Muitas ofensivas tem por objetivo interferir na performance dos *hosts*. Os dados coletados no nível do hospedeiro possuem importante valor para identificar adulterações em arquivos de sistema, escala de privilégios, acessos e *logs* não autorizados. O uso de CPU e memória pelo sistema e pelo usuário evidenciam métricas significativas, visto que, no caso de um ataque de DDoS, o uso da CPU pelo sistema e uso da memória apresentam um crescimento excepcional. Diferentes *logs* também são gerados através dos *hosts*, tanto *logs* de operação de aplicações como de equipamento. Os *logs* de operação de equipamentos englobam informações geradas pelos equipamentos atrelados ao *host*, como sensores, cursores, entre outros. Já os *logs* de operação de aplicações compreendem os dados relacionados às ações de um usuário ao executar uma aplicação específica, isso é, em grande parte, a criação de eventos (JING; YAN; PEDRYCZ, 2019).

Considerando toda a variedade de dados que podem ser coletados no sistema, é essencial que se estabeleça métodos de análise escaláveis e flexíveis, desde análises estatística até *machine learning*. Em se tratando de uma análise estatística, é necessário estabelecer um perfil de comportamento do tráfego que seja considerado usual. A partir desse referencial, qualquer atividade cujas métricas desviem do esperado, um alarme de segurança deve ser gerado. Um ataque acometido por um *Botnet*, por exemplo, pode causar certas anomalias nos padrões da rede e ser detectado por uma análise estatística de distribuição de tráfego (JING; YAN; PEDRYCZ, 2019).

Já uma análise pautada no uso de *machine learning* procura fundamentar modelos de padrões analisados. Muitos algoritmos de aprendizado supervisionado são utilizados na análise de segurança de redes, como Máquinas de Vetores de Suporte (SVM), Redes Neurais Artificiais (ANN), K-Vizinhos mais proximos (KNN), entre diversos outros. Também são empregados algoritmos de aprendizado não supervisionado e, no geral, técnicas de *machine learning* apresentam taxas de detecção de ataques consideravelmente elevadas. Em contra partida, essas são técnicas que consomem grande parte dos recursos do sistema, tanto nas etapas de treinamento quanto execução (JING; YAN; PEDRYCZ, 2019).

2.6 SDN

Os diversos avanços tecnológicos recorrentes nos últimos anos demonstram como eles estão amplamente correlacionados com a crescente iniciativa de automação de processos. A utilização de infraestruturas e serviços em *cloud* e a manipulação de técnicas de *scripting* para execução automatizada de ações são exemplos de como a automação e, consequentemente, a simplificação de processos, são essenciais para a redução do tempo gasto com práticas manuais, extensas e complexas, restando mais períodos livres para o foco em atividades mais relevantes. A arquitetura SDN (*Software Defined Networks*) propõe-se a facilitar o gerenciamento de redes através da automação e da agilidade para manutenção de redes (AHMAD et al., 2015).

A arquitetura SDN (AHMAD et al., 2015) realiza a divisão da rede em dois pontos, um plano de controle e um plano de encaminhamento. No plano de controle é realizada toda a configuração de lógica, políticas e regras, restando ao plano de encaminhamento apenas a aplicação destas configurações através das tabelas de fluxo dinâmicas, de certa forma análogas a uma tabela de roteamento. Nesta arquitetura, o plano de controle é centralizado e permite a configuração dinâmica de todo o plano de encaminhamento.

O protocolo Openflow (MCKEOWN et al., 2008) é a implementação mais utilizada desta arquitetura, permitindo a configuração do plano de controle via API, geralmente realizada através do auxílio de aplicações que suportam este protocolo no plano de aplicações. Dessa forma, a configuração e controle de toda a rede está disponível ao administrador da mesma (AHMAD et al., 2015). Embora essa arquitetura seja orientada para utilização em redes LAN, a visibilidade completa do estado da rede garante observabilidade de áreas não facilmente monitoradas em uma arquitetura tradicional, permitindo uma reação ágil no caso de um comportamento indevido da rede, seja por performance de protocolos ou por algum incidente de segurança.

Embora disponha de um inovador paradigma para a arquitetura de redes, a solução SDN também produz uma nova série de possíveis vulnerabilidades de segurança. Apesar de possibilitar a orquestração de diversas ferramentas e *frameworks* pelo plano de aplicação do Openflow, como *firewall*, IDS/IPS, monitoramento de tráfego, controle de acesso, inspeção de conteúdo e amostragem de fluxos, os planos podem estar vulneráveis à inserção de regras de fluxo maliciosas por aplicações comprometidas com acesso direto ao plano de controle, por ataques DoS no plano de controle, devido à centralização, escalabilidade e disponibilidade desse plano, por ataques de alagamento de regras de fluxo no plano de dados, entre outros disponíveis em (AHMAD et al., 2015, Table 1). Apesar das formas de ataques serem diferentes de uma arquitetura tradicional, a concepção é semelhante e é indispensável a noção de que nenhuma solução é independente destes problemas. Contudo, as ferramentas para prevenção desses casos existem e estão cada vez mais sendo aprimoradas colaborativamente com o esforço da comunidade, dada a inclinação *open-source* da arquitetura SDN e do protocolo Openflow.

O real valor desta arquitetura origina da capacidade de programabilidade do comportamento da rede, mesmo em casos desafiadores, como incidentes de segurança, falhas em equipamentos físicos ou até mesmo erros de configuração (AHMAD et al., 2015). Como exemplo, em uma

situação em que é identificada a origem de um ataque, é possível automaticamente definir o fluxo de comportamento da rede nesta situação, bloqueando a regra de fluxo específica ou redirecionando o tráfego do atacante para outra rota, tomando medidas cabíveis para posterior análise.

Por fim, esta inovação em SDN ainda está em desenvolvimento e passando por diversos processos de padronização pelos esforços da ONF (*Open Networking Foundation*), mas vale ressaltar que tal tecnologia apresenta promissoras propostas para o futuro, vide esforços do Facebook, Google, Microsoft, Cisco, entre outras empresas, na implementação e desenvolvimento de novas soluções utilizando essa solução.

2.7 Trabalhos Relacionados

Neste estudo, planejou-se a implementação de uma metodologia similar à de um NSOC para o gerenciamento dos eventos, informações e métricas internas à estrutura desenvolvida. Hae et al. (2016) descreve os benefícios da visão holística de toda a topologia para um eficiente gerenciamento da mesma. Em casos reais, a definição de equipes específicas e bem treinadas para o trabalho em um NOC e SOC são essenciais, porém há uma perda significativa de performance e eficiência gerada pela burocracia entre a comunicação dessas esquipes, sendo então um NSOC a convergência inteligente desses setores.

Nesse aspecto, o estudo proposto cumpre com os requisitos da implementação de um NSOC de forma prática contemplando a centralização dos processos, análise dos dados coletados e a resposta à incidentes. A sincronização e sinergia das ferramentas implementadas, o gerenciamento centralizado de incidentes e o enriquecimento de dados em formato padronizado, assim como descritos por Hae et al. (2016), são definições essenciais dessa implementação.

A referência inicial para a realização do estudo apresentado neste documento segue de acordo com Brezula (2017), infraestrutura essa que fundamentou a construção da topologia a ser apresentada. Na proposta de Brezula (2017), foi construída uma rede dividida em 4 áreas, *campus*, *datacenter*, DMZ e ISPs. A divisão por áreas evidenciou uma estrutura similar ao utilizado por redes corporativas reais, sendo a implementação do autor uma versão compacta das mesmas. Por esse motivo, essa arquitetura foi utilizada como um ponto inicial de estudo.

Para a estruturação do campus, Brezula utiliza o conceito de arquitetura *3-tier* (CISCO, 2008a). Através da implementação dessa arquitetura, o autor discute a divisão das camadas de acesso, distribuição e *core* na área de campus, isso é, a porção da infraestrutura que prove o acesso a serviços e recursos aos usuários e dispositivos finais (CISCO, 2008a). A topologia se conecta à internet através de dois *Internet Service Providers*, ISPs, diretamente ligados a um roteador de borda com links diretos para cada um dos *firewalls* de entrada das áreas descritas. A partir da estrutura implementada por Brezula, foi possível propor mudanças e melhorias em contribuição com outros estudos, dadas as limitações.

No decorrer do desenvolvimento deste trabalho, migrou-se de uma arquitetura de redes legado que utilizava, por exemplo, de roteamento OSPF entre as áreas da organização para uma arquite-

tura de redes SDN híbrida. Essa mudança se deu, além da motivação do estudo de uma tecnologia em ascensão, pelos benefícios dessa implementação, incluindo configuração e gerenciamento centralizado da rede, facilidade de implementação de redundância na rede, performance, entre outros. Hong et al. (2016), por exemplo, avalia como com apenas 20% de migração para uma arquitetura SDN, pode-se reduzir em média 32% do uso máximo de um enlace em arquiteturas SDN híbridas para ambientes *enterprise* e para ISPs.

Nguyen e Kim (2016) citam mais alguns benefícios da arquitetura SDN em campus e ambientes *enterprise* e propõem um módulo novo para gerenciamento de VLANs através do controlador FloodLight. Neste trabalho, será apresentado o gerenciamento de VLANs, regras de ACL, controle de redundância e outras técnicas através do controlador Faucet que, de acordo com as pesquisas realizadas, ainda não foi avaliado de forma prática no contexto científico.

A implementação de ferramentas e sistemas como soluções de *firewall*, IDS, IPS, ACL e EDR são indispensáveis para manter a segurança e disponibilidade dos servidores e dos *hosts* internos. Tal como descrito por Scarfone e Hoffman (2009), há uma longa lista de documentos que descrevem boas práticas, requisitos mínimos e requisitos recomendáveis para implementações dessas ferramentas e sistemas. Embora seja impossível ter comprovação da eficiência total de um sistema, dada a emergente inovação de técnicas de invasão e exploração de vulnerabilidades, é imprescindível que se faça o máximo possível para barrar atividades maliciosas. Neste trabalho foram explorados e implementados os sistemas PfSense, Suricata, Wazuh Agent, NGINX e funcionalidades da controladora SDN Faucet de forma prática, seguindo com as melhores definições de boas práticas documentadas publicamente, para cumprir com esse objetivo.

Para contemplar o objetivo de gerenciamento de eventos e informações de segurança deste trabalho é essencial a implementação de uma sistema SIEM na topologia. Assim como descreve Žgela e Penga (2019), o escopo de atuação de um sistema SIEM é extremamente extenso e complexo, contemplando técnicas já descritas, como o monitoramento em tempo real de um grande número de eventos, sua correlação e as respostas automáticas à possíveis incidentes. Žgela e Penga (2019) também informam, através de uma versão anterior do relatório de brechas de segurança da *Verizon* desenvolvido por Widup et al. (2021), que mais de 28% dos casos de incidentes de segurança com consequência de vazamento de dados envolve atores internos da organização. Sendo assim, é essencial a implementação de um sistema que seja capaz de lidar com o gerenciamento de comportamento e permissões de cada usuário, por exemplo. Por fim, Žgela e Penga (2019) também descrevem casos, identificados pelo *logs* utilizados em estudo e analisados via SIEM, em que se percebeu o comportamento anormal de funcionários pelo IP de onde estes se conectavam dentro da organização e também a adição de sites em *blacklists* após o *download* de *malwares* identificáveis por outros funcionários.

Contudo, assim como cita Vazão et al. (2019), verifica-se que as soluções líderes de mercado são pagas e, além disso, possuem preço elevado até mesmo para pequenas e médias empresas. A empresa *Gartner*, que inclusive criou o termo SIEM, é quem realiza esses relatórios esporádicos, porém não inclui a avaliação de soluções *open-source*. Vazão et al. (2019) contribui realizando e testando algumas das principais sistemas SIEM *open-source*, como OSSIM, Splunk Free, Graylog,

ELK Stack, indicando a ELK Stack como principal em escalabilidade e flexibilidade. Complementando, Cózar (2020) acrescenta a avaliação de outras ferramentas como Prelude OSS e Wazuh, indicando o Wazuh como melhor escolha dentre as opções.

Assim como descreve Cózar (2020), esse sistema é totalmente gratuito e *open-source*, é facilmente escalável dado sua integração com ELK Stack, possui documentação extremamente completa, é baseado em um HIDS de prestígio e, como mais importante, cumpre todos os requisitos definidos tanto para a implementação específica deste trabalho como para um SIEM em geral. Além disso, Cózar (2020) aprofunda, de forma prática, em praticamente todas as funcionalidades disponíveis pelo Wazuh, descrevendo seus casos de uso e limitações. Baseado nisso, este presente trabalho visa, além de explorar as funcionalidades do Wazuh, utilizá-lo na detecção de vulnerabilidades e formas de ataques mais profundos e diversos, de forma também a analisar sua efetividade.

Quando se fala das possíveis ofensivas que uma rede corporativa pode sofrer, muitos estudos advogam que uma arquitetura SDN bem estruturada pode ser fundamental na defesa contra ataques DDoS. Análises e estudos de segurança em redes SDN são cada vez mais frequentes e relevantes, visto a crescente demanda por implementações SDN que essa geração propõe. O estudo de Wang et al. (2015) exemplifica como a estrutura SDN pode ser um diferencial na defesa aos ataques mesmo em uma rede que usa computação em nuvem. Os autores defendem que a arquitetura SDN torna a lógica de detecção avançada e os processos subsequentes mais fáceis de serem implementados.

Kim et al. (2020) define como se pode usar a arquitetura de uma rede SDN como forma de realizar a verificação de regras de *firewall* à cada enlace da rede, ao contrário de em uma fronteira pré-estabelecida como é feito em arquiteturas legado. A virtualização de funções de redes (NFV) permitem o acoplamento de novos módulos à uma controladora SDN que abrem um leque de oportunidades de implementação. Através das regras de ACL e gerenciamento de VLANs pela controladora SDN Faucet, o presente trabalho implementa, de forma prática, verificações de regras em todos enlaces da rede interna, impedindo diversas formas de exploração de vulnerabilidades.

Capítulo 3

Arquitetura Proposta

Neste capítulo serão descritas as decisões e definições práticas acerca da arquitetura proposta para a topologia deste estudo, de forma a cumprir com os objetivos apresentados.

3.1 *Software* de emulação da rede

Visando implementar uma topologia de rede experimental, que possibilitasse explorar diferentes dispositivos e cenários de configuração necessários para as múltiplas análises às quais esse estudo se propôs, foi necessária a escolha de um *software* capaz de emular a rede e gerenciar os dispositivos virtualizados. No mercado há duas principais soluções que cumprem com esse objetivo: GNS3 e EVE-NG.

Apesar de ambas soluções serem extremamente similares em disponibilidade de recursos, a escolha de utilização foi dada ao GNS3 devido a sua disponibilização *open-source* completamente gratuita e pela sua comunidade mais ativa e robusta. Além disso, há uma grande facilidade na instalação e no compartilhamento remoto da topologia através do sistema de VPN OpenVPN. Dessa forma é possível que múltiplas pessoas possam trabalhar no mesmo projeto em tempo real.

A principal funcionalidade do GNS3, ou qualquer outro *software* de emulação de rede, é a integração com ferramentas de virtualização como VirtualBox, VMWare, Docker e KVM. Essas ferramentas tornam possível a virtualização de diversos sistemas em máquinas virtuais (VMs). Já o GNS3 é responsável por realizar a orquestração dessas VMs, gerenciando os enlaces virtuais para interligação da rede da topologia, controlando o uso de recursos da infraestrutura física, entre outros pontos.

Além da decisão pela utilização do *software* GNS3, utilizou-se *containers* Docker para virtualização dos *switches* Open vSwitch e KVM para os demais sistemas, priorizando desempenho e eficiência de recursos físicos, dado que não é necessário a utilização da VM do GNS3 para virtualização em ambientes Linux.

3.2 Descrição da infraestrutura física disponível e limitações

Há duas principais limitações para o desenvolvimento deste projeto: limites da infraestrutura física disponível pelos autores para hospedagem do laboratório virtual e o acesso restrito à *softwares* e ferramentas.

A infraestrutura física disponível para utilização pelos autores é um *desktop* composto principalmente de um processador Ryzen 3 3100 (4 *cores*, 8 *threads*, totalizando 32 *vCPUs*) e 24GB de memória RAM DDR4 3200MHz. Apesar de não ser uma característica impeditiva, essa infraestrutura própria inviabiliza a construção de maiores e mais complexas topologias, além da implementação de *softwares* com maiores requisitos de processamento.

Já se tratando do acesso restrito à *softwares*, como este estudo propôs realizar a implementação e validação de soluções *open-source* e gratuitas, muitos sistemas mais próximos do estado da arte são desconsiderados. Esta restrição não é, de fato, impeditiva, dado que vem ocorrendo um crescente aumento do desenvolvimento de soluções *open-source*. Além disso, o mercado corporativo também utiliza em larga escala diversas soluções *open-source* que cumprem com seus requisitos.

3.3 Definição da topologia

Como base para o estudo, foi proposta uma topologia inspirada na infraestrutura de um *campus*, ou, mais especificamente, de uma organização que faça o gerenciamento de um *campus*. Como referência principal, utilizou-se a topologia proposta por Brezula (2017), apoiando-se em outras soluções para realizar as devidas adaptações de melhores práticas do mercado identificadas. Inicialmente, foi proposta uma topologia como pode ser visualizado no diagrama da Figura I.1. Após sucessivas evoluções realizadas com o avanço dos estudos e com os testes diversos de implementação, a solução final proposta pode ser visualizada na Figura I.2, e mais detalhadamente nas Figuras 3.1-3.5. Foram definidas 6 áreas principais, identificadas como *ISP1*, *ISP2*, *organization edge*, *DMZ*, *campus* e *datacenter*.

3.3.1 Definição das áreas

As áreas *ISP1* e *ISP2* são definidas como os pontos de conexão da organização à Internet. Dada o arranjo da redundância dos enlaces definido, configura-se uma estrutura *dual multihomed*, estabelecida com a intenção de manter alta disponibilidade de conexão, mesmo após possíveis falhas nos roteadores ou enlaces. Na Figura 3.1, estão definidos os roteadores que simbolizam os ISPs e formam uma ponte com a área de *organization edge*. Ressalta-se a inclusão de um atacante externo na rede WAN do ISP2, sendo este utilizado para análise de tentativas de ataques externos à organização.

A Figura 3.2 representa a área de *organization edge*, ou seja, o *backbone* da organização. Definida por dois roteadores de borda e dois *firewalls*, essa área é responsável por manter a conexão com os dois ISPs e estabelecer a primeira camada de defesa contra ataques. Os roteadores, assim como

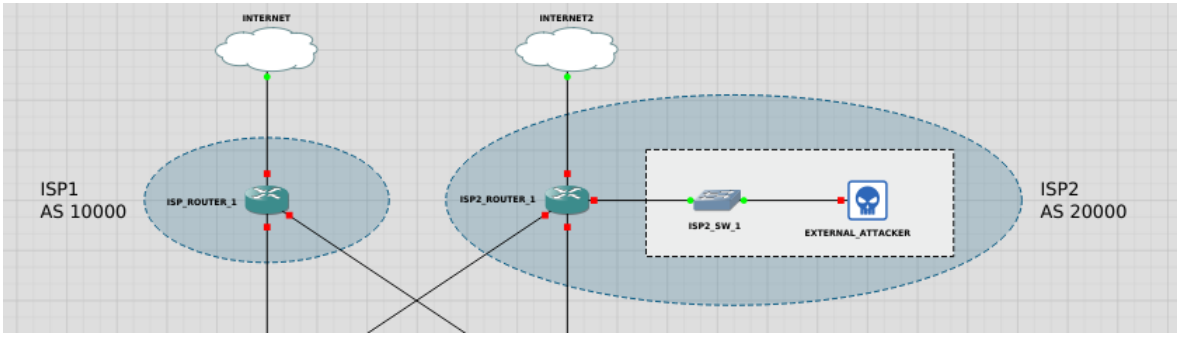


Figura 3.1: Apresentação detalhada da área dos ISPs, ou seja, os provedores de serviço de Internet.

os *firewalls* foram planejados para operarem em redundância, de forma a suportar tanto falhas nos equipamentos quanto nos enlaces. A redundância para os roteadores pode ser facilmente configurada através dos protocolos de roteamento, já os equipamentos de *firewall*, que geralmente possuem estado de vida dinâmico, necessitam que seus sistemas tenham essa capacidade de operação em alta disponibilidade acessível.

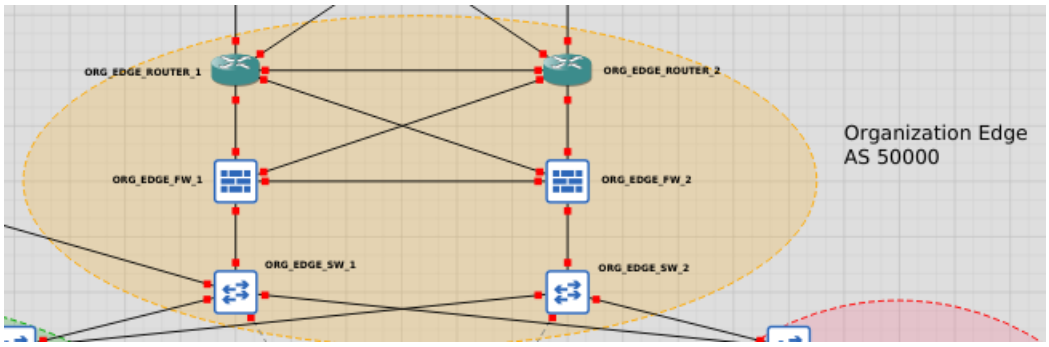


Figura 3.2: Apresentação detalhada da área *organization edge*, representando o *backbone* da organização.

Ao visualizar as áreas internas da organização, depara-se com a implementação de uma rede com arquitetura SDN. Planejada com o intuito de minimizar burocracias de configuração, maximizar o desempenho do tráfego interno, proporcionar facilitada implementação de redundância, implementar regras ACL para controle de acesso e analisar uma tecnologia em ascensão, a implementação dessa arquitetura em detalhes será descrita em próximas seções. Também pode-se observar a presença de uma pequena área de agregação das conexões OOB (*out-of-band*) da arquitetura SDN, mas para simplicidade esta pode ser considerada integrante da área do *datacenter*.

A área de campus, representado na Figura 3.3, apresenta três partes principais: o *campus edge*, o *campus core* e a zona interna de acesso LAN. As zonas de *campus edge* e *campus core* basicamente definem o *backbone* interno dessa área, através de *switches* SDN interconectados entre si e entre as demais áreas adjacentes. A zona interna para acesso da rede LAN por pessoas comuns e funcionários da organização é separada por mais um *firewall*, com a finalidade de restringir ainda mais as possibilidades de ataques, dado que essa zona interna é um ponto de vulnerabilidade da organização à agentes mal intencionados. A disposição da zona interna remete à configuração *three-tier* amplamente utilizada no mercado, que define três níveis de hierarquia até alcançar o

acesso ao usuário final. Além disso, em uma das conexões finais está disposto outro atacante, inserido internamente à organização para teste e análise de suas capacidades de invasão e ataque aos sistemas e equipamentos internos.

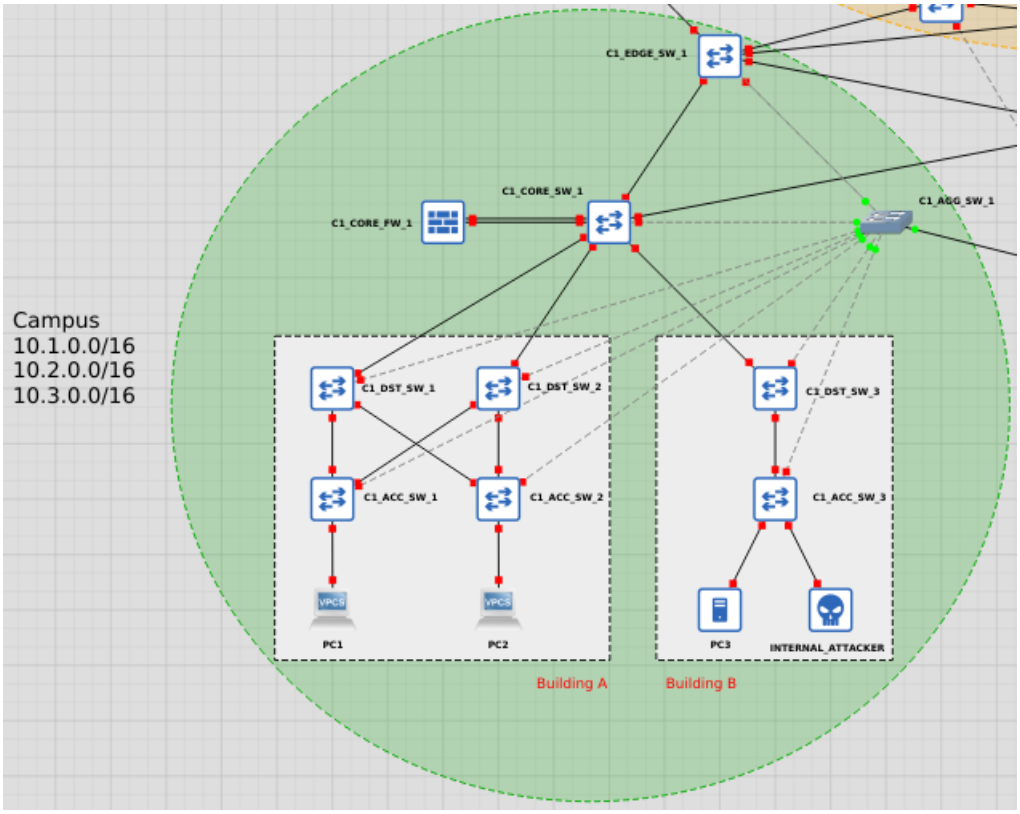


Figura 3.3: Apresentação detalhada da área do campus 01.

A área DMZ, regularmente descrita como zona DMZ ou zona desmilitarizada e representada na Figura 3.4, é projetada para servir aplicações acessíveis internamente e externamente à organização. Dado sua característica de livre acesso às aplicações, essa área também apresenta um grave ponto de vulnerabilidade à organização, facilitando o ataque aos sistemas e, possivelmente, aos servidores que as hospeda. Há diversas técnicas que dificultam esse processo como, por exemplo, o isolamento do tráfego de rede entre a zona DMZ e a rede interna, a implementação de tráfego TLS/SSL na aplicação e a inserção de *reverse proxies* que serão descritos em próximas seções. Neste estudo foi implementando um servidor *web* na área DMZ para análise das vulnerabilidades de segurança passíveis de exploração.

A última área definida para a topologia é o *datacenter*, representado na Figura 3.5. O *datacenter* é focado na alta disponibilidade dos servidores onde estão instaladas as ferramentas de gerenciamento de métricas de desempenho, *logs*, eventos e informações de segurança da organização, incluindo também a controladora da rede SDN. Para visualização dessas ferramentas é definido um *desktop* Debian para gerência, centralizando o local de acesso do administrador da rede ao restante da topologia. Vale ressaltar que foram definidos três servidores no *datacenter* de forma a simular um ambiente mais realístico de dependência da orquestração das ferramentas entre servidores distintos.

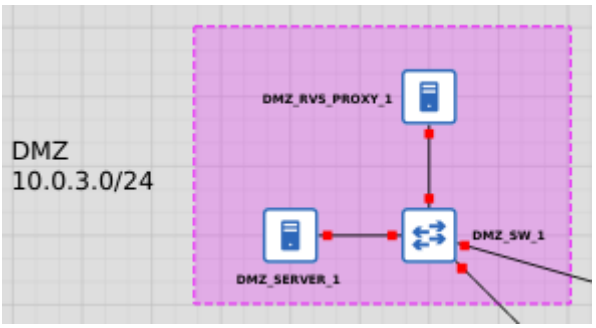


Figura 3.4: Apresentação detalhada da área DMZ, ou zona DMZ.

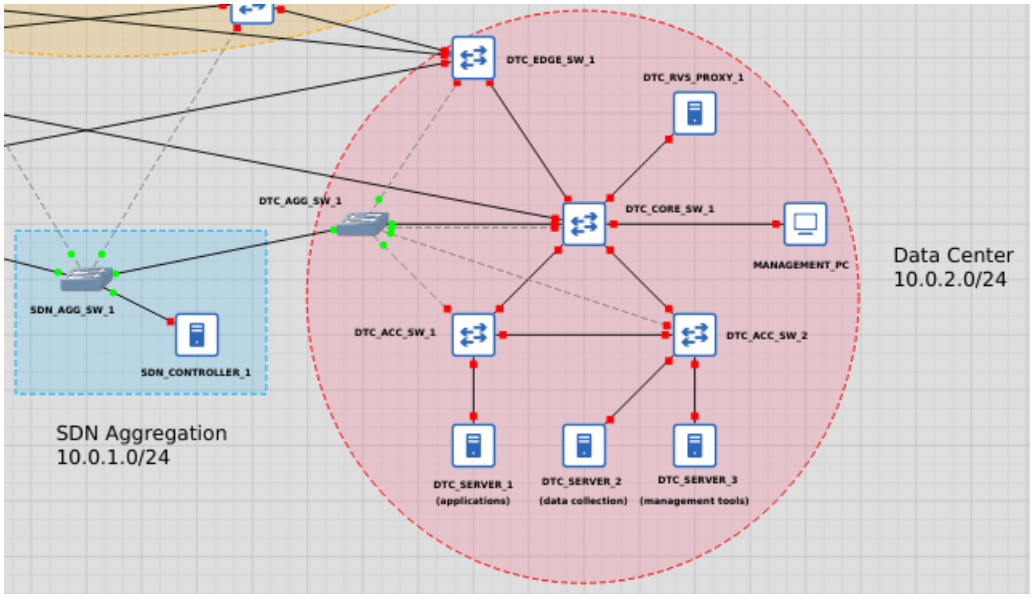


Figura 3.5: Apresentação detalhada da área do datacenter.

3.3.2 Descrição dos dispositivos e sistemas operacionais utilizados

Em ambientes virtualizados, a disponibilidade de máquinas virtuais gratuitas, principalmente para dispositivos de rede, é escassa. Em virtude do crescimento das soluções *open-source*, essa realidade tem gradualmente mudado.

Optou-se pela utilização do sistema VyOS como roteador, definido para os componentes *isp-router-1, isp-router-2, org-edge-router-1* e *org-edge-router-2* da topologia. Este é um sistema baseado em Debian e também é considerado uma das principais soluções do mercado *open-source* e disponibiliza uma plataforma leve e de alta performance na configuração dos principais protocolos de roteamento como OSPF, BGP e outras funções como VPN, NAT, VRRP, entre outros (VYOS, 2021).

Assim como o VyOS, o sistema PfSense é baseado em FreeBSD e é uma das grandes soluções *open-source* do mercado. Apesar de também contar com um sistema completo de soluções internas, nesta proposta este é utilizado principalmente como *firewall*, mas inclui também aplicações DHCP, IDS/IPS com Suricata e roteamento estático (PFSENSE, 2021).

Para os servidores da área DMZ e *datacenter* utilizou-se a distribuição Debian em versão *server* e *desktop*. Esta distribuição é especial pela sua composição completa de *softwares* gratuitos e *open-source*, além de ser consagrada por sua estabilidade e desempenho. Outras distribuições como o Ubuntu, baseado no Debian, também são boas opções, mas nesse caso há, por exemplo, inclusão de *software* proprietário na solução.

Para a pesquisa e execução de ataques à topologia pelas máquinas *external-attacker* e *internal-attacker*, selecionou-se a distribuição Kali Linux, também baseado no Debian. Composta de uma grande variedade das mais diversas ferramentas pré-instalas e configuradas, essa distribuição é extremamente recomendada para ataques de penetração e pesquisa em segurança. Há também outros sistemas similares disponíveis como o ParrotOS, mas a escolha principal é subjetiva dado que estas distribuições são apenas facilitadoras do processo.

Por fim, a seleção do *switch* virtual, além do *switch* disponibilizado nativamente pelo GNS3, foi indiscutivelmente concedida ao Open vSwitch. Utilizado em larga escala como sistema de *bridge* entre VMs, este *switch* é um *software* que disponibiliza diversas funções como gerenciamento de VLANs e, principalmente, suporte ao protocolo OpenFlow (OPEN VSWITCH, 2016). Nesta proposta, o Open vSwitch (OVS) é utilizado para implementar o *backbone* da arquitetura SDN, que será discutida em detalhes em próximas seções.

A Tabela 3.1 evidencia as versões utilizadas para cada um dos sistemas implementados neste projeto.

3.3.3 Decisões de endereçamento IP, VLANs, NAT e boas práticas

O endereçamento IP segue conforme o potencial de escalabilidade da topologia para cada uma das áreas apresentadas. É importante ter em mente uma possível expansão da rede em questão, visto o aumento exponencial de dispositivos conectados nos últimos anos, já discutido nas seções

Aplicação	Versão
Debian Server	11 (Bullseye)
Debian Desktop	11 (Bullseye)
Kali Linux	2021.2
Open vSwitch	2.4.0
PfSense	2.5.2
VyOS	Crux

Tabela 3.1: Descrição das versões dos sistemas utilizados nesta proposta.

anteriores. Além disso, devem ser consideradas as boas práticas de implementação tanto para facilitar organização da rede, prevenir a má configuração do roteamento e também para estabelecer algumas medidas de segurança.

A Tabela 3.2 descreve todo o design e as decisões tomadas acerca do endereçamento IP. Nessa proposta de topologia virtual são implementados poucos dispositivos em relação a uma implementação real, logo são necessárias poucas faixas de endereços IP vagos para comportá-los. Contudo, como já mencionado, as definições foram feitas de acordo com uma possível expansão de cada área e também para o suporte à topologia maiores.

As redes e sub-redes da organização foram definidas em algumas faixas principais, incluindo:

- Sub-redes 181.0.0.0/29 e 191.0.0.0/29: endereços públicos liberados respectivamente pelo ISP1 e ISP2.
- Sub-rede 192.168.0.0/24: faixa de endereços para conexão entre *peers* iBGP.
- Sub-rede 192.168.1.0/24: faixa de endereços para conexão entre *peers* OSPF, incluindo: roteadores e *firewalls* da área *organization edge*.
- Sub-redes 172.16.10.0/24 e 172.16.20.0/24: faixa de endereços para configuração de interfaces *loopback* dos roteadores e *firewalls*.
- Sub-rede 10.0.0.0/14: faixa de endereços que compreende todas as sub-redes internas à organização.
- Sub-rede 10.0.0.0/16: faixa de endereços que compreende as zonas de *backbone, DMZ e datacenter*.
  - Sub-rede 10.0.0.0/24: faixa de endereços do *backbone* (interfaces internas dos *firewalls*).
  - Sub-rede 10.0.1.0/24: faixa de endereços dos *switches* da arquitetura SDN. Rede com tráfego e comunicação *out-of-band*.
  - Sub-rede 10.0.2.0/24: faixa de endereços dos *desktops* e servidores da área do *datacenter*.
  - Sub-rede 10.0.3.0/24: faixa de endereços dos servidores da zona DMZ.
- Sub-redes 10.1.0.0/16 a 10.3.0.0/16: faixa de endereços disponibilizados aos usuários internos do campus.

Dispositivo	Interface	IP	VLAN
ISP-ROUTER-1	eth1	181.0.0.1/29	-
	eth2	181.0.0.9/29	-
	eth0	DHCP	-
ISP2-ROUTER-1	eth0	DHCP	-
	eth1	-	-
	eth2	191.0.0.1/29	-
	eth3	191.0.0.9/29	-
ORG-EDGE-ROUTER-1	loopback	172.16.10.1/32	-
	eth0	181.0.0.2/29	-
	eth1	191.0.0.10/29	-
	eth2	192.168.0.1/30	-
	eth3	192.168.1.1/30	-
ORG-EDGE-ROUTER-2	loopback	172.16.10.2/32	-
	eth0	191.0.0.2/29	-
	eth1	181.0.0.10/29	-
	eth2	192.168.0.2/30	-
	eth3	192.168.1.5/30	-
ORG-EDGE-FW-1	LAN	10.0.0.1/29	10
	WAN.1	192.168.1.2/30	-
	WAN.2	192.168.1.10/30	-
	SYNC1	172.16.0.1/30	-
ORG-EDGE-FW-2	LAN	10.0.0.2/29	10
	WAN.1	192.168.1.6/30	-
	WAN.2	192.168.1.14/30	-
	SYNC2	172.16.0.2/30	-
ORG-EDGE-FW-CARP	CARP	10.0.0.3/29	10
C1-CORE-FW	WAN	10.0.0.4/29	10
	LAN	10.1.0.1/16	TRUNK-101-102
	LAN.101	10.1.1.1/24	101
	LAN.102	10.1.2.1/24	102
DMZ-RVS-PROXY-1	ens4	10.0.3.13/28	500
DMZ-SERVER-1	ens4	10.0.3.1/28	500
DTC-SERVER-1	ens4	10.0.2.129/28	530
DTC-SERVER-2	ens4	10.0.2.17/28	525
DTC-SERVER-3	ens4	10.0.2.18/28	525
DTC-RVS-PROXY-1	eth0	10.0.2.141/28	530
DTC-MANAGEMENT-PC	eth0	10.0.2.1/28	520
SDN-CONTROLLER-1	eth0	10.0.1.254/24	1000
SWITCHES-SDN (OOB)	-	10.0.1.0/24	1000

Tabela 3.2: Endereçamento IP da topologia.

As demais sub-redes internas, definidas em máscaras /28 e /29, foram assim configuradas para evitar lacunas de endereços IP vazios, de forma que também a expansão de cada rede /24 em novas sub-redes /28 ou /29 se torna mais simples e menos propensa a erros. Através da configuração da arquitetura SDN descrita na Seção 3.5, será possível compreender a facilidade da inclusão de novas sub-redes e VLANs relacionadas.

Iniciando pelo extremo da topologia, as áreas dos ISPs e *organization-edge*, serão descritos as boas práticas implementadas na configuração do endereçamento IP da topologia. Em enlaces simples de conexão entre dois *hosts*, como entre *org-edge-router-1* e *org-edge-router-2*, a sub-rede /30 que comporta apenas dois *hosts* é a melhor escolha. Neste caso, não há necessidade de expansão e este enlace deve ser fixo entre ambos, dificultando a possibilidade de má configuração futura e prevenindo que algum atacante interno se aproveite de endereços extras. Já em casos em que pode e deve haver expansão, é aconselhável escolher entre 126 e 510 *hosts* por sub-rede, apesar dessa quantia não ser necessariamente restritiva. Seguindo este princípio, a maioria das sub-redes definidas apresentam máscara de rede /24 que comporta 254 usuários. Algumas sub-redes, definidas entre os ISPs e a área *organization-edge*, são isentas desta prática devido à necessidade de implementação do protocolo NAT para tradução de endereços IP públicos.

Novamente, em casos em que se há uma demanda fixa de endereços IP para *hosts*, a máscara de rede deve ser limitada. No caso da zona DMZ, por exemplo, foi prevista a máscara /28 para poder comportar até 14 servidores. Essa decisão foi tomada de forma que seja possível suportar apenas o *proxy* NGINX e os servidores redundantes das aplicações *web*, FTP, entre outras, operando em modo de balanceamento de carga.

O campus, por sua vez, trata-se de uma área com uma constante demanda por endereços IPs disponíveis, principalmente pela necessidade da inclusão de novas VLANs para os prédios, departamentos, salas ou pontos de acesso para convidados. Com isso em mente, a faixa de IPs definida para o *campus* deve abarcar a possibilidade de uma grande quantidade de futuros *hosts*. Foi inicialmente definido o domínio IP 10.1.0.0/16 para toda a área interna do *campus*, sendo assim suportando aproximadamente 65500 *hosts*. Como já foi descrito, as demais sub-redes 10.2.0.0/16 e 10.3.0.0/16 se encontram disponíveis para inclusão sob demanda.

A implementação das VLANs é essencial para a divisão dos domínios dos tráfegos na rede e também para a definição de regras de acesso tanto pelo *firewall* quanto pela controladora SDN. O ID de uma VLAN não retém muito valor, mas pode ser utilizado para facilitar a organização da rede. A principal aplicação das VLANs, como já mencionado, é segmentar os tráfegos, separando em nível de rede, por exemplo, o tráfego de gerência e o tráfego de usuários convidados. Foram definidas as VLANs: ID 10 para organização interna do campus, incluindo *firewalls*; ID 101 para usuários do prédio A; ID 102 para usuários do prédio B; ID 500 para a zona DMZ; ID 525 para os servidores do *datacenter* que mantém as ferramentas que compõem o núcleo de gerenciamento da topologia; ID 530 para as aplicações do *datacenter* acessíveis pelos dispositivos internos à organização; ID 520 para os *desktops* de gerenciamento.

Acrescentando à lógica de endereçamento IP anterior, a VLAN 101 utiliza a sub-rede 10.1.1.0/24 para suportar seus usuários, e a VLAN 102 utilizada a sub-rede 10.1.2.0/24 para o mesmo objetivo,

Origem	Tipo	Tradução
181.0.0.6/29	DNAT	10.0.3.13
192.168.1.0/24	SNAT	181.0.0.3
10.0.0.0/24	SNAT	181.0.0.3
10.0.2.0/24	SNAT	181.0.0.3
10.1.0.0/16	SNAT	181.0.0.4 - 181.0.0.5
10.2.0.0/16	SNAT	181.0.0.4 - 181.0.0.5
10.3.0.0/16	SNAT	181.0.0.4 - 181.0.0.5

Tabela 3.3: Definição das traduções de endereços IP públicos via protocolo NAT pelo gateway principal.

cada uma suportando 254 usuários. A adição sob demanda de mais VLANs e, consequentemente, mais usuários, é facilitada por boas decisões de pré-separação de faixas de endereços IP livres.

Uma ressalva importante a ser feita é a respeito da rede 10.0.1.0/24, utilizada pelos *switches* da rede SDN. Essa rede é definida como *out-of-band* (OOB), ou fora de banda, que significa que estes endereços não são roteáveis para o restante da topologia, apenas entre *switches* e controladora SDN, abrindo uma exceção apenas para uma conexão dedicada com o servidor do Prometheus que será descrita posteriormente. Esta técnica é geralmente utilizada para definir um canal de alto desempenho e alta disponibilidade de conexão, além de imensamente seguro, dado que não é possível acesso direto à tal sub-rede. Mais informações sobre esta implementação serão descritas em seções específicas.

Como o espaço de endereços IPv4 públicos disponíveis praticamente se esgotou, é importante economizar as sub-redes disponíveis. Como este é um experimento virtual, não houve esforço em se adequar a esta condição, de forma que as configurações de NAT foram definidas de maneira similar entre os roteadores de borda, provendo, por exemplo, 4 endereços distintos para a tradução DNAT especificada na Tabela 3.3. Esta implementação pode ser adaptada utilizando o protocolo IPv6, o qual pode evitar a implementação do protocolo NAT.

Por fim, resta definir as configurações de tradução de endereços IP públicos via protocolo NAT, ou mais especificamente, SNAT e DNAT. Essas configurações são realizadas de forma similar nos roteadores *org-edge-router-1* e *org-edge-router-2* que atuam em modo de balanceamento de carga. A implementação DNAT é utilizada para encaminhar o tráfego externo recebido pela organização para o *proxy* reverso da zona DMZ que encaminhará o tráfego aos servidores das aplicações respectivos à cada fluxo. Já as implementações SNAT são responsáveis por garantir o acesso à Internet para os endereços IP internos que não podem ser roteados para fora da organização. Estas definições estão resumidas na Tabela 3.3 para o *gateway* principal da rede, o endereço 181.0.0.1.

3.3.4 Protocolos de roteamento implementados e configurações de alta disponibilidade em nível de rede

O protocolo BGP, *Border Gateway Protocol*, é o padrão utilizado para configurar o roteamento inter e intra sistemas autônomos (ASs), garantindo conexão da Internet pública com as redes

privadas. É através deste protocolo que foi estabelecida a comunicação entre a organização e os dois ISPs definidos na topologia. A configuração deste protocolo implica na definição de *peers*, isto é, os dispositivos que estabelecem uma conexão TCP entre si, e é a localização desses dispositivos nos ASs que descrevem a configuração em eBGP ou iBGP. O iBGP define a conexão entre roteadores em uma mesma AS, que devem manter uma conexão *full-mesh* entre si para evitar *loops*. De maneira oposta, quando a conexão é estabelecida entre roteadores de diferentes ASs, implementa-se uma configuração eBGP entre os chamados roteadores de borda (ASBR). Através dessa conexão é que se garante a comunicação entre diferentes sistemas autônomos e, geralmente, domínios de diferentes organizações (CISCO, 2008b).

Nesta proposta são os roteadores de borda da área *organization edge* que estabelecem as conexões eBGP com os roteadores de borda das áreas ISP1 e ISP2. Para as áreas ISP1 e ISP2 foram definidos os ASNs, números únicos para ASs públicas, 10000 e 20000, respectivamente. Para a AS da organização foi definido o ASN 50000, sendo então possível a comunicação entre os três ASs. Através dos roteadores VyOS foram estabelecidos os *peers* e os vínculos entre si, garantindo a comunicação da organização com a rede pública provida pelos ISPs, como pode ser visualizado na Figura 3.6. O sistema VyOS permite a configuração de pesos, métricas e filtros das rotas manipuladas, como também configurações de segurança, como autenticação das sessões entre *peers*, mas estas características não compõem o escopo desta proposta. Em uma implementação real, a configuração minuciosa e otimizada do protocolo BGP é essencial para prevenir até mesmo grandes impactos em escala global da Internet, como já ocorreram na história. Além disso, é importante ressaltar que este protocolo também é suscetível a vulnerabilidades como qualquer outro, mas podem resultar em consequências ainda mais drásticas.

```
vyos@org-edge-router-1:~$ show ip bgp neighbors | grep AS
BGP neighbor is 172.16.10.2, remote AS 50000, local AS 50000, internal link
    4 Byte AS: advertised and received
BGP neighbor is 181.0.0.1, remote AS 10000, local AS 50000, external link
    4 Byte AS: advertised and received
BGP neighbor is 191.0.0.9, remote AS 20000, local AS 50000, external link
    4 Byte AS: advertised and received
```

Figura 3.6: Apresentação da tabela de vizinhos BGP do roteador org-edge-router-1.

Garantir redundâncias ao longo da arquitetura, principalmente em regiões mais críticas e com maior vazão de tráfego, como o *backbone*, é vital para garantir a operação em alta disponibilidade da rede projetada. Isso define um grande desafio para os administradores da rede, que é viabilizar o método mais eficiente de roteamento e garantir redundância de equipamentos e sistemas para manter a disponibilidade de recursos mesmo em caso de uma possível eventualidade. Nesta proposta, foi implementado um design *dual multi-homed* para a área dos roteadores de borda, ou seja, foram definidos dois enlaces diferentes à cada um dos ISPs através dos roteadores de borda da organização. Desse modo, foram assegurados quatro enlaces dedicados aos ISPs, garantindo redundância de enlaces e sistemas em caso de falhas e possibilitando a configuração de balanceamento de carga ou *failover* para conexão da organização com o meio externo. Assim como pode ser visualizado na Figura 3.7, há três rotas padrão (0.0.0.0/0), originadas dos *peers* BGP, definidas no roteador *org-edge-router-1*. De acordo com a documentação do VyOS (2021), a primeira seleção é

feita com base no peso de cada rota, que foram manualmente definidas de acordo com a prioridade desejada, sendo a conexão *org-edge-router-1<->isp-router-1* a principal e as demais definidas como backup em caso de falha.

```
vyos@org-edge-router-1# run show ip bgp
BGP table version is 8, local router ID is 172.16.10.1, vrf id 0
Default local pref 100, local AS 50000
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*  0.0.0.0/0        191.0.0.9              100           100 20000 ?
*>                  181.0.0.1               10          1000 10000 ?
* i                  191.0.0.1               10    100     10 20000 ?

Displayed  1 routes and 3 total paths
```

Figura 3.7: Apresentação da tabela de rotas BGP do roteador *-edge-router-1.*

Para possibilitar o roteamento entre os roteadores de borda da organização e os *firewalls* dedicados, *org-edge-fw-1* e *org-edge-fw-2*, implementou-se o protocolo OSPF. Esta mesma configuração poderia ter sido realizada através da definição de rotas estáticas, dado que naturalmente esta parte da infraestrutura é fixa. Contudo, para promover um ambiente de alta disponibilidade, é necessário estar preparado para eventuais ocorrências de falhas nos sistemas e assegurar a existência de solução de *backup* automatizada com as mesmas funcionalidades em segundo plano. O sistema de *firewall* PfSense foi configurado com a intenção de operar em alta disponibilidade, mas este tema será tratado na próxima seção. Entretanto, é importante atestar que a configuração de roteamento dinâmico com OSPF foi efetuada de forma que não houvesse um impeditivo para que esta resposta executasse em tempo real (CISCO, 2013).

Através da configuração dos roteadores de borda da organização e dos *firewalls* via protocolo OSPF, é mantido um vínculo de vizinho que permite, dinamicamente, definir as rotas de melhor caminho nesta área. Inicialmente, o roteador *org-edge-router-1* é o principal ponto de saída da rede, logo este redistribui a rota padrão (0.0.0.0/0), de origem dos *peers* eBGP, para os outros vizinhos. Como estão definidos enlaces redundantes entre os roteadores e os sistemas de *firewall*, poderiam ser estabelecidos diversos caminhos através da definição de métricas nas rotas, entretanto apenas um *firewall* é mantido ativo por vez. O dispositivo *org-edge-fw-1* é o principal *firewall* e o único ativo até que ocorra uma falha em seu sistema ou em uma sua conexão LAN que o torne indisponível, migrando assim a conexão das áreas internas da organização para o *firewall org-edge-fw-2*. Este revezamento exige que o roteamento seja dinâmico e que novas rotas sejam estabelecidas com as falhas ocorridas. Através das configurações do protocolo OSPF nos sistemas VyOS e FRR (disponível no PfSense), esse requisito é cumprido, dado que está disponível uma configuração no PfSense que permite desabilitar o pacote FRR quando o protocolo CARP (protocolo de alta disponibilidade) está em modo de "BACKUP". Essa situação pode ser visualizada nas Figuras 3.8 e 3.9, que descrevem um trecho da tabela de rotas do roteador *org-edge-router-1* que apresenta uma rota para a rede do *org-edge-fw-2* (192.168.1.14/30), mas não o apresenta como vizinho conhecido

pois o mesmo está desabilitado devido ao modo "BACKUP"do protocolo CARP que será descrito em próximas seções.

```
vyos@org-edge-router-1:~$ show ip route | grep 192.168
O>* 10.0.0.0/14 [110/20] via 192.168.1.2, eth3, 00:13:16
O   172.16.10.2/32 [110/20] via 192.168.0.2, eth2, 00:44:16
S>* 172.16.10.2/32 [1/0] via 192.168.0.2, eth2, 00:45:04
O>* 181.0.0.8/29 [110/20] via 192.168.0.2, eth2, 00:44:16
O>* 191.0.0.0/29 [110/20] via 192.168.0.2, eth2, 00:44:16
O   192.168.0.0/30 [110/1] is directly connected, eth2, 00:44:24
C>* 192.168.0.0/30 is directly connected, eth2, 00:45:04
O   192.168.1.0/30 [110/1] is directly connected, eth3, 02:31:01
C>* 192.168.1.0/30 is directly connected, eth3, 02:31:03
O>* 192.168.1.4/30 [110/2] via 192.168.0.2, eth2, 00:10:45
O>* 192.168.1.8/30 [110/2] via 192.168.0.2, eth2, 00:13:28
O   192.168.1.12/30 [110/1] is directly connected, eth4, 00:10:45
C>* 192.168.1.12/30 is directly connected, eth4, 02:31:04
```

Figura 3.8: Apresentação de um trecho tabela de rotas OSPF do roteador *org-edge-router-1.*

```
vyos@org-edge-router-1:~$ show ip ospf neighbor

Neighbor ID     Pri State           Dead Time Address         Interface                        RXmtL RqstL DBsmL
172.16.20.1       1 Full/Backup       31.072s 192.168.1.2     eth3:192.168.1.1                     0     0     0
172.16.10.2       1 Full/DR           34.816s 192.168.0.2     eth2:192.168.0.1                     0     0     0
```

Figura 3.9: Apresentação da tabela de vizinhos OSPF do roteador *org-edge-router-1*, sem a inclusão do *firewall org-edge-fw-2*.

É relevante detalhar a configuração das interfaces de *loopback* dos roteadores para bom funcionamento destes protocolos em redundância, assim como pôde ser visualizado na Figura 3.9. Os *loopbacks* são interfaces de rede que estão sempre disponíveis enquanto os dispositivos estão ativos, de forma que esta característica garante que o roteador esteja visível mesmo que um dos enlaces diretamente conectados seja desativado. Configurar os *peers* dos protocolos iBGP e OSPF através destas interfaces é recomendável para assegurar conectividade entre todos os roteadores internos à organização, mesmo que não diretamente. Esta prática já não é recomendada para todos os casos de implementação eBGP, pois pode significar um falso positivo em um caso que não tenha outra rota disponível para esse *peer* externo. Entretanto, esta configuração é relevante para uma topologia com enlaces redundantes (CISCO, 2008b).

3.4 Solução de Firewall

No processo de robustecer o sistema de segurança, é vital a implementação de uma forte solução de *firewall*. De acordo com Scarfone e Hoffman (2009), restringir conectividade tanto de/para redes internas é uma política muito adotada em redes corporativas com o propósito de evitar vulnerabilidades e permite controlar acessos não autorizados a sistemas e recursos.

O sistema PfSense (2021), construído sobre FreeBSD, é uma plataforma focada no gerenciamento de *firewall* que permite a instalação de pacotes para implementação de uma enorme diversidade de serviços e aplicações adicionais. Dessa forma, é possível configurar protocolos de roteamento, IDS/IPS, servidor syslog, servidor VPN, exportador de métricas e outras diversas

soluções. A escolha por essa solução foi feita para a implementação do *firewall* e outras medidas de segurança ao longo de toda a topologia.

Ao configurar o PfSense, é fundamental compreender como suas interfaces são designadas. Por padrão, esse sistema conta com uma interface LAN e uma interface WAN. A interface WAN é aquela visível a conexões externas e redes públicas e, no caso dos dois *firewalls* na área *organization edge*, forma os enlaces que se comunicam com os roteadores de borda. Já a interface LAN estabelece uma conexão com uma rede local. Essa é a interface do *firewall* geralmente configurada com um IP privado. É importante ressaltar que, sendo necessário definir mais *links* entre *hosts* internos e o *firewall*, é possível configurar diferentes interfaces virtuais através das interfaces físicas.

Como mencionados anteriormente, foram instalados e configurados ao todo três *firewalls* na topologia aqui proposta: os dois *firewalls* da área de *organization edge*, *org-edge-fw-1* e o *org-edge-fw-2*, e o *firewall* da camada *core* do campus, o *c1-core-fw*. Para cada um desses *firewalls*, foram implementados diferentes serviços, de acordo com as necessidades definidas em cada área, todos utilizando o sistema PfSense. Entre os *firewalls* do *backbone*, vale ressaltar as configurações de alta disponibilidade e, no *firewall* do campus, a definição das interfaces virtuais. Essas implementações serão especificadas com mais detalhes nas seções a seguir.

3.4.1 Configuração de alta disponibilidade

Como mencionado nas seções anteriores, garantir a disponibilidade dos serviços e sistemas é um dos grandes desafios dos administradores da rede. O *backbone* da rede é a parcela da infraestrutura que interconecta as diferentes sub-redes definidas entre si e também estabelece a conexão de cada uma delas com a rede externa. Isso significa um elevado fluxo de informações, o que demanda altas taxas de transmissão, alta performance de processamento e uma forte política de segurança. Um dos recursos utilizados pelos autores foi a implementação do protocolo CARP como uma solução para garantir a alta disponibilidade dos *firewalls* do *backbone* (NETGATE, 2021a).

É crucial que toda informação e serviço que adentre a rede interna pela interface WAN e, consequentemente, tenha acesso aos servidores da zona DMZ ou até mesmo outros sistemas, seja inspecionada pelo *firewall*. Isso implica em uma forte demanda desse serviço que, em um cenário de eventualidade, se torna um impeditivo para o tráfego de informações, no caso de se encontrar indisponível. Por esse motivo, é vital definir uma estrutura de redundância ao serviço, garantindo assim, um sistema de alta disponibilidade.

O protocolo CARP foi introduzido pela OpenBSD e se apresenta como uma alternativa ao protocolo VRRP que é proprietário (NETGATE, 2021a). O CARP foi implementado de maneira a proporcionar essa configuração de alta disponibilidade, através das ferramentas do próprio PfSense, como uma solução para o sistema de redundância entre os *firewalls* da área de *organization edge*. De acordo com Netgate (2021a), para configurar a alta disponibilidade, é necessário estabelecer uma terceira interface, além das interfaces LAN e WAN, usada apenas para garantir o sincronismo das configurações e *status* dos *firewalls* em questão. No caso dessa interface SYNC, basta definir as configurações desejadas no sistema primário que as mesmas serão configuradas pela sincroniza-

ção com a interface SYNC do outro sistema, neste caso secundário. As opções de sincronização utilizadas nesta proposta podem ser visualizadas na Figura 3.10.

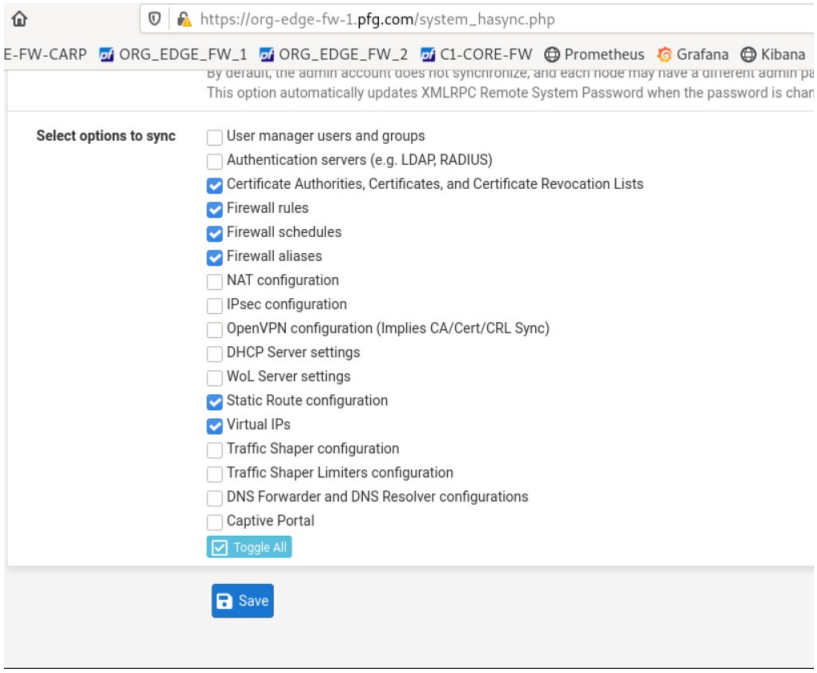


Figura 3.10: Configuração de sincronismo no *firewall org-edge-fw-1*, definido como sistema primário.

Para completar a configuração de alta disponibilidade também é necessário criar o endereço IP virtual compartilhado entre os sistemas primário e secundário. Através da Figura 3.11, fica ilustrada a definição da interface virtual na interface LAN, semelhante em cada um dos *firewalls*, configuradas com o IP CARP compartilhado 10.0.0.3/29. Também pode ser visualizada a relação entre ambos os sistemas, sendo que o *firewall org-edge-fw-1* é considerado o sistema primário "MASTER", o *firewall org-edge-fw-2* somente pode ser considerado como o sistema secundário "BACKUP".

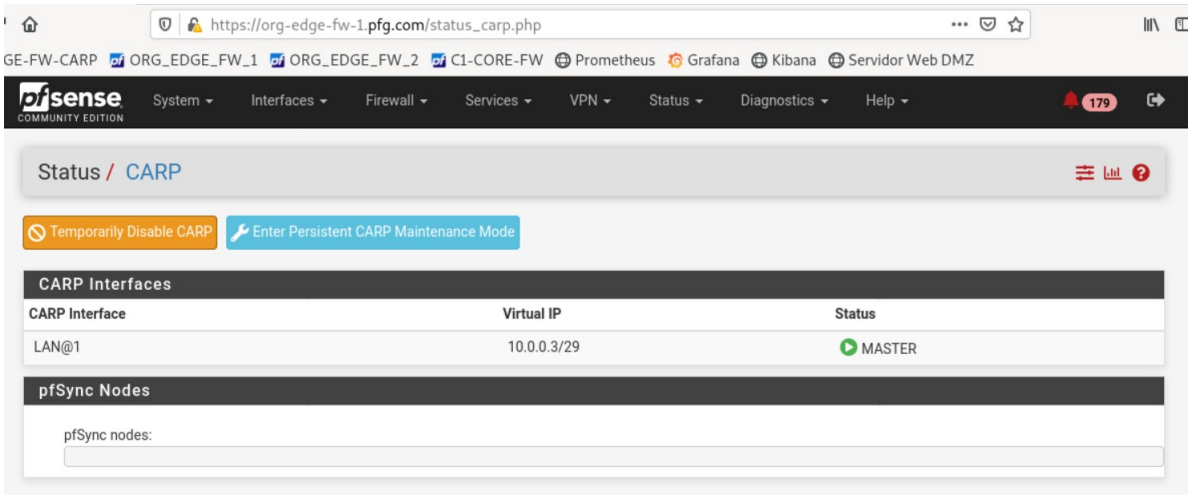


Figura 3.11: Verificação do status do protocolo CARP no *firewall org-edge-fw-1*.

Com a completa configuração de sincronismo e alta disponibilidade, toda regra criada ou mudança realizada no *firewall* primário são automaticamente replicadas no *firewall* secundário. Isso garante, por exemplo, que se o *firewall* primário se encontrar indisponível, o *firewall* secundário possa assumir seu lugar com as mesmas regras e configurações estabelecidas no sistema primário. Essa característica expande o grau de disponibilidade do serviço, evitando grandes períodos de interrupções no tráfego e nas regras de segurança, além de manter completa transparência da substituição dos sistemas para os usuários e dispositivos da topologia.

3.4.2 Definição de regras

Através das regras definidas nas interfaces do *firewall* é que se limita determinados fluxos, pacotes e conexões, de acordo com as especificações desejadas. Netgate (2021b) explica que tráfegos iniciados do lado da rede LAN são filtrados por regras na interface LAN e o mesmo acontece para a WAN e qualquer outra interface definida. Isso justifica as diferentes regras definidas em cada uma das interfaces, de acordo com as demandas do *firewall* em questão, sendo para isso especificados protocolos, endereços de origem, destino e portas utilizadas.

O *firewall org-edge-fw-1* tem suas regras de passe e bloqueio das interfaces LAN e WAN ilustradas nas Figuras 3.12 e 3.13. As regras são aplicadas na ordem em que estão organizadas, portanto, é sempre importante definir as regras de maneira lógica e sequencial. É vital definir a regra de anti-bloqueio, o que evita que o administrador da rede seja bloqueado e não tenha acesso ao próprio *firewall* e suas configurações. São também definidas as regras que permitem o repasse do tráfego DNS, HTTP e HTTPS, através da especificação de suas portas, visto os serviços fornecidos na topologia, que devem ser acessíveis para todos os seus usuários. O tráfego ICMP foi definido de maneira a permitir que *hosts* internos consigam estabelecer um fluxo entre si e acessar dispositivos externos a rede, porém sem gerar *replies* para requisições iniciadas externamente. Qualquer outro tráfego de pacotes para além dos descritos são bloqueados pela última regra. Como já descrito anteriormente, as regras definidas no *firewall org-edge-fw-2* são sincronizadas às definidas no *firewall* primário.

Já as regras do *firewall c1-core-fw* definem uma segunda camada de proteção mais próxima aos usuários da rede. As regras de *Floating*, que permite configurações abrangentes entre múltiplas interfaces, estão expostas na Figura 3.14 e especificam a liberação de requisições DNS, HTTPS e ICMP, bloqueando qualquer outro tipo de requisição e resposta. Na Figura 3.15 estão as regras da interface WAN, bloqueando sub-redes não alocadas e qualquer outro tráfego que não utilize o protocolo HTTPS, que não seja proveniente da aplicação do Prometheus ou que não corresponda a uma requisição ou resposta ICMP. Vale especificar também que todo o tráfego DNS permitido pelos *firewalls* corresponde apenas com a conexão com o servidor DNS disponibilizado no *datacenter*, sendo ele o responsável pelo encaminhamento para servidores externos caso seja necessário.

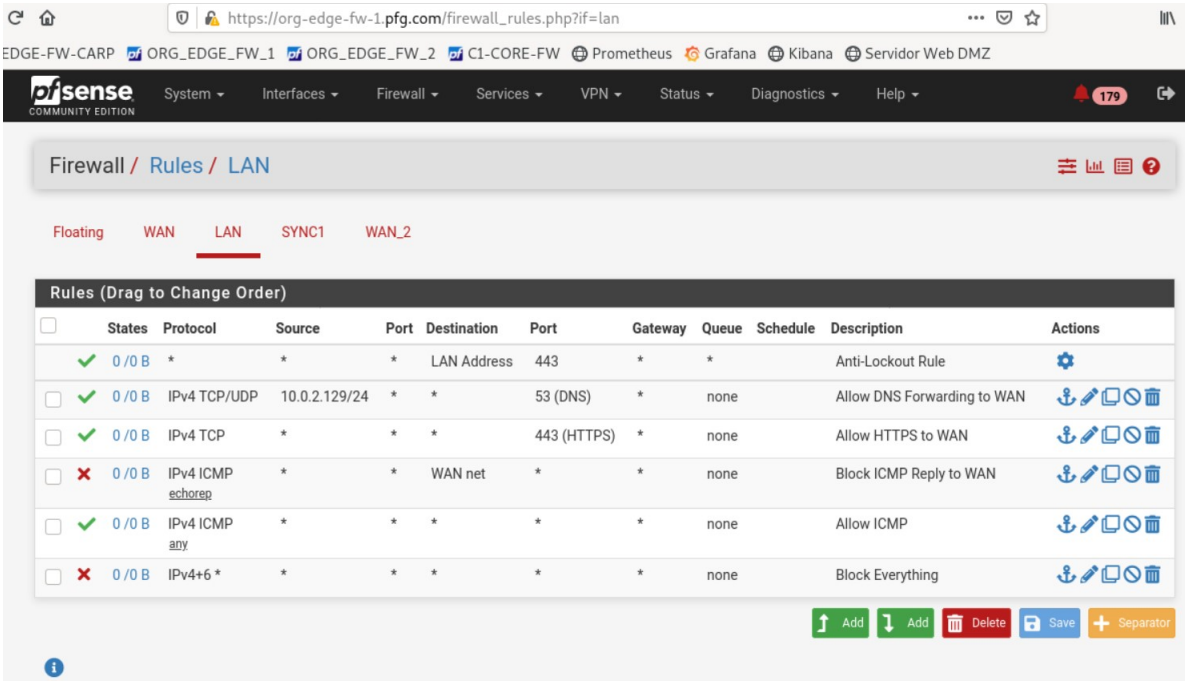


Figura 3.12: Regras definidas para a rede LAN do *firewall org-edge-fw-1*.

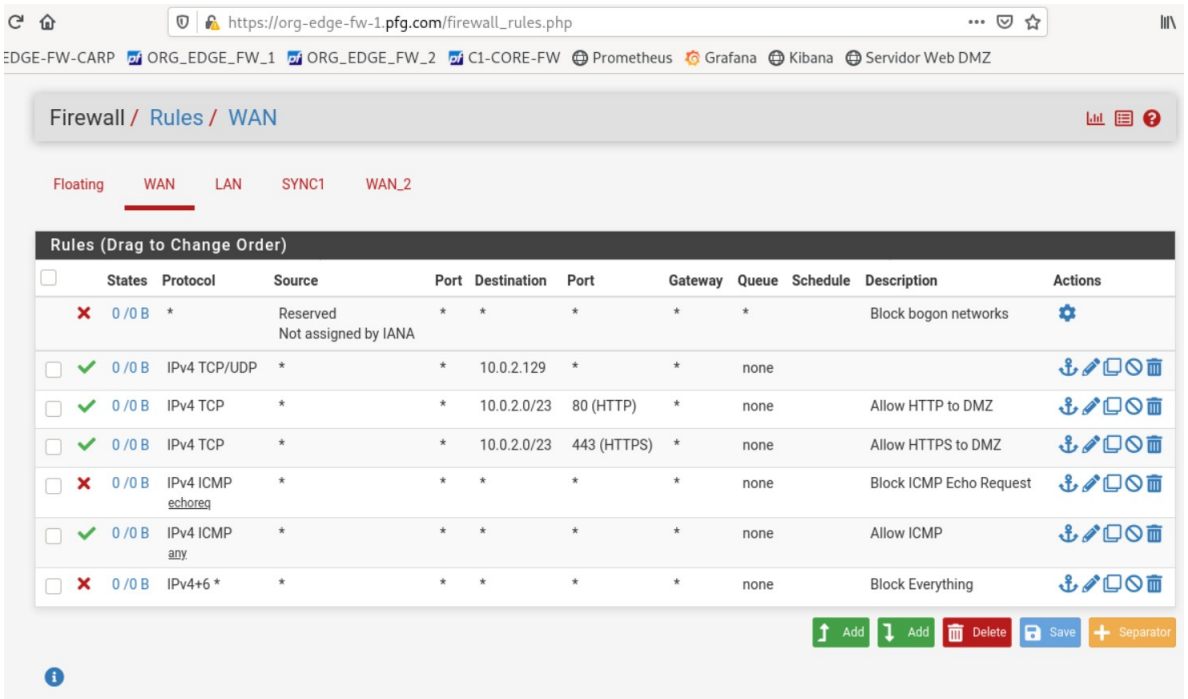


Figura 3.13: Regras definidas para a rede WAN do *firewall org-edge-fw-1*.

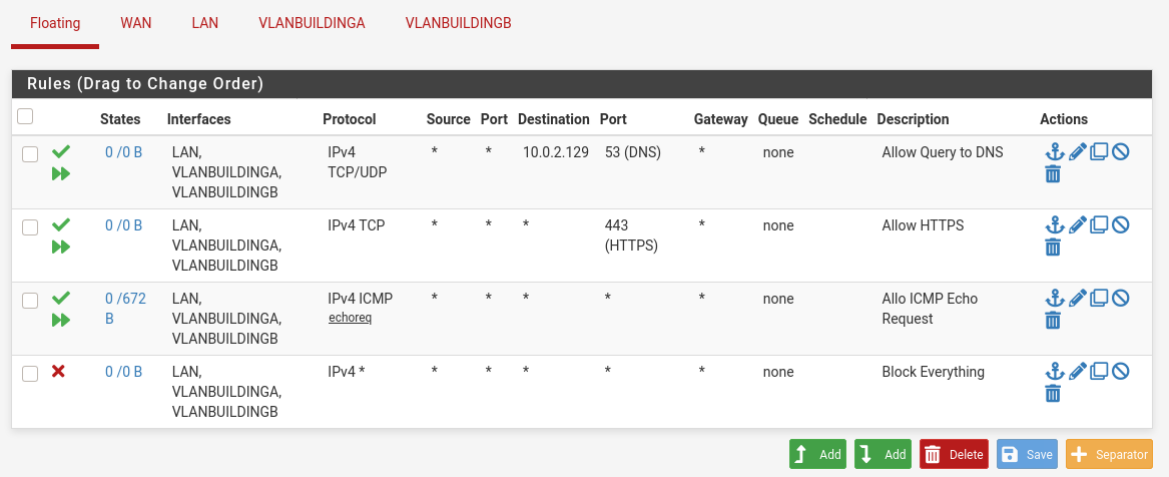


Figura 3.14: *Floating rules* definidas para múltiplas interfaces do *firewall c1-core-fw*.

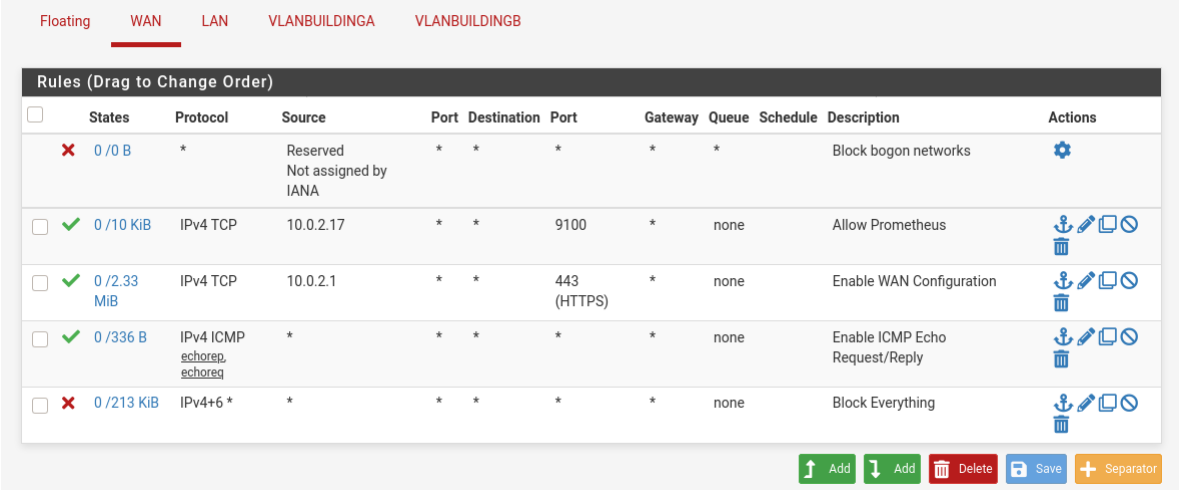


Figura 3.15: Regras definidas para a rede WAN do *firewall c1-core-fw*.

3.4.3 Implementação de NIDS com Suricata

A detecção e prevenção de intrusões, principalmente relacionada ao tráfego de rede, é uma técnica de muito valor, porém realizar sua configuração é uma tarefa complexa e delicada. Quando o tráfego de usuários pela Internet era concentrado principalmente no protocolo HTTP, a verificação de pacotes e fluxos era muito mais direta, sendo possível checar por padrões suspeitos e anômalos. Com o avanço do poder de processamento, técnicas de criptografia e leis de privacidade, esta possibilidade de detecção se torna muito difícil ou até impossível. Por exemplo, a desencriptação de tráfego HTTPS é realizado por muitas empresas para garantir que seus funcionários não exportem dados sensíveis para destinos indevidos, mas isto deve ser feito de acordo com contrato claro previamente estabelecido. Esta prática já não pode ser empregada para o tráfego de usuários externos que atingem os servidores da zona DMZ, por exemplo, o que dificulta extensivamente a detecção de tentativas de ataques.

Esta técnica de detecção de intrusões pela análise do tráfego de rede se chama NIDS, e nesta proposta foi implementada em todos os *firewalls* através da ferramenta chamada Suricata. O Suricata é uma aplicação similar ao Snort, ferramenta já consagrada como NIDS, mas que apresenta configurações mais completas e processamento de dados mais eficiente. Esta ferramenta permite a análise do tráfego de rede em diversas interfaces, adição de regras de validação, geração de alertas e também o funcionamento como uma ferramenta NIPS, que funciona como prevenção contra ataques e intrusões (OISF, c2019).

Como todo NIDS, a adição e o ajustes das regras de validação para contemplar os requisitos necessários da organização podem levar até meses. Além disso, a análise realizada pode sempre retornar resultados falso positivos ou falso negativos, de forma que o monitoramento automático desses alertas não é uma tarefa fácil. Neste caso, a implementação de um NIPS se torna ainda mais complexa, dado que o bloqueio automático de fluxos através dos alertas gerados pode se tornar um impeditivo na experiência dos usuários inocentes.

De acordo com essas descrições, optou-se por utilizar o sistema do Suricata apenas como IDS, proporcionando a análise de alertas sem possíveis obstruções de tráfego. A configuração inicial foi realizada através das regras providas pela organização Proofpoint e sua coleção chamada *Emerging Threats*, e também pelas regras públicas da comunidade do Snort. Por padrão muitas regras dessas listas não são ativadas por resultarem em um excesso de alarmes falsos, logo, além das ativações padrões, foram ativadas apenas as regras que fariam sentido para as análises finais. Alguns exemplos de alertas classificados como alarmes falsos podem ser visualizados na Figura 3.16.

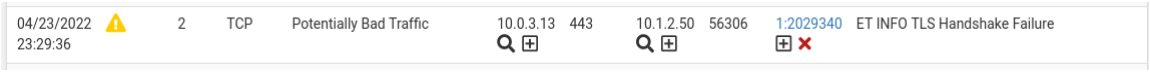


Figura 3.16: Alerta apresentado pelo Suricata com alta probabilidade de ser definido como falso positivo.

Além da definição de regras respectivas a cada escopo implementado, também é necessário determinar que interfaces deverão ser analisadas. A validação das regras definidas é um processo que

demanda muitos recursos, pois este é realizado para cada pacote recebido em uma interface, logo este é o principal obstáculo para a implementação desta ferramenta em todos os locais possíveis. No caso do *firewall c1-core-fw*, por exemplo, o NIDS foi implementado apenas na interface LAN, de forma que apenas o tráfego dos usuários internos fosse analisado, verificando anomalias e fluxos proibidos que possam ser originados principalmente da VLAN de visitantes.

3.5 Arquitetura SDN

Como já mencionado, SDN é uma arquitetura de redes que expande as capacidades de roteamento legado ainda majoritariamente implantado na rede de computadores mundial. A ascensão desta arquitetura é fundamentada pela dinamicidade proporcionada por esta implementação, o que pode ser explicado pela velocidade de transformação para um cenário cada vez mais virtual e automatizado. Organizações como Cisco, Juniper, IBM, entre outras, atualmente já disponibilizam diversas soluções baseado em SDN para *datacenters*, como SD-WAN.

Dessa forma, o ambiente *cloud* se torna o principal foco dessa tecnologia no cenário atual. O Google, um dos principais fornecedores de solução em *cloud*, foi um dos pioneiros nessa implementação, disponibilizando o uso em produção no ano de 2011, apenas três anos depois do lançamento do artigo sobre o OpenFlow. Essa solução proprietária chamada Orion evoluiu bastante com o tempo, mas ainda utiliza do protocolo OpenFlow para comunicação entre o plano de configuração e de dados (*switches*) (FERGUSON et al., 2021).

3.5.1 Controladora SDN

Apesar de existirem muitas soluções proprietárias e pagas para a implementação de redes SDN, o foco desta proposta se mantém em ferramentas *open-source* que cumpram com os requisitos necessários. Após testes em controladoras SDN populares como ONOS e OpenDaylight, foi definido a utilização da controladora Faucet, que até mesmo é apresentada em tutoriais do sistema Open vSwitch. Essa escolha foi realizada principalmente a partir da facilidade de configuração proporcionada e pela documentação detalhada com exemplos.

O Faucet é um sistema completo que disponibiliza todos os requisitos necessários para este estudo, incluindo: configuração de VLANs; configuração de ACLs; roteamento estático, BGP e inter-VLANs; monitoramento via Prometheus e Grafana através do Gauge; entre outros (FAUCET, 2021c). Nesta proposta esta controladora foi incluída no servidor *sdn-controller-1*, agregado à área do *datacenter*.

Uma das principais vantagens e dificuldades de uma arquitetura SDN é a centralização do controle da rede nas controladoras SDN. Em implementações mais complexas deve sempre haver um estudo de melhor caso para definição do nível de redundância das controladoras, de forma que falhas na comunicação entre controladoras e *switches* pode cessar todo o funcionamento da rede. Também não é possível estabelecer um grande *cluster* de controladoras a fim de manter a alta disponibilidade, pois isso causará desperdício de recursos e dificuldades de gerenciamento

(FERGUSON et al., 2021). Nesta proposta, utilizou-se apenas uma controladora Faucet para gerenciamento da rede, de forma que os *switches* Open vSwitch realizam o objetivo de prover redundância à topologia. É importante ressaltar que essa comunicação é feita *out-of-band*, ou seja, a rede é dedicada apenas para comunicação entre controladora e *switches*, promovendo confiabilidade da transmissão e evitando brechas de segurança.

Através do Faucet, a configuração de toda a rede SDN pode ser feita em um arquivo *faucet.yaml* no servidor *sdn-controller-1*. Neste arquivo está descrita a definição das VLANs, configuração das VLANs em cada porta, regras de ACL, roteadores inter-VLAN, rotas estáticas e configurações de redundância. Além disso, o Faucet atua como um serviço do sistema, sendo então possível manter facilmente seu controle de operação, como o reinício após modificação de sua configuração. Essa característica também pode ser manipulada pela definição de *scripts*, de forma que é viável alterar as definições da rede apenas modificando esse arquivo de configuração e reiniciando o serviço.

De acordo com a arquitetura descrita por Faucet (2021a), toda a configuração dos *switches* é realizada em 7 tabelas OpenFlow que seguem uma sequência pré-determinada, sendo que o pacote seguirá para o próximo enlace apenas quando for autorizado por todas essas tabelas. A atualização dessas tabelas pode ser observada por cada *switch* Open vSwitch ou até mesmo pelos *logs* da controladora que descrevem grande parte do fluxo de tomada de decisão da rede. Um exemplo desse fluxo pode ser visualizado na Figura 3.17.

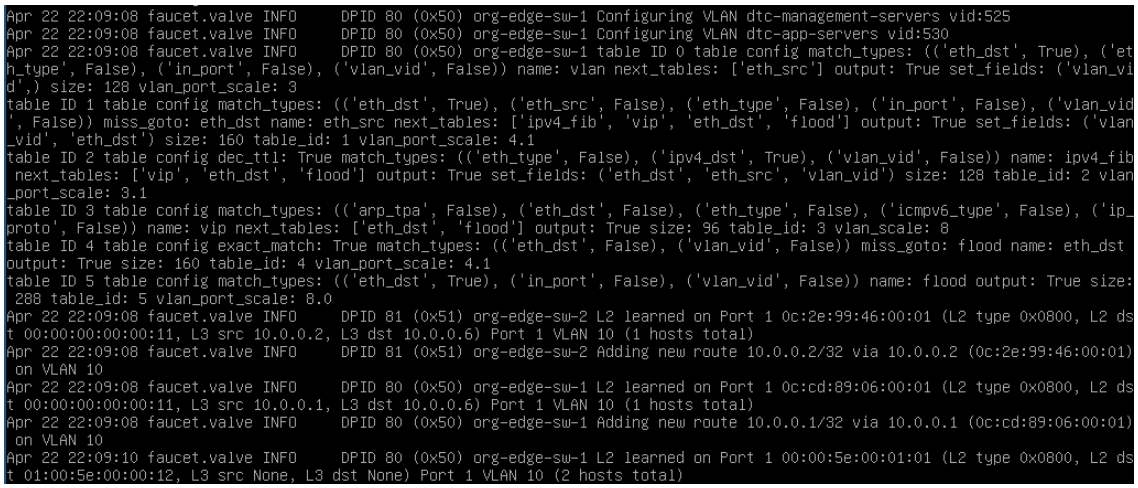


Figura 3.17: Trecho do arquivo de log da controladora Faucet que descreve algumas definições realizadas nos *switches* (*datapaths*).

3.5.2 Configuração de redundância

A controladora Faucet disponibiliza um simples método de configuração de redundância que foi explorado para viabilização do *backbone* do campus como um ambiente de alta disponibilidade. Esta característica é tratada como uma árvore, ou seja, cada *switch* possui uma prioridade na hierarquia. Quanto menor o valor de prioridade, maior é o nível da hierarquia, logo o valor 1 é a raiz da árvore, o nó principal. Através desse nó, o *switch org-edge-sw-1*, derivam diversos *switches* que continuam a hierarquia até alcançar o nó de maior nível.

A definição dos enlaces é dada de forma que o *switch* de prioridade n é conectado ao *switch* de prioridade n+1, formando uma conexão redundante se houver mais de um *switch* de prioridade n. Através dessa configuração, evitam-se possíveis problemas relacionado a falhas no enlace ou até mesmo colapso de um *switch*. Um exemplo dessa configuração pode ser visualizado na Figura 3.18. É importante observar a simplicidade na definição de dispositivos redundantes.

```
dps:
    org-edge-sw-1:
        dp_id: 0x50
        hardware: "Open vSwitch"
        stack:
           priority: 1
        interfaces:
            1:
                description: "iface to org-edge-fw-1"
                native_vlan: backbone
            2:
                description: "stack link to c1-edge-sw-1"
                stack:
                   dp: c1-edge-sw-1
                   port: 1
            4:
                description: "stack link to dtc-edge-sw-1"
                stack:
                   dp: dtc-edge-sw-1
                   port: 4
            5:
                description: "stack link to dmz-sw-1"
                stack:
                   dp: dmz-sw-1
                   port: 4
```

Figura 3.18: Exemplo de configuração de *switches* redundantes, modo *stack*, pela controladora Faucet.

É sempre importante realizar um estudo sobre *trade-off* na configuração de redundância, de forma a balancear o objetivo de alta disponibilidade com o alto consumo de recursos e a complexidade da configuração. Na topologia proposta o *backbone* interno da organização é o foco desta implementação, de forma que alguma falha nesta área não prejudique todo o restante da organização. Em alguns casos, como na conexão direta com os dispositivos finais, a falta de redundância é inevitável, dado que muitos dispositivos nem ao mesmo suportam duas conexões simultâneas.

Mais especificamente na rede do *campus*, a redundância de enlaces é essencial dado que os *switches* estão geralmente conectados entre grandes distâncias. Em comparação, nos *datacenters* normalmente os servidores e *switches* estão inseridos dentro de um mesmo *rack*, facilitando o ajuste em casos de falhas.

3.5.3 Definição das VLANs, roteamento inter-VLAN e roteamento estático

Assim como nas práticas de roteamento legado, a definição de VLANs é essencial para divisão lógica do tráfego de rede. Apesar de ser realizado de forma relativamente diferente a partir da controladora Faucet, essa configuração permite separar o tráfego de diferentes sub-redes em um mesmo *switch* e realizar o roteamento entre as mesmas apenas quando definido explicitamente.

Um exemplo prático dessa configuração pode ser observado na Figura 3.19, sob o parâmetro *vlans*. Inicialmente para cada VLAN é definido um nome para facilitar a configuração posteriormente, um ID único no intervalo 1-4094, um endereço MAC arbitrário para servir de base para criação da interface e também um endereço IP virtual que será o *gateway* da VLAN. Esses passos

são normalmente abstraídos na configuração de um *switch layer* 2 comum, sendo necessário definir apenas a porta em que cada VLAN está definida. Contudo, o sistema Open vSwitch utiliza do kernel Linux e das ferramentas disponíveis, como *bridges e namespaces*, por isso essas configurações são necessárias.

```
vlans:
    backbone:
        vid: 10
        description: "VLAN for organization internal backbone"
        faucet_vips: ["10.0.0.6/29"]
        faucet_mac: "00:00:00:00:00:11"
        routes:
            - route:
                ip_dst: "10.0.0.3/32"
                ip_gw: "10.0.0.3"
            - route:
                ip_dst: "0.0.0.0/0"
                ip_gw: "10.0.0.3"
            - route:
                ip_dst: "10.1.0.0/16"
                ip_gw: "10.0.0.4"
```

Figura 3.19: Exemplo da configuração de VLANs pela controladora Faucet.

A definição das VLANs respectivas a cada enlace é dada de duas formas: acesso nativo ou entrocamento. Um exemplo dessa configuração pode ser visualizada na Figura 3.18 e é similar ao funcionamento de um *switch* genérico. O acesso nativo opera basicamente designando a VLAN definida para cada porta utilizada e o método de entrocamento é similar, porém é definida a agregação de duas ou mais VLANs conectadas no mesmo enlace, visto que o tráfego de múltiplas VLANs devem trafegar pelo mesmo canal. O método de entroncamento é utilizado, por exemplo, no enlace que conecta o *firewall c1-core-fw* com o *switch c1-core-sw-1* e, dado que o tráfego de múltiplas VLANs são concentradas na mesma interface LAN deste *firewall*, é necessário que cada VLAN será gerenciada por uma interface virtual específica.

Dado que o gerenciamento de toda a arquitetura projetada pela controladora Faucet é baseada em VLANs, todos os dispositivos conectados à rede devem também estar designados a uma VLAN específica. Portanto, em alguns casos, é essencial que VLANs distintas possam trafegar dados entre si, e isso é realizado através do parâmetro *routers*. Definindo um nome e indicando quais VLANs podem se comunicar, essa abstração de roteadores implementa o conceito de roteamento inter-VLAN, similar às práticas legado. Um exemplo desse caso de uso pode ser visualizado na Figura 3.20.

```
routers:
    router-backbone-pcs:
        vlans: [backbone, dtc-management-pcs]
    router-backbone-dmz:
        vlans: [backbone, dmz-servers]
    router-backbone-appservers:
        vlans: [backbone, dtc-app-servers]
    router-backbone-mgmtservers:
        vlans: [backbone, dtc-management-servers]
```

Figura 3.20: Exemplo da configuração de roteadores inter-VLAN pela controladora Faucet.

Verifica-se também a presença de rotas estáticas incluídas pelo parâmetro *routes* dentro da configuração de uma VLAN específica. A rota padrão (0.0.0.0/0), por exemplo, é essencial para conduzir o tráfego para fora da organização, ou seja, para a Internet. Como pode ser visualizado, o destino definido é o IP em modo CARP dos *firewalls org-edge-fw-1 e org-edge-fw-2*, de maneira a proporcionar conexão mesmo que um dos sistemas esteja indisponível. Como a sub-rede do IP

CARP não é detectada automaticamente pelo anúncio ARP dos *firewalls*, também é definido uma rota estática para o endereço 10.0.0.3/32.

Também é importante ressaltar a presença de uma rota para a sub-rede 10.1.0.0/16 referente aos usuários internos do *campus*. Essa característica foi definida uma vez que é necessário fazer com que o fluxo para o *firewall c1-core-fw* seja iniciado pela interface WAN que, por sua vez, deve seguir internamente pela interface LAN, alcançando as interfaces virtuais de cada VLAN respectiva. Esta disposição de *firewall* isolado foi estabelecida para não causar a separação da rede SDN em ilhas isoladas, evitando a necessidade da inclusão de outros *switches* para agregação dos enlaces na rede LAN e WAN do *firewall*.

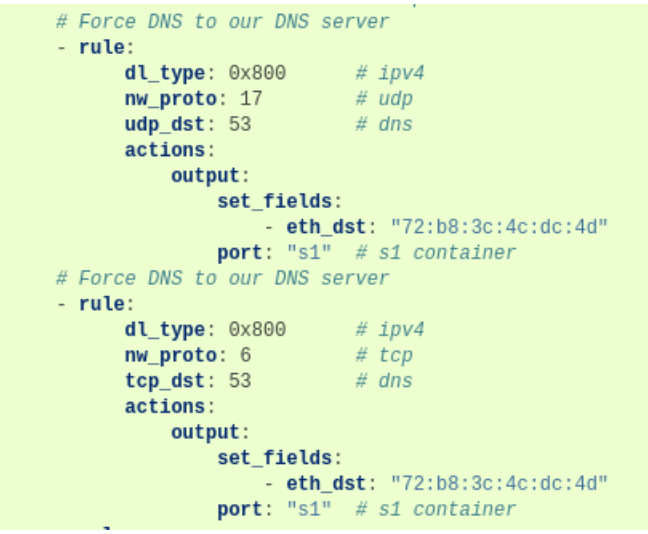
3.5.4 Definição das listas de controle de acesso (ACLs)

Muitas vezes as regras de ACL são comparadas e confundidas com regras de *firewall*. Listas de controle de acesso são regras analisadas para cada pacote que fluem pela entrada de uma interface, seja esta de um *switch*, roteador ou *firewall*. Essas listas realizam uma comparação simples e direta entre o pacote e cada uma das regras definidas, de forma a aceitar ou bloquear o fluxo daquele pacote. Por sua vez, regras de *firewall* são responsáveis por analisar todo o fluxo do pacote recebido por uma interface, realizando múltiplas checagens antes de determinar um destino apropriado. O *firewall* também mantém controle do estado do fluxo, o que agiliza análises posteriores para o mesmo fluxo, sendo assim designado como *stateful*, enquanto listas de controle de acesso são designadas como *stateless*.

Através das possibilidades proporcionadas pelo protocolo OpenFlow, a controladora Faucet disponibiliza a configuração de ACLs com algumas vantagens. A princípio, listas de controle de acesso realizam uma análise individual para cada pacote trafegado, aprovando ou não sua entrada, porém a controladora implementada permite: aprovação, bloqueio, redirecionamento para outras portas, espelhamento para outras portas e alteração de campos. Através destas características é possível definir comportamentos ainda mais complexos que implementações comuns de ACLs, como espelhar os pacotes recebidos para análise em uma ferramenta *out-of-band* ou até mesmo adicionar uma regra para bloquear um atacante através de *scripts* dinâmicos. Um exemplo comum das regras de ACL implementadas é a forçação de requisições DNS serem encaminhadas para o servidor DNS da própria organização ao invés de um servidor qualquer que pode ser hospedado por um atacante, sendo este exemplo representado na Figura 3.21.

3.6 Aplicações implementadas

A presente seção descreve as aplicações implementadas e disponibilizadas na rede para consumo dos usuários internos e/ou externos, proporcionando também análises de diversos cenários de ataques, exploração de vulnerabilidades e consequente observação de suas respostas por parte dos administradores da rede.



```
# Force DNS to our DNS server
- rule:
      dl_type: 0x800      # ipv4
      nw_proto: 17        # udp
      udp_dst: 53         # dns
      actions:
          output:
              set_fields:
                  - eth_dst: "72:b8:3c:4c:dc:4d"
              port: "s1"  # s1 container
# Force DNS to our DNS server
- rule:
      dl_type: 0x800      # ipv4
      nw_proto: 6         # tcp
      tcp_dst: 53         # dns
      actions:
          output:
              set_fields:
                  - eth_dst: "72:b8:3c:4c:dc:4d"
              port: "s1"  # s1 container
```

Figura 3.21: Exemplo de configuração de uma regra de ACL para forçar o redirecionamento de requisições DNS para um servidor específico (FAUCET, 2021b).

3.6.1 Servidor DNS

O serviço de DNS é extremamente importante tanto no contexto privado quanto da Internet global. Responsável por traduzir nomes textuais de domínios em endereços IP, erros na configuração ou propagação destes domínios podem causar indisponibilidade de acesso a milhões de usuários simultaneamente, como já aconteceu múltiplas vezes na história da Internet. Há também diversas formas de ataques que se aproveitam da funcionalidade da aplicação DNS para espalhar domínios falsos que redirecionam a endereços maliciosos, ataques de negação de serviço através da inundação de tráfego DNS redirecionado à vítima, entre outros (ISC, c2022).

Nesta proposta foi implementada a aplicação DNS Bind9, sendo este um dos primeiros e mais utilizados sistemas DNS em ambientes Linux (ISC, c2022). Este software é uma solução *open-source* e dispõe de todas as funcionalidades necessárias de uma aplicação DNS. Nesta proposta, o serviço de DNS interno foi implementado e configurado no servidor *dtc-server-1* do *datacenter* e pode ser consultado por toda a extensão das áreas do *campus, datacenter* e da zona DMZ. O mapeamento de domínios e *hostnames* é feito através das tabelas de *forward* e de *reverse lookup*. Inicialmente foram estabelecidas as zonas de domínio e, de acordo com as necessidades deste estudo, optou-se por criar um domínio genérico "pfg.com"e um domínio específico para a zona DMZ "dmz.com". Quando o usuário faz uma consulta direta, isto é, quando ele realiza uma requisição usando um *hostname*, o servidor DNS utiliza a tabela de *forward* do domínio especificado para traduzir o *hostname* para o endereço IP definido. O mesmo é válido para as tabelas de consulta reversa, dado que o usuário possui o endereço IP e busca consultar o *hostname* definido para o mesmo. Estas consultas por endereços não mapeados internamente são recursivamente encaminhados ao servidor público do Google, 8.8.8.8.

A Figura 3.22 evidencia o arquivo de configuração em que se estabelecem as zonas de domínio definidas arbitrariamente no servidor, tanto a "pfg.com"e "dmz.com", quanto suas respectivas zonas reversas. Nota-se que também são referenciados os arquivos de tradução para cada uma

```
zone "pfg.com" IN {
        type master;
        file "/etc/bind/forward.pfg.com";
};

zone "0.10.in-addr.arpa" IN {
        type master;
        file "/etc/bind/reverse.pfg.com";
};

zone "dmz.com" IN {
        type master;
        file "/etc/bind/forward.dmz.com";
};

zone "3.0.10.in-addr.arpa" IN {
        type master;
        file "/etc/bind/reverse.dmz.com";
};
```

Figura 3.22: Detalhes do arquivo de configuração das zonas definidas no servidor DNS da organização.

dessas zonas e, nas Figuras3.23 e 3.24 estão ilustradas essas tabelas de *forward* e de *reverse lookup* para a zona "pfg.com", respectivamente. São incluídos os endereços e respectivos *hostnames* de maneira paralela nos dois arquivos, para que a tradução ocorra em ambas direções.

```
$TTL    604800
@       IN      SOA     dtc-server1.pfg.com root.dtc-server1.pfg.com. (
                        20220401        ; Serial
                          604800        ; Refresh
                           86400        ; Retry
                         2419200        ; Expire
                          604800 )      ; Negative Cache TTL
;
@       IN      NS      dtc-server1.pfg.com.

dtc-server1     IN      A       10.0.2.129
dtc-server2     IN      A       10.0.2.17
dtc-server3     IN      A       10.0.2.18
dtc-proxy       IN      A       10.0.2.141
mgmt-pc         IN      A       10.0.2.1
org-edge-fw-1   IN      A       10.0.0.1
org-edge-fw-2   IN      A       10.0.0.2
org-edge-fw     IN      A       10.0.0.3
c1-core-fw      IN      A       10.0.0.4
```

Figura 3.23: Detalhes do arquivo de configuração da tabela *forward* DNS do domínio pfg.com.

```
$TTL    604800
@       IN      SOA     dtc-server1.pfg.com. root.dtc-server1.pfg.com. (
                        20220401        ; Serial
                          604800        ; Refresh
                           86400        ; Retry
                         2419200        ; Expire
                          604800 )      ; Negative Cache TTL
;
@       IN      NS      dtc-server1.pfg.com.

129.2   IN      PTR     dtc-server1.pfg.com.
17.2    IN      PTR     dtc-server2.pfg.com.
18.2    IN      PTR     dtc-server3.pfg.com.
141.2   IN      PTR     dtc-proxy.pfg.com.
1.2     IN      PTR     mgmt-pc.pfg.com.
1.0     IN      PTR     org-edge-fw-1.pfg.com
2.0     IN      PTR     org-edge-fw-2.pfg.com
3.0     IN      PTR     org-edge-fw.pfg.com
4.0     IN      PTR     c1-core-fw.pfg.com
```

Figura 3.24: Detalhes do arquivo de configuração da tabela *reverse* DNS do domínio pfg.com.

De acordo com Debian (2022), há diversas formas de garantir a configuração de um servidor DNS com Bind9 de forma segura. Entre estas, cita-se a definição das faixas de endereços IP aceitas para busca e recursão, a configuração do parâmetro de "versão"do serviço exposto e também o isolamento do serviço do Bind9 no servidor hospedado através de um usuário próprio e através um ambiente lógico fixo via comando *chroot*. Embora estas técnicas não tenham sido empregadas nesta proposta para análise, é importante definir a importância de manter cada aplicação segura mesmo esta sendo utilizada apenas no ambiente interno da organização.

3.6.2 Proxies

Apesar das diferentes técnicas implementadas até aqui, o processo de conexão e consumo dos serviços dos servidores da topologia continuam diretamente visíveis, gerando suscetibilidade para um possível atacante. Por esse motivo, foi implementada uma solução de *proxy* reverso para acesso aos servidores distribuídos pelas áreas do *datacenter* e DMZ. A escolha foi pela aplicação *open-source* Nginx.

A solução oferecida por NGINX (c2022a) atua interceptando todas as requisições endereçadas aos servidores, isto é, quando configurado como um *proxy* reverso. Isso quer dizer que, na realidade, os clientes fazem suas requisições ao servidor *proxy* e este as encaminha aos serviços apropriados, através da URL utilizada. As configurações do *Nginx* permitem que o administrador configure *endpoints* que redirecionam a requisição para o endereço e porta indicados no parâmetro *proxy-pass*. No caso da topologia aqui descrita, foram configurados dois *proxies* reversos, *dtc-rvs-proxy-1* para os servidores no *datacenter*, e *dmz-rvs-proxy-1* para o servidor *web* na zona DMZ. A Figura 3.25 traz, como exemplo, a configuração principal do Nginx como *proxy* reverso. Nesta lógica está principalmente o encaminhamento das conexões realizadas em *proxy.dmz.com/webserver* para o servidor *web*, além das demais boas práticas de segurança. Essa mesma lógica também foi empregada para o *proxy* reverso do datacenter, de modo a encaminhar as conexões para o Grafana, Prometheus e Kibana, redirecionando-as para os servidores hospedeiros nas portas expostas para cada serviço. Na maioria dos casos, as ferramentas que dispõem de interfaces *web* já disponibilizam trechos de configuração para definição de um *proxy* reverso como interceptador.

A Figura 3.25 também define a configuração da conexão HTTPS com os usuários através de um certificado auto-assinado gerenciado pelo sistema de autoridade certificadora (CA) definido no PfSense de borda da organização, de forma a facilitar o controle de chaves nas áreas internas da organização. Este mesmo processo deveria ser realizado através de uma CA confiável caso a implementação fosse realizada em um domínio da Internet pública. Prosseguindo, através dessa configuração, mantém-se uma conexão segura entre o *proxy* reverso e os usuários das aplicações servidas. Contudo, também é necessário a implementação de outras medidas de segurança para dificultar a exploração de vulnerabilidades, como cabeçalhos para forçar a utilização do protocolo HTTPS por navegadores e evitar técnicas de XSS, configuração de parâmetros da conexão SSL como versão aceita, tempo para *timeout* da sessão, entre outros (NGINX, c2022b). Para esta proposta, as boas práticas de configuração para o servidor Nginx foram necessárias devido à quantidade de análises realizadas sobre este.

```
server {
        listen 443 ssl;
        ssl_certificate /etc/ssl/certs/pfsense-cert.crt;
        ssl_certificate_key /etc/ssl/private/pfsense-cert.key;

        ssl_protocols TLSv1.3;
        ssl_prefer_server_ciphers on;
        ssl_dhparam /etc/nginx/dhparam.pem;
        ssl_ciphers EECDH+AESGCM:EDH+AESGCM;
        ssl_ecdh_curve secp384r1;
        ssl_session_timeout  10m;
        ssl_session_cache shared:SSL:10m;
        ssl_session_tickets off;
        ssl_stapling on;
        ssl_stapling_verify on;

        resolver 10.0.2.129 valid=300s;
        resolver_timeout 5s;

        add_header Strict-Transport-Security "max-age=63072000; includeSubDomains; preload" always;
        add_header X-Frame-Options DENY;
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";

        root /var/www/html;

        index index.html index.htm index.nginx-debian.html;

        server_name proxy.dmz.com;

        location /webserver {
                proxy_pass http://10.0.3.1:3567/;
        }
}
```

Figura 3.25: Detalhes do arquivo de configuração do *proxy* reverso da zona DMZ.

O uso do *proxy* reverso torna a infraestrutura interna desconhecida para qualquer usuário que tentar iniciar uma conexão ou enviar requisições aos serviços disponibilizados, dado que seus endereços são abstraídos. Não tendo conhecimento da topologia interna da zona *DMZ*, por exemplo, um atacante externo terá dificuldade de penetrar e inviabilizar as aplicações servidas, além de todas suas requisições serem inicialmente filtradas pelo *proxy*. O uso desta ferramenta também possibilita a configuração de balanceamento de carga, controle de acesso a cada endereço, cache de dados estáticos, além de facilitar o processo de manutenções internas, que podem ser realizadas de forma transparente para os usuários.

3.6.3 Servidor web

A aplicação mais comum a ser implementada em zonas DMZ, de forma a estarem disponíveis publicamente, são as aplicações *web*. Elas permitem fácil acesso a informações e aos sistemas interativos de uma organização para os usuários finais e, por isso, agregam muito valor. Nesta proposta, a implementação de um servidor *web* se deu para validação dos possíveis ataques que este pode sofrer, como por exemplo, ataques de negação de serviço.

Utilizou-se do *framework* NodeJs para a implementação de uma simples aplicação *web*, operacional no servidor *dmz-server-1* e incluído na zona DMZ. Esta aplicação desenvolvida retorna um simples texto, apenas para avaliação do seu funcionamento. A principal proposta é o monitoramento do fluxo e evidências dos ataques realizados sobre esta aplicação, logo, foram implementadas apenas configurações básicas de segurança, sendo que a linha principal da análise é o *proxy* re-

verso já descrito, que interceptará toda a comunicação do servidor *web* com os usuários internos ou externos. A Figura 3.26 mostra o servidor *web* em execução pelo acesso através do navegador Mozilla no *desktop* de gerência. É importante ressaltar o endereço observado no navegador, que indica conexão com o *proxy* reverso e consequente encaminhamento da conexão para o servidor *web*. Reitera-se também a presença de um aviso na conexão, indicando que o emissor do certificado não é naturalmente conhecida pelo Mozilla, o que em casos normais indicaria que algum atacante inseriu indevidamente o certificado raiz no navegador do usuário. Nesta proposta, a inserção foi realizada manualmente para verificação da conexão.

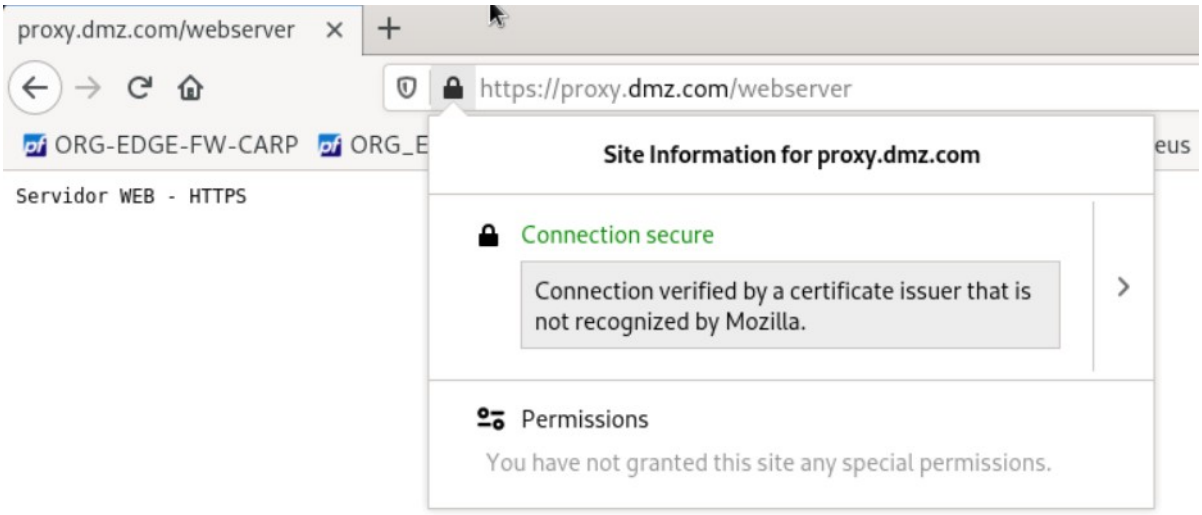


Figura 3.26: Resposta da requisição do servidor *web* hospedado na zona DMZ. Verifica-se o estado da conexão HTTPS, gerada por certificados auto-assinados.

Dado que vulnerabilidades na camada de aplicação são diversas e complexas, principalmente quando incluídos bancos de dados que armazenam informações pessoais e sensíveis, as análises de ataques mais especialistas foram adiadas para trabalhos futuros, de modo que não contempla o escopo da proposta atual. O foco se estabelece então na análise de ataques no nível de rede. Um último ponto apropriado é o fato do *proxy* Nginx também possibilitar a configuração da conexão HTTPS com o servidor *web*, por exemplo, dessa forma mantendo dois canais completamente independentes e seguros, até mesmo contra agentes internos da organização com acesso ao tráfego .

3.6.4 Servidor DHCP

Nesta proposta, foram definidas duas VLANs para usuários da área interna do campus, VLAN 101 para o prédio A e a VLAN 102 para o prédio B, sendo esta última utilizada para que usuários visitantes se conectem à rede da organização. Apesar de ambas VLANs não possuírem comunicação entre si, é necessário que estas se conectem à interface LAN do *firewall* para estabelecerem conexão com o restante da rede. Para isso foram definidos servidores DHCP para cada uma das interfaces

virtuais derivadas da interface primária LAN.

Como já mencionado, cada VLAN foi definida em um endereço com máscara de rede 255.255.255.0, ou /24, derivadas do endereço 10.1.0.0/16 que foi inicialmente designado à área interna do campus. Quando um usuário se conecta fisicamente à rede, o mesmo deve fazer uma busca por servidores DHCP através de mensagens de requisição *broadcast* e, ao encontrar um servidor, estabelecer a sessão DHCP através da troca de mensagens *unicast*. Com o estabelecimento da sessão, é concedido ao usuário um endereço IP pertencente à sub-rede, de forma que esse obtém permissão de se conectar na rede e na VLAN em que se encontra inserido (ROSS; LIANG; SHARKEY, 2021).

Em sua maioria, as mensagens de *DHCPDISCOVER*, transmitidas em *broadcast* na rede, podem ser utilizadas para ataques de DHCP *spoofing*, por exemplo. Nesta técnica, um servidor DHCP falso pode responder à requisição realizada pelo cliente e conceder um endereço IP que o tenha como *gateway*, ou seja, a definição de um destino de encaminhamento direto ao atacante. Formas de ataque como essa, não observadas pelo *firewall*, são críticas vulnerabilidades e, para reduzir essas possibilidades, a maioria dos *switches* implementam técnicas de prevenção ao DHCP *spoofing* através da definição de uma porta confiável no *switch*. No caso desta proposta, utilizando as possibilidades da controladora Faucet, restrições como essa podem ser realizadas através da definição de regras de ACL para requisições DHCP e mensagens *broadcast*.

3.7 Solução de monitoramento de métricas de desempenho utilizando Prometheus e Grafana

Dada a extensa dimensão da arquitetura desenvolvida até aqui, inspirada em topologias corporativas, é inevitável ir de encontro com as mesmas dificuldades que os administradores dessas infraestruturas enfrentam: o monitoramento de métricas de desempenho e gerenciamento de eventos de segurança ao longo de toda a rede. O gerenciamento de um alto número de ativos de rede e diferentes fluxos se torna um desafio. Tendo em vista esse cenário, foram implementadas duas soluções em conjunto para a coleta, formatação, análise e visualização centralizada de métricas e estatísticas.

De acordo com a própria documentação do Prometheus (PROMETHEUS, c2022b), este trata-se de uma ferramenta de coleta e armazenamento de métricas como dados em séries de tempo, isso é, são métricas referenciadas pela data e hora da coleta, além dos rótulos definidos pela própria aplicação. O Prometheus utiliza desses rótulos para melhor dimensionar cada tipo de métrica, visto que, dependendo da origem, o usuário pode desejar ter acesso a diferentes informações. Através dos rótulos e nomes associados, é possível instanciar corretamente cada métrica em diferentes categorias, o que permite uma visualização mais apropriada dessas métricas.

O Grafana foi outra solução escolhida por ser uma aplicação *open-source* focada no monitoramento e visualização de métricas em tempo real que permite a criação de *dashboards* customizáveis (SHIVANG, 2022). Uma das principais integrações do Grafana é o Prometheus, que atua como sua base de dados. A estruturação de dados por período de tempo permite traçar perfis de comporta-

mento de aplicações, usuários e ativos, frequência de determinados eventos e até mesmo identificar erros ou incidentes. Através da análise de métricas de desempenho, o administrador da rede pode identificar gargalos, problemas de processamento e até mesmo ataques de *flooding*, podendo assim, propor e atuar em melhorias e atualizações na topologia e em seus ativos. As duas aplicações foram disponibilizadas nos servidores do *datacenter, dtc-server-2 e dtc-server-3*.

3.7.1 Configuração do acesso às métricas dos sistemas implementados

O servidor do Prometheus é responsável por manter o banco para armazenamento de dados temporais, o servidor *web* para fácil visualização das métricas e do inventário e também coletar métricas dos seus clientes (PROMETHEUS, c2022b). Essa coleta é realizada através da conexão com exportadores, *softwares* geralmente desenvolvidos pela comunidade para suporte a sistemas operacionais ou aplicações específicas.

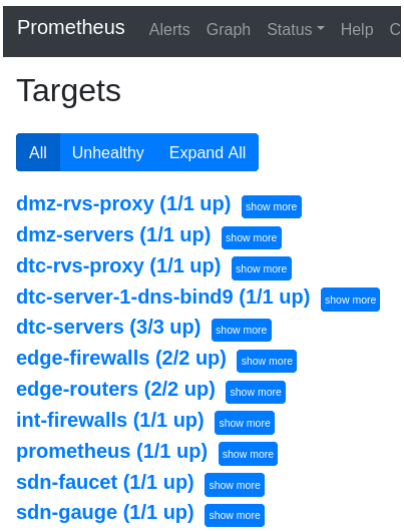


Figura 3.27: *Targets* (ou *hosts*) conectados e disponíveis para coleta de métricas. Visualização pela aplicação *web* do Prometheus.

Esses exportadores são responsáveis por coletarem as métricas providas pelos sistemas em seus *logs* respectivos, formatá-las e disponibilizá-las via uma API HTTP, geralmente exposta em *http://localhost:9100/metrics* (PROMETHEUS, c2022a). Atualmente a conexão com esses *end-points* pode ser feita via protocolo HTTPS, mantendo seguro o transporte das métricas ao servidor do Prometheus. A Figura 3.27 evidencia os chamados *targets*, isso é, os ativos previamente definidos e disponíveis para coleta de métricas.

Os exportadores utilizados para esta proposta foram o *node exporter*, responsável pela coleta de métricas de sistemas Linux, e o *bind exporter*, responsável pela coleta de estatísticas do servidor DNS Bind9 (PROMETHEUS, c2022a). Como já fora mencionado, existem uma infinidade de exportadores para outras distribuições que podem ser aproveitados para complementar ainda mais a visibilidade e o acompanhamento dos servidores, ativos de rede e das aplicações servidas. A visualização de algumas das métricas expostas pelo *node exporter* podem ser visualizadas na Figura

3.28.

```
# HELP node_cpu_guest_seconds_total Seconds the CPUs spent in guests (VMs) for each mode.
# TYPE node_cpu_guest_seconds_total counter
node_cpu_guest_seconds_total{cpu="0",mode="nice"} 0
node_cpu_guest_seconds_total{cpu="0",mode="user"} 0
node_cpu_guest_seconds_total{cpu="1",mode="nice"} 0
node_cpu_guest_seconds_total{cpu="1",mode="user"} 0
# HELP node_cpu_seconds_total Seconds the CPUs spent in each mode.
# TYPE node_cpu_seconds_total counter
node_cpu_seconds_total{cpu="0",mode="idle"} 502.43
node_cpu_seconds_total{cpu="0",mode="iowait"} 1145.6
node_cpu_seconds_total{cpu="0",mode="irq"} 0
node_cpu_seconds_total{cpu="0",mode="nice"} 2.98
node_cpu_seconds_total{cpu="0",mode="softirq"} 9.61
node_cpu_seconds_total{cpu="0",mode="steal"} 3.25
node_cpu_seconds_total{cpu="0",mode="system"} 19.91
node_cpu_seconds_total{cpu="0",mode="user"} 9.64
node_cpu_seconds_total{cpu="1",mode="idle"} 532.69
node_cpu_seconds_total{cpu="1",mode="iowait"} 1120.51
node_cpu_seconds_total{cpu="1",mode="irq"} 0
node_cpu_seconds_total{cpu="1",mode="nice"} 3.49
node_cpu_seconds_total{cpu="1",mode="softirq"} 0.06
node_cpu_seconds_total{cpu="1",mode="steal"} 3.04
node_cpu_seconds_total{cpu="1",mode="system"} 24.35
node_cpu_seconds_total{cpu="1",mode="user"} 11.88
```

Figura 3.28: Trecho de métricas dos sistemas Linux expostas para o Prometheus pelo exportador *node exporter*.

3.7.2 Criação e ajuste de dashboards

Com o Grafana configurado na topologia e sua base de dados sendo constantemente alimentada pelo Prometheus, foram criados diferentes *dashboards* para a visualização gráfica e intuitiva dos dados. As Figuras 3.29 e 3.30 apresentam alguns dos *dashboards* definidos para esta proposta, com acesso através do endereço *dtc-proxy.pfg.com/grafana* pelo *desktop* de gerência.

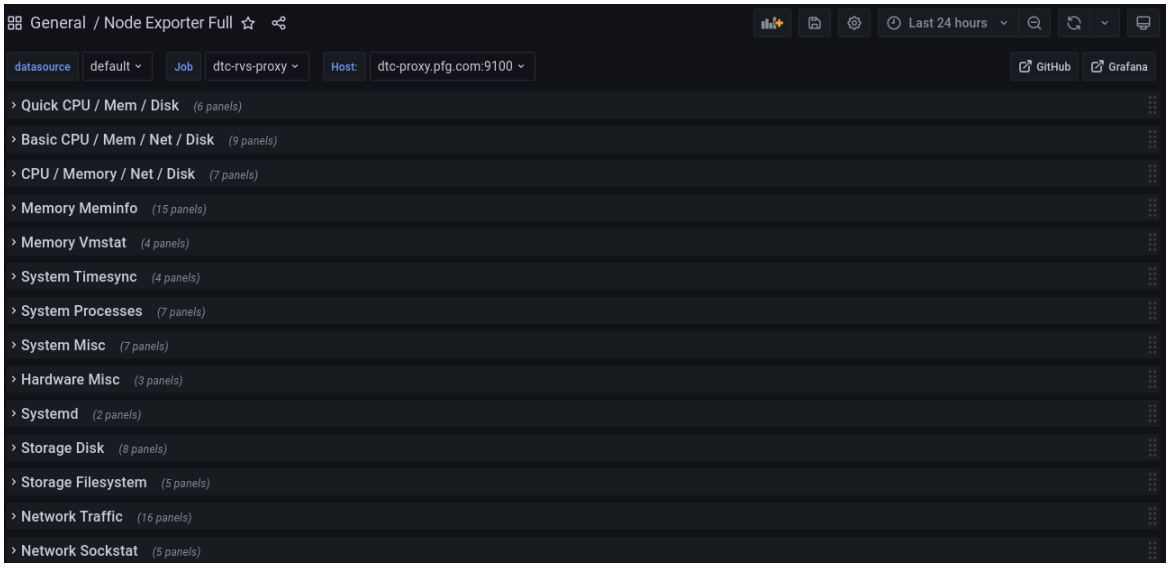


Figura 3.29: Exemplo de *dashboard* do Grafana para visualização de métricas dos sistemas Linux com múltiplos painéis distintos.

A Figura 3.29 apresenta um exemplo de *dashboard* obtido pelo repositório público da comunidade do Grafana que permite a visualização de mais de 100 painéis distintos criados a partir das métricas exportadas pelo *node exporter* dos sistemas Linux. A criação de *dashboards* permite agregar diferentes painéis em uma única tela de visualização e cada um desses painéis pode ser alimentado com qualquer fonte de dados configurada no Grafana. A visualização pode até mesmo

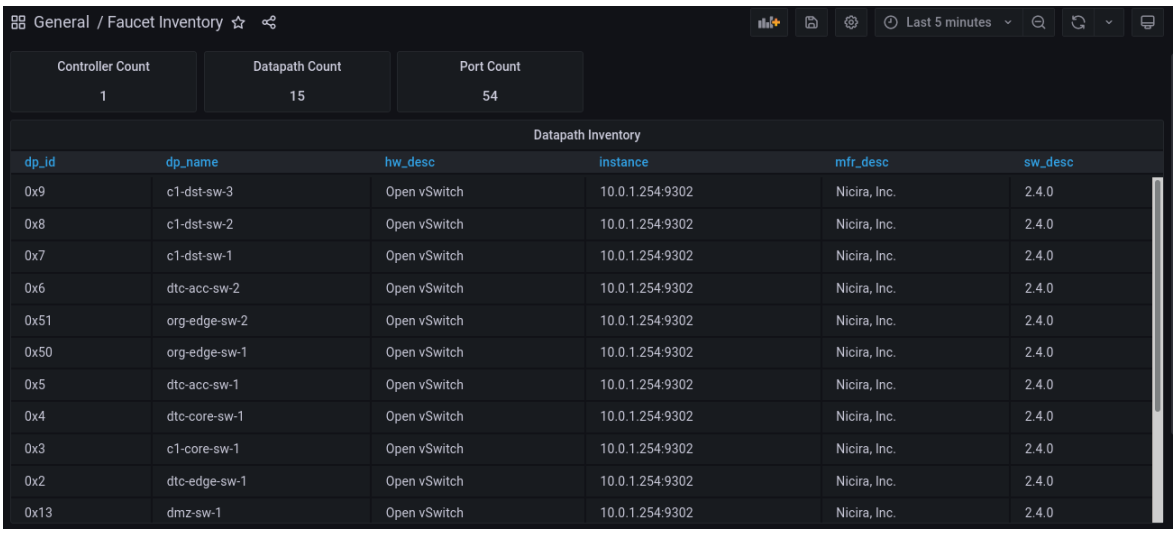


Figura 3.30: Inventário dos *switches* (*datapaths*) conectados à controladora Faucet e visualizados pelo Grafana.

ser individualizada para cada um dos ativos cadastrados, logo é possível segmentar os dados apresentados em cada um dos painéis para um *job* ou *host* específico.

Já a Figura 3.30 apresenta um dos *dashboards* disponibilizados pela controladora Faucet. Neste caso, é apresentado o inventário de *switches* conectado à controladora, de forma que se torna fácil manter controle dos dispositivos conectados ou não. Além disso, há outros *dashboards* disponibilizados para visualização das métricas do sistema e também para visualização do tráfego de rede em cada uma das portas de cada um dos *switches* conectados, o que também pode auxiliar em demais análises.

3.7.3 Definição de alertas e serviço de mensageria pelo Telegram

O gerenciamento de métricas e eventos em um NSOC é complexo devido à grande quantidade de informações geradas, necessitando de uma equipe experiente e dedicada para tal atividade. Portanto, é imprescindível utilizar-se de ferramentas que possam agregar tais informações e produzir um resumo preciso do estado atual da rede e dos sistemas. Uma dessas ferramentas é o sistema de alertas, disponível para diversas ferramentas de gerência.

Através da agregação de dados de eventos em tempo real proporcionado pela integração entre Prometheus e Grafana, é possível definir alertas para sinalizar situações possivelmente anormais. Esses alertas podem ser enviados por diversos canais de comunicação, como email, Slack, *webhooks*, entre outros. O Telegram foi o canal escolhido para esta proposta por sua facilidade de integração e visualização das ocorrências.

Para esta integração entre Grafana e Telegram basta utilizar do *bot @BotFather* para inserção de um *bot* em um canal de anúncios específico e configurar a conexão dos canais de notificação no Grafana. Os demais detalhamentos desta etapa serão abstraídos deste estudo, mas uma notificação de teste desta integração pode ser visualizado na Figura 3.31.

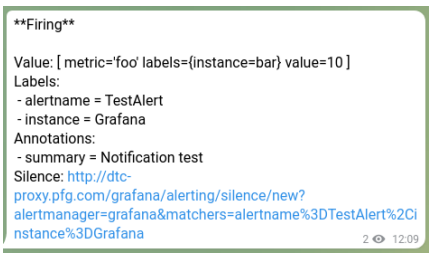


Figura 3.31: Teste de notificação de alerta pelo Telegram enviado pelo Grafana.

A criação de alertas pode ser realizada para qualquer painel incluso em uma *dashboard* existente, logo pode ser configurado para diversos casos imagináveis. Um exemplo prático, utilizado nesta proposta e visualizado na Figura 3.32, foi a definição de um alerta para detecção de conexões TCP estabelecidas no *proxy* reverso da área DMZ, *dmz-rvs-proxy-1*, acima de um limite pré-definido. Neste caso, uma situação anormal em que haja um número de conexões acima do limite pode significar um grande número de usuários se conectando simultaneamente ou uma tentativa de ataque de negação de serviço, como o ataque *TCP SYN Flood*.

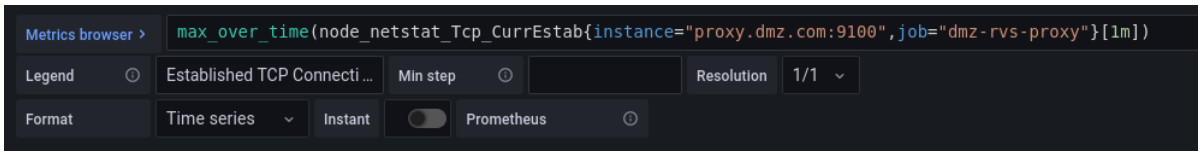


Figura 3.32: *Query* em formato PromQL para definição de alerta no Grafana com o objetivo de observar quantidade de conexões TCP atípicas no servidor *dmz-rvs-proxy-1*.

Apesar dos alertas serem utilizados para notificar situações anormais, é importante ressaltar que estes não necessariamente significam problemas ou tentativas de ataque, mas sinalizam a necessidade de intervenção humana para avaliação do estado e definição de respostas.

3.8 Solução de gerenciamento de eventos e informações de segurança com o sistema Wazuh

Os *logs* gerados pelos sistemas operacionais, ferramentas e aplicações apresentam ricas informações sobre sua execução. Na maioria dos casos, os *logs* são gerados em uma frequência acelerada, tornando impossível o monitoramento em tempo real por um ser humano, experiente ou não. Em uma implementação extensa, como esta proposta, o gerenciamento de uma quantidade elevada de arquivos de *log* se torna uma tarefa ainda mais difícil. Para simplificar essa situação e garantir observabilidade dos eventos ocorridos na topologia, são desenvolvidos diversos sistemas capazes de coletar, aprimorar, agregar e analisar estes *logs* e, em um contexto de segurança da informação, são definidos os SIEMs (*Security Information and Management Systems*).

Assim como para os demais sistemas apresentados neste estudo, as soluções *open-source* de SIEMs têm recebido cada vez mais suporte de desenvolvimento, mas essa ainda é uma das áreas menos avançada nesse aspecto. Os SIEMs mais utilizados são pagos e apresentam custos elevados

até para pequenas e médias organizações, principalmente devido à sua complexidade de implementação e disponibilização de suporte qualificado. Em geral, um SIEM é responsável por realizar a orquestração de múltiplas ferramentas e sistemas, integrando a maior quantidade possível de informações para disponibilização de dados relevantes aos administradores de rede.

Entre os SIEMs *open-source* e "gratuitos", alguns das principais soluções são o SecurityOnion e o SIEMonster. Esses sistemas apresentam serviços em *cloud* e também serviços de suporte pagos, mas também estão disponíveis gratuitamente. Entre os sistemas que essa solução entrega está o Wazuh, que é o sistema escolhido para esta proposta.

Devido às limitações de infraestrutura física já descritas, não é possível utilizar um dos demais sistemas descritos por apresentarem altos requerimentos de memória e processamento. Contudo, o Wazuh é um sistema SIEM individual e completo que apresenta os requisitos necessários para o desenvolvimento desta proposta. O Wazuh atua principalmente sobre *hosts* como uma forma de EDR (*endpoint detection and response*), mas também realiza a análise de *logs* de diversas origens. Através de sua implementação, estão disponíveis o acesso a ferramentas de detecção de vulnerabilidades, detecção de intrusão em *hosts* (HIDS), monitoramento de integridade de arquivos, coleta e análise de *logs*, entre outras que serão descritas. (WAZUH, c2022c)

3.8.1 Implementação do Wazuh, Elasticsearch, Kibana e Filebeat

O Wazuh é implementado inicialmente com base em dois componentes principais: wazuh-manager e wazuh-agent (WAZUH, c2022b). O agente é o software implementado em cada *host* que se deseja monitorar, como uma forma de EDR, e este é capaz de coletar arquivos de *logs*, realizar análise de vulnerabilidade no sistema operacional e em *softwares* instalados, entre outros. Para esta proposta, serão analisado apenas essas capacidades do Wazuh, de forma que possíveis aprimoramentos sejam realizados em trabalhos futuros. O outro componente, wazuh-manager, é o servidor responsável pela agregação das informações coletadas pelos agentes, de forma que também é responsável por gerenciar a autenticação e configuração dos agentes. Tanto o processo de autenticação quanto o processo de transferência de dados entre agente e servidor é realizado através de implementações do protocolo TLS, garantindo privacidade às informações trafegadas pela rede. É importante ressaltar que o servidor do Wazuh pode ser definido em modo individual ou em modo *cluster*, disponibilizando uma estrutura distribuída para garantir disponibilidade e escalabilidade. Por limitações da infraestrutura, foi utilizada a versão individual nesta proposta.

O sistema do Wazuh também conta com a integração de diversas outras ferramentas *open-source*, incluindo Elasticsearch, Kibana e Filebeat (WAZUH, c2022b). A versão utilizada para os sistemas Elasticsearch e Kibana são derivações *open-source*, mantidas pela Amazon com o nome de Open Distro, da versão oficial mantida pelo Elasticsearch. Essas duas ferramentas são utilizadas para indexação, análise de *logs* e visualização gráfica dos mesmos. Já a ferramenta Filebeat é utilizada para coleta e envio dos arquivos de *log* para o Elasticsearch.

O Wazuh utiliza dessa integração entre as demais ferramentas para auxiliar na sua análise e, consequentemente, proporcionar informações com mais eficiência e detalhamento. Para o gerenci-

amento do Wazuh, é utilizado um *frontend* desenvolvido como *plugin* para o Kibana, agregando diversos tipos de informações em painéis para visualização. Um exemplo dos painéis gerados pode ser visualizado na Figura 3.33.

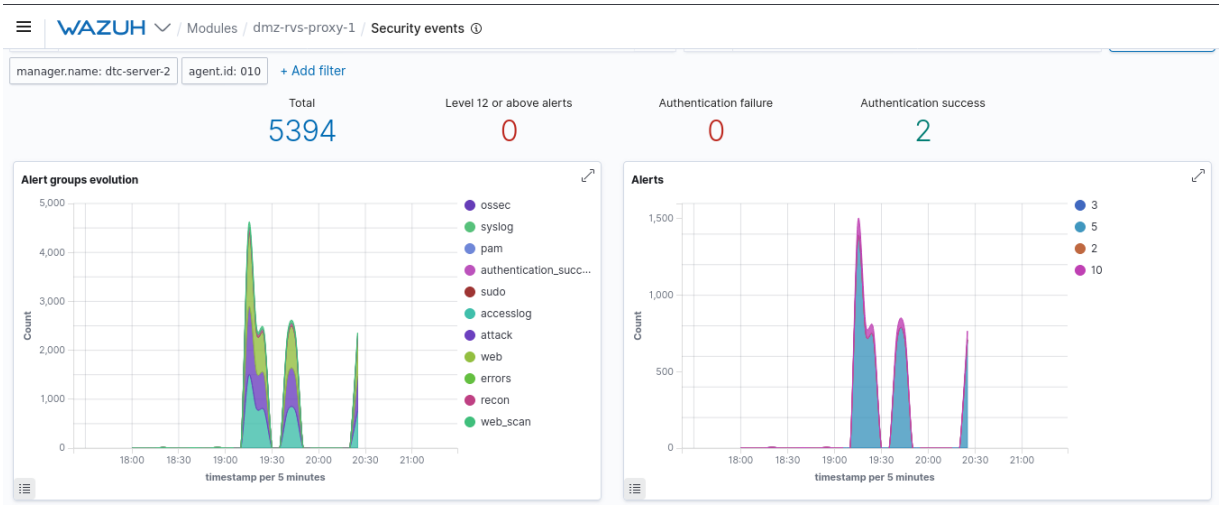


Figura 3.33: Apresentação dos painéis de visualização do *plugin* do Wazuh para o Kibana. Exemplo da agregação de eventos de segurança para o agente *dmz-rvs-proxy-1*.

3.8.2 Monitoramento de agentes com Wazuh Agent

Como já definido, os agentes do Wazuh atuam como EDR nos *hosts* instalados, de forma a coletar informações, enviá-las ao servidor central e implementar respostas quando necessário. A autenticação é um dos primeiros passos realizados para garantir a conexão segura entre um *host* confiável e o servidor central, evitando a contaminação de dados por fontes indevidas.

O controle de agentes também pode ser realizado graficamente pela interface disponível no Kibana, gerenciado em forma de inventário. Essa visualização, que pode ser observada na Figura 3.34, descreve o *hostname* de cada agente, endereço IP, sistema operacional utilizado e o estado atual da conexão. Além disso, é importante ressaltar que todo o *dashboard* é orientado com base nos agentes, dado que eles originam os dados analisados. Sendo assim, é possível filtrar cada painel de eventos e informações com base em um ou múltiplos agentes.

Apesar da ferramenta wazuh-agent ser um software multiplataforma, ainda há impeditivos de sua instalação em ativos como alguns roteadores e *firewalls*, por exemplo. Para estes casos, estão disponíveis formas de monitoramento remotas, como a transferência de *logs* por um servidor *syslog*.

3.8.3 Análise de vulnerabilidade dos agentes

Assim como já definido anteriormente, o Wazuh é capaz de realizar análise de vulnerabilidades conhecidas de um sistema operacional e dos softwares nele instalados. Esta análise é ativada por padrão na configuração dos agentes compatíveis e os resultados podem ser visualizados pelo *plugin* do Kibana. Tal como pode ser observado na Figura 3.35, é realizado um *benchmark* do estado atual

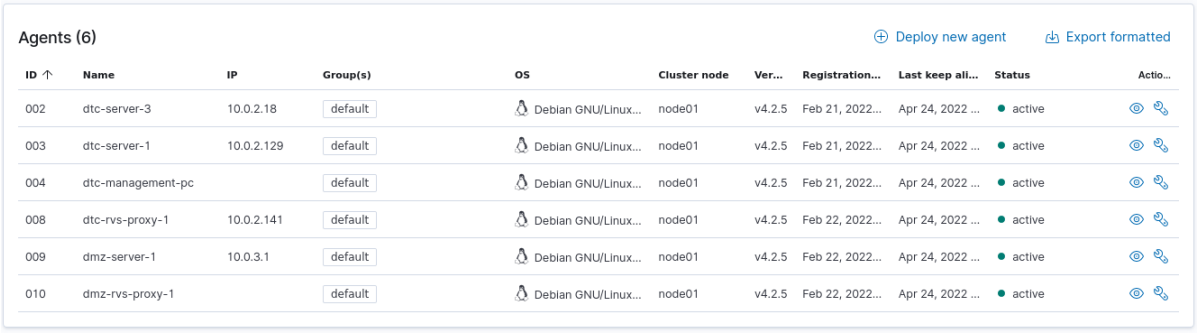


Figura 3.34: Inventário de agentes ativos conectados ao servidor do Wazuh. Visualização disponível pelo *plugin* do Wazuh no Kibana.

do sistema através das políticas de segurança providas pela CIS Benchmarks (WAZUH, c2022a).

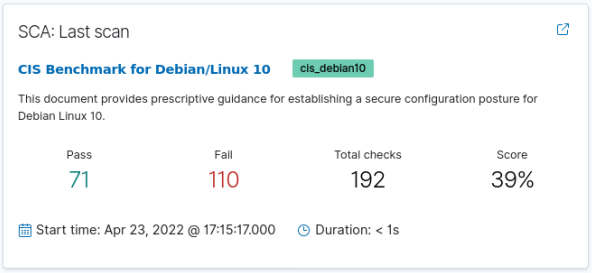


Figura 3.35: Análise de vulnerabilidades do servidor *dmz-rvs-proxy-1* pelo Wazuh.

Além das informações de *score* geral, cada uma das regras avaliadas pode ser manualmente verificada pelo próprio *dashboard*, proporcionando a análise de detalhes específicos de cada uma. A descrição de cada regra também contém o método de mitigação apropriado, de forma que possam ser realizados ajustes das vulnerabilidades detectadas. Um exemplo desse caso pode ser visualizado na Figura 3.36, que apresenta um alerta sobre a má configuração do *banner* do sistema (*message of the day*), dado que este especifica detalhes privados do sistema quando um usuário realiza *login* via SSH, por exemplo. A correção dessas vulnerabilidades não são obrigatoriamente necessárias para todos os casos, mas esta ferramenta é uma fonte de informações essencial para prevenção da exploração de vulnerabilidades ocultas.

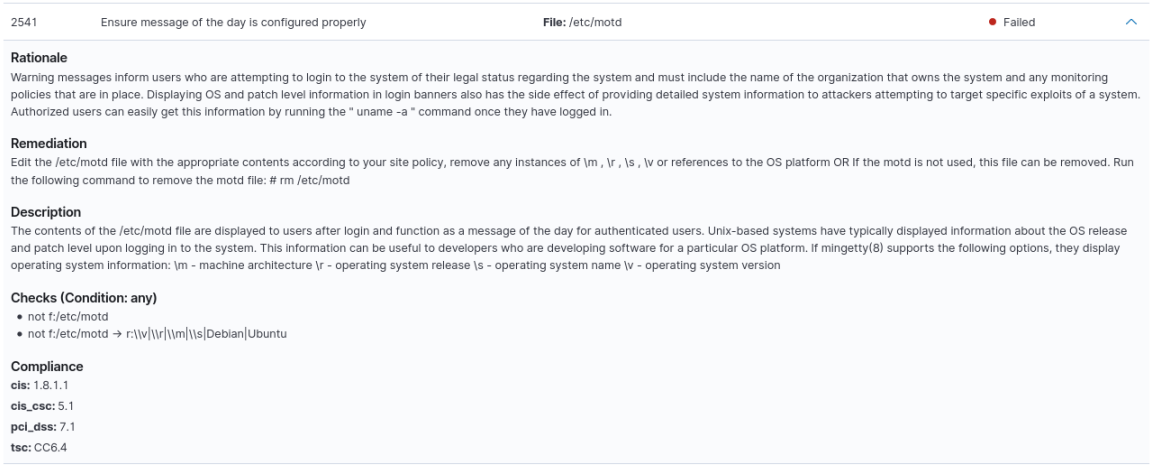


Figura 3.36: Exemplo detalhado de vulnerabilidade avaliada pelas políticas da CIS Benchmark para o sistema Debian. Visualização proporcionado pelo *plugin* do Wazuh no Kibana.

Capítulo 4

Análise de Resultados

Através da arquitetura proposta e implementada, torna-se necessária a análise de sua execução com a integração total entre os sistemas e ferramentas implantados. Consequentemente, também foi proposta a definição de diversos cenários para teste. O escopo se baseia em dois componentes principais, que também são completamente relacionados entre si, alta disponibilidade e monitoramento de segurança. Nesta seção, são apresentados os resultados obtidos para cada cenário apresentado, demonstrando concretização dos objetivos e da solução proposta.

4.1 Análise de alta disponibilidade da topologia

Como já definido e explicado, foram empregados níveis de redundância em diversas camadas da infraestrutura proposta. Cada configuração realizada é única e específica, de modo que cada uma dessas implementações resulta em um grande valor para o objetivo desse estudo. Cada nível será descrito e exemplificado nessa seção como forma de demonstrar efetividade das soluções.

4.1.1 Roteadores

Os roteadores *org-edge-router-1* e *org-edge-router-2*, localizados na borda da organização, são responsáveis pela conexão da organização com o meio externo, a Internet. Sendo assim, há uma grande compromisso desses dispositivos em disponibilizarem a vazão necessária de tráfego tanto para a rede interna quanto para a rede externa. O tráfego externo, originado da Internet, visa o acesso à zona DMZ e as aplicações servidas, nesse caso a aplicação *web* HTTPS. Já o tráfego interno visa o acesso de usuários dos *campus* e visitantes tanto para a zona DMZ quanto para o tráfego HTTPS da Internet.

Para realizar a comunicação com o meio externo, ambos roteadores são configurados como *peers* da conexão eBGP com as ASs públicas do ISP1 e ISP2. Inicialmente, já é possível visualizar a implementação redundantes através dos múltiplos enlaces dedicados com os ISPs. Através dessa configuração, estão definidos quatro pontos de entrada e saída para a organização, que são organizados de acordo com pesos e métricas definidas. Os enlaces principais estão estabelecidos nas

interfaces eth0 de ambos os roteadores de borda, sendo então os enlaces diagonais, visualizados na Figura 3.2, configurados como conexões de *backup* ou *failover*. Há três cenários principais que podem testar a efetividade da redundância dos roteadores: a queda do enlace principal, a queda do enlace principal e do enlace secundário ou a queda do roteador.

O dispositivo para avaliação desses casos será o roteador *org-edge-router-1*, que desempenha papel equivalente ao *org-edge-router-2*, não sendo estabelecida qualquer desproporção de tráfego entre ambos. Inicialmente, em estado de operação regular, as rotas para a rede WAN dos ISPs são definidas assim como a Figura 3.7, tendo a interface eth0 como saída principal, seguido da interface de *failover* e, por fim, a conexão *ebgp-multihop* através do seu *peer* iBGP, o roteador *org-edge-router-2*. Com uma falha no enlace principal, o roteador perde conexão com o *peer* eBGP principal e a tabela de rotas é automaticamente adaptada para direcionar o fluxo pela interface de *failover*, assim como na Figura 4.1.

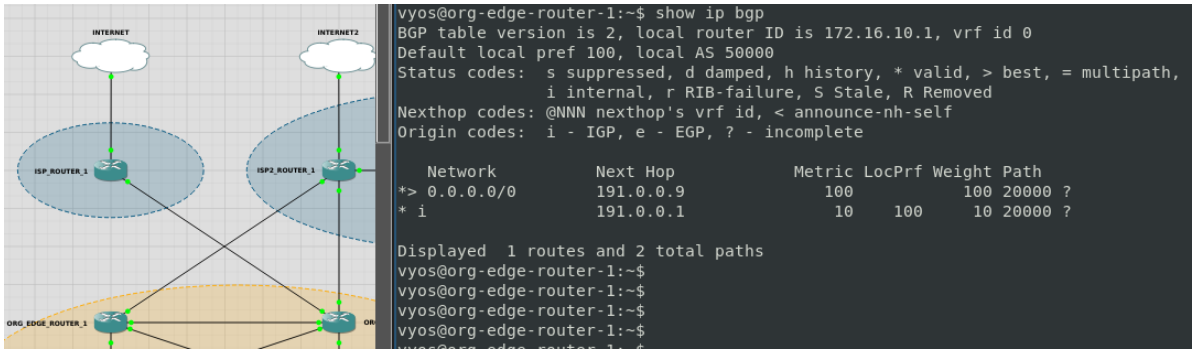


Figura 4.1: Resposta do protocolo BGP com a queda do enlace principal que mantém a conexão eBGP com o ISP1.

Em um caso extremamente improvável, a queda do enlace de *failover* simultaneamente com a queda do enlace principal ainda não é suficiente para interromper o fluxo pelo roteador *org-edge-router-1*, como poder ser visualizado pela Figura 4.2, dado que ainda resta a conexão direta com seu *peer* iBGP que mantém conexão com outros dois enlaces dedicados. A conexão *ebgp-multihop*, se configurada corretamente, poderá manter a conexão eBGP do roteador *org-edge-router-1* até mesmo se a conexão direta com seu *peer* iBGP vizinho também for encerrada, dado que poderá utilizar do roteamento OSPF interno para manter essa ponte.

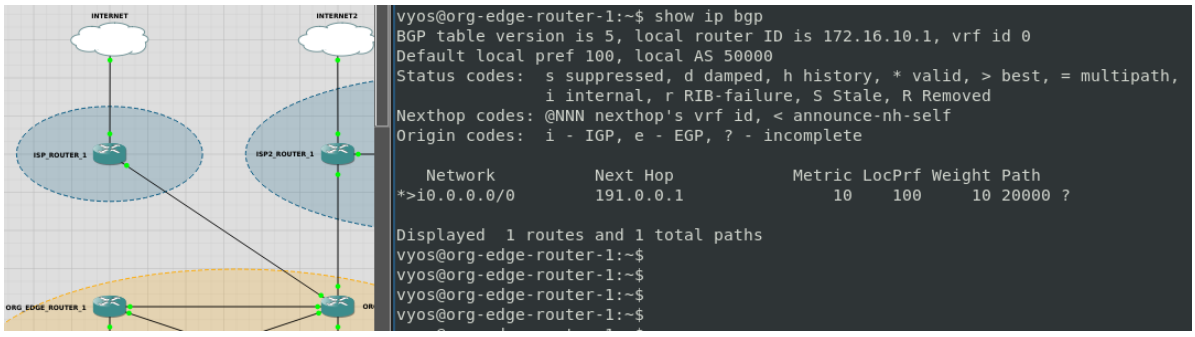


Figura 4.2: Resposta do protocolo BGP com a queda do enlace principal e do enlace de *failover* que mantinham a conexão eBGP com o ISP1 e ISP2, respectivamente.

Por fim, o terceiro cenário é respectivo a um problema interno do roteador, tornando-o indisponível. Nesse caso, a redundância de enlaces não tem efeitos diretos, mas a solução deste cenário recai sobre a redundância de *firewall* e suas interfaces WAN que serão descritas na próxima subseção.

4.1.2 Firewalls

Conforme já foi extensivamente descrito, o sistema de *firewall* PfSense, além de ser nativamente completo e integrado com múltiplas ferramentas, ainda oferece suporte a diversos pacotes com funcionalidades extras. Já incorporado neste sistema reside a configuração de sincronismo entre *firewall* e alta disponibilidade. Através da sincronia, é possível garantir que dois *firewall* possuam as mesmas configurações em tempo real, servindo como um *backup* ou até como uma forma de balanceamento de carga dos níveis inferiores. Além desta característica, foi configurada uma interface virtual CARP, que realiza a agregação de dois endereços IP, de duas interfaces de *firewall* distintos, em um único endereço. Através do gerenciamento automático desse endereço único pelo *firewall* principal, acordado pelas configurações de sincronismo, é possível garantir que sempre haverá um *gateway* acessível, independente de falhas nos enlaces ou indisponibilidade de um dos sistemas.

Assim como explicitamente demonstra a Figura 4.3, a utilização do protocolo CARP na interface LAN permite a acessibilidade contínua dos *firewall*, dado que ao menos um esteja ativo e operacional. No principal cenário de teste, a queda da conexão da interface LAN do *firewall master* resulta automaticamente no redirecionamento do tráfego para o *firewall* secundário, como pode ser visto na Figura 4.4. Mesmo sendo realizado entre *switches* diferentes, a controladora Faucet observa o anúncio ARP do novo endereço do endereço IP CARP e torna esse redirecionamento ainda mais simples.

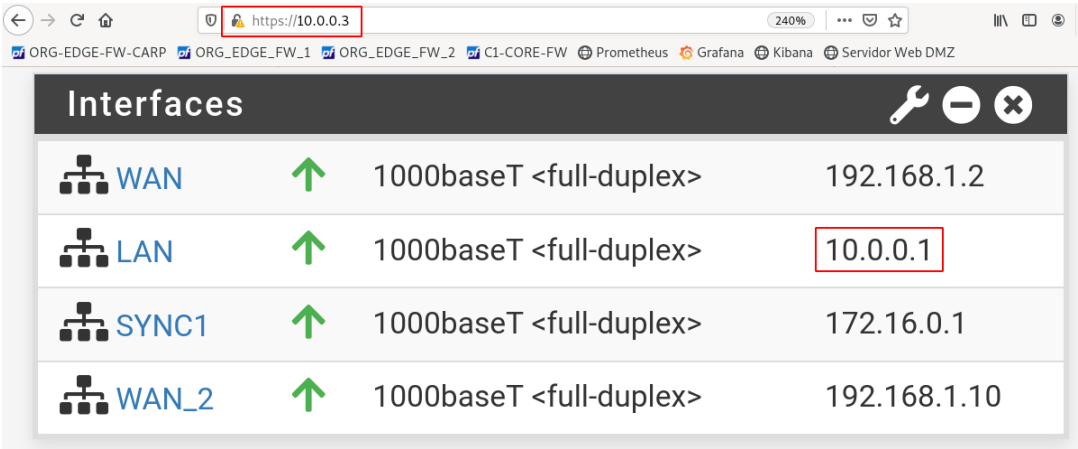


Figura 4.3: Apresentação do resultado da configuração da interface CARP no *firewall* PfSense principal, ou *master*.

Prosseguindo com a lógica da subseção anterior, o sistema de *firewall* também foi configurado para realizar balanceamento de carga pelas suas interfaces WAN conectadas aos roteadores de borda, como pode ser observado na Figura 4.5. Novamente facilitada pela abstração disponível, a

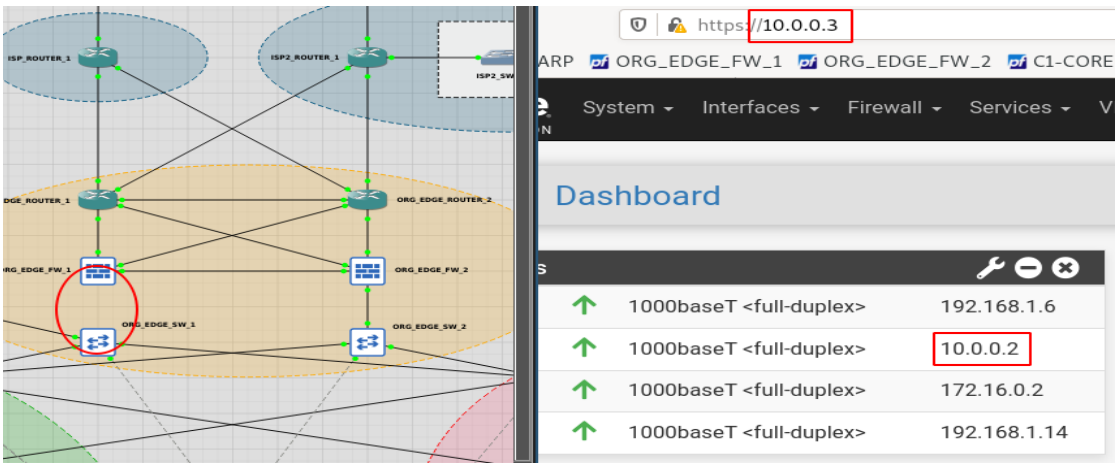


Figura 4.4: Transição de estado do *firewall* secundário para principal, consequência da indisponibilidade do *firewall org-edge-fw-1* pela rede LAN.

configuração deste balanceamento de carga é baseado apenas na definição de grupos de *gateways* e seus pesos. No caso de uma falha no enlace de conexão com um dos roteadores, ou até indisponibilidade do roteador, o protocolo OSPF detectará esse cenário e atualizará a tabela de roteamento da aŕea, eliminando essa rota e forçando todo o tráfego pela única interface disponível. É importante ressaltar que como apenas um *firewall* permanece operacional regularmente, o roteamento OSPF do *firewall* secundário também será mantido desativado, impedindo que fluxos possam fluir por ele automaticamente. Por fim, reforça-se a concepção de que a configuração CARP também poderia ser mantida nas interfaces WAN, mas não foi o objetivo desta proposta.

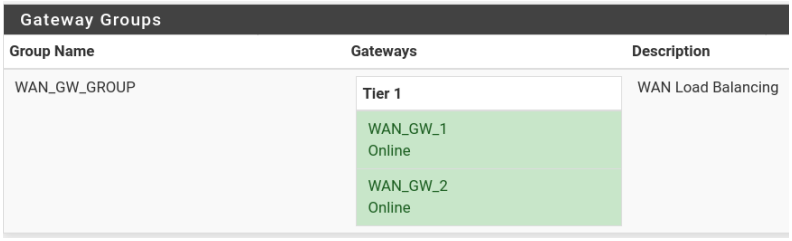


Figura 4.5: Estado do balanceamento de carga das interfaces WAN do *firewall org-edge-router-1*.

4.1.3 Arquitetura SDN

Apesar de não haver um grande esforço atualmente na padronização e no desenvolvimento de novas soluções *open-source* de controladoras SDN, as soluções atuais já disponibilizam um grande arsenal de ferramentas e possibilidades. A controladora Faucet foi profundamente implementada e analisada nesta proposta, garantindo um *backbone* interno extremamente dinâmico, flexível e performático. Além disso, a simplicidade de configuração se tornou um grande benefício após a adaptação inicial com a ferramenta.

A configuração de redundância implementada já foi amplamente descrita, mas não exemplificada. Como a controladora realiza a orquestração de todos os *switches*, ou *datapaths*, esta possui

visibilidade total sobre a rede interna da organização. Embora a centralização pareça uma solução simples e óbvia, lidar com a convergência e definição de melhores rotas em um ambiente dinâmico é uma grande dificuldade, principalmente relacionada ao custo de processamento.

Nesta proposta, foram definidos 15 *switches* Open vSwitch operando sobre OpenFlow em 4 níveis principais de hierarquia respectivos à seu nível de redundância, sendo então: *switches* da borda da organização; *switches* da borda do *campus*, *datacenter* e DMZ; *switches core* do campus e *datacenter*; demais *switches*. Além da visão topológica, a agregação dessas informações em inventário pode ser visualizado na Figura 3.30. Essa definição de hierarquia está sendo realizada para indicar os possíveis pontos de falha da arquitetura proposta. Analisando cada *switch* e seus enlaces, verifica-se que apenas há possíveis pontos de falhas de enlaces nas conexões com os dispositivos finais, dado que estes casos são praticamente inevitáveis. Contudo, ainda estão disponíveis *switches* sujeitos a indisponibilidade, ocorrências estas que podem impedir o tráfego de certa parte da organização. Apesar de ser fácil lidar com esses casos em um ambiente virtual, também é necessário expor que a redundância é uma implementação contínua e custosa, inviável para pequenas e médias organizações, sendo então necessário um estudo aprofundado para definição do equilíbrio entre disponibilidade e custo.

Para garantir alta disponibilidade, todo o *backbone* da organização é redundante, perdendo esta característica nos *switches core*, ponto único de saída de seus dispositivos internos. Em casos como este, é possível monitorar esses eventos através dos *dashboards* providos, como pode ser visto na Figura 4.6. Através desses painéis observados, é possível notar comportamentos anômalos de tráfego e, se integrado com o *dashboard* definido na Figura 3.30, é possível também determinar a falha em um dos *switches* gerenciados. A versatilidade garantida pela controladora Faucet é possível devido às ferramentas oferecidas pelas definições do protocolo OpenFlow, que segue sendo evoluído.

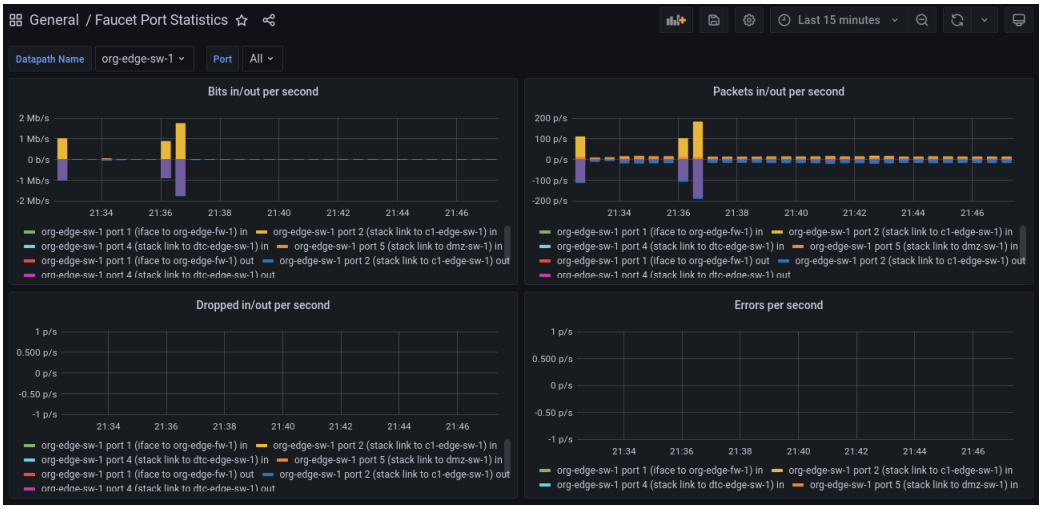


Figura 4.6: Apresentação do *dashboard* de análise de tráfego em cada porta de cada *switch*, proporcionada pela integração entre Faucet (e seu componente Gauge), Prometheus e Grafana.

4.2 Análise do monitoramento de eventos e informações de segurança

De forma a garantir observabilidade dos acontecimentos da organização simulada, foram implementados sistemas de gerenciamento de métricas e informações de segurança. Essa proposta considera a integração entre múltiplas ferramentas, como: Prometheus, Kibana, Elasticsearch, Grafana, Wazuh, entre outros. A implantação e a validação da efetividade do uso de ferramentas *open-source* também garante um resultado significativo para a disseminação do uso e evolução desses sistemas. Esta seção é então dedicada à definição de cenários de ataques e tentativas de intrusão para avaliar a característica de observabilidade proposta, além da proposição de correções e melhorias para os casos analisados.

É importante ressaltar que a realização de tentativas de ataques complexos e variados, em prol da negação de serviços ou penetração dos sistemas, não faz parte do escopo deste estudo, sendo que as atividades realizadas em sequência foram executadas apenas para validação de toda a topologia proposta.

4.2.1 Mapeamento de vulnerabilidades através das ferramentas Nmap e Nikto

O Nmap é uma ferramenta consagrada de análise de vulnerabilidades em *hosts*, sendo possível escanear serviços em execução, modelo e versão dos sistemas operacionais, regras de filtragem do *firewall*, entre outros (NMAP, [s.d.]). Desenvolvido primariamente para auditoria de segurança, esta ferramenta possui dezenas de parâmetros disponíveis para validação das configurações implementadas em um sistema, entre elas parâmetros para utilização em forma furtiva. Nesta proposta, utilizou-se o escaneamento do Nmap em modo agressivo, de forma a avaliar sua capacidade de escaneamento e a capacidade de detecção destes eventos pelos sistemas implementados. As Figuras 4.7 e 4.8 apresentam duas checagens agressivas que apresentam: a descrição geral das portas abertas, os serviços e suas respectivas versões implementadas, tentativa de descrição do sistema operacional, *traceroute* e lista de vulnerabilidades da porta HTTPS aberta. É importante evidenciar que no segundo teste são utilizados 105 *scripts* explorados pela comunidade para análise de vulnerabilidades, que definem algumas potenciais brechas para exploração. Apesar de serem checagens agressivas, pouco é possível observar de resultado no Wazuh sem prévia configuração detalhada, dado que o Nmap é uma ferramenta que utiliza de formas autênticas para tais checagens, misturando-se com falsos positivos usuais, como pode ser visto na Figura 4.9. Nesta imagem são apresentadas apenas mensagens de códigos estranhos de ICMP e erro no *TCP handshake*, mensagens essas que podem ser observadas a todo momento em uma implementação real, não apresentando potencial de risco imediato. Reitera-se que o Nmap, por si só, é apenas uma ferramenta de escaneamento, mas que pode descrever vulnerabilidades para outras formas de ataque.

O Nikto já é apresentado como uma ferramenta extremamente agressiva e não indicada para escaneamentos furtivos (CIRT.NET, c2022). Esta ferramenta é utilizada para análise de vulnerabilidades indicadas pela versão do servidor e pelas suas configurações. A utilização desta ferramenta é extremamente simples e o resultado pode ser visualizado na Figura 4.10. A ausência de resulta-

```
┌──(osboxes㉿internal-attacker)-[~]
└─$ sudo nmap -A -T4 proxy.dmz.com
Starting Nmap 7.91 ( https://nmap.org ) at 2022-04-26 22:48 EDT
Nmap scan report for proxy.dmz.com (10.0.3.13)
Host is up (0.0081s latency).
Not shown: 999 filtered ports
PORT    STATE SERVICE  VERSION
443/tcp open  ssl/http nginx 1.18.0
|_http-server-header: nginx/1.18.0
|_http-title: Site doesn't have a title (text/plain).
| ssl-cert: Subject: commonName=proxy.dmz.com/organizationName=UNB/stateOrProvinceName=Brasilia/countr
yName=BR
| Subject Alternative Name: DNS:proxy.dmz.com
| Not valid before: 2022-04-10T21:08:12
|_Not valid after:  2032-04-07T21:08:12
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|media device
Running (JUST GUESSING): Linux 2.6.X|3.X|4.X|5.X (93%), Dish embedded (90%)
OS CPE: cpe:/o:linux:linux_kernel:2.6.32 cpe:/h:dish:hopper cpe:/o:linux:linux_kernel:3 cpe:/o:linux:l
inux_kernel:4 cpe:/o:linux:linux_kernel:5
Aggressive OS guesses: Linux 2.6.32 (93%), Dish Network Hopper media device (90%), Linux 2.6.32 - 2.6.
35 (90%), Linux 3.10 - 4.11 (87%), Linux 3.11 - 4.1 (87%), Linux 3.2 - 3.8 (87%), Linux 3.2 - 4.9 (87%
), Linux 2.6.32 - 3.10 (86%), Linux 2.6.32 - 3.13 (86%), Linux 4.15 - 5.6 (86%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 3 hops

TRACEROUTE (using port 443/tcp)
HOP RTT      ADDRESS
1   8.46 ms  10.1.2.1
2   ...
3   11.88 ms 10.0.3.13

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.15 seconds
```

Figura 4.7: Saída da execução da ferramenta de escaneamento Nmap com parâmetros agressivos.

```
┌──(osboxes㉿internal-attacker)-[~]
└─$ sudo nmap --script vuln proxy.dmz.com
Starting Nmap 7.91 ( https://nmap.org ) at 2022-04-26 22:47 EDT
Nmap scan report for proxy.dmz.com (10.0.3.13)
Host is up (0.0055s latency).
Not shown: 999 filtered ports
PORT    STATE SERVICE
443/tcp open  https
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to debug)
| http-phpmyadmin-dir-traversal:
|   VULNERABLE:
|   phpMyAdmin grab_globals.lib.php subform Parameter Traversal Local File Inclusion
|     State: UNKNOWN (unable to test)
|     IDs:  CVE:CVE-2005-3299
|       PHP file inclusion vulnerability in grab_globals.lib.php in phpMyAdmin 2.6.4 and 2.6.4-pl1 all
ows remote attackers to include local files via the $__redirect parameter, possibly involving the subf
orm array.
|
|     Disclosure date: 2005-10-nil
|     Extra information:
|       ../../../../../etc/passwd :
|   Servidor WEB - HTTPS
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2005-3299
|_      http://www.exploit-db.com/exploits/1244/
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_sslv2-drown:

Nmap done: 1 IP address (1 host up) scanned in 76.75 seconds
```

Figura 4.8: Saída da execução da ferramenta Nmap para exploração de vulnerabilidades nas portas abertas através de uma lista de *scripts*.

Suricata: Alert - SURICATA ICMPv4 unknown code	3	86601
Suricata: Alert - SURICATA ICMPv4 unknown code	3	86601
Suricata: Alert - SURICATA ICMPv4 unknown code	3	86601
Suricata: Alert - SURICATA ICMPv4 unknown code	3	86601
Suricata: Alert - ET INFO TLS Handshake Failure	3	86601

Figura 4.9: Apresentação dos poucos alertas gerados pelo Suricata e agregados no Wazuh devido ao escaneamento do Nmap.

dos profundos é relativo à boa configuração realizada sobre o *proxy* NGINX, de forma que a lista de vulnerabilidades após a instalação do mesmo era extensa. Apesar dos resultados dessas ferramentas não indicarem certeza das exposições descritas, a análise humana individual e específica é necessária para assegurar confiança nos sistemas implementados. Como forma de provar seu impacto, múltiplas mensagens de alertas de nível elevados são geradas pelas tentativas de exploração realizadas pelo Nikto, como visto na Figura 4.11.

```
┌──(osboxes㉿internal-attacker)-[~]
└─$ sudo nikto -h https://proxy.dmz.com/webserver

[sudo] password for osboxes:
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.0.3.13
+ Target Hostname:    proxy.dmz.com
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=proxy.dmz.com/C=BR/ST=Brasilia/L=DF/O=UNB/OU=PFG
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /CN=internal-ca/C=BR/ST=Brasilia/L=DF/O=UNB/OU=PFG
+ Start Time:         2022-04-26 23:16:44 (GMT-4)
---------------------------------------------------------------------------
+ Server: nginx/1.18.0
+ The site uses SSL and Expect-CT header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ 7868 requests: 0 error(s) and 1 item(s) reported on remote host
+ End Time:           2022-04-26 23:22:12 (GMT-4) (328 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figura 4.10: Saída da execução da ferramenta Nikto sobre o servidor *web* da zona DMZ.

4.2.2 Ataque de negação de serviço (DoS) através da ferramenta Slowloris

O Slowloris é uma ferramenta de ataque que visa a negação de serviço (DoS) em servidores *web* HTTP/HTTPS, sendo sua principal característica o baixo consumo de largura de banda. De acordo com Cloudflare (2022), o ataque ocorre já na camada de aplicação, através da execução de múltiplas requisições ao servidor *web* simultaneamente, mascaradas como tráfego usual. Quando é atingido o número máximo de "conexões estabelecidas"suportadas pelo servidor em ataque, este se torna indisponível para novas conexões, concluindo o ataque de negação de serviço. Mesmo através do baixo consumo de banda, mantendo o canal de comunicação relativamente livre, o servidor *web*

dmz-rvs-proxy-1	T1055	T1083	Defense Evasion, Privilege Escalation, Discovery	Multiple common web attacks from same source ip.	10	31153
dmz-rvs-proxy-1	T1055 T1190	T1083	Defense Evasion, Privilege Escalation, Discovery, Initial Access	Common web attack.	6	31104
dmz-rvs-proxy-1	T1055 T1190	T1083	Defense Evasion, Privilege Escalation, Discovery, Initial Access	Common web attack.	6	31104
dmz-rvs-proxy-1	T1190		Initial Access	A web attack returned code 200 (success).	6	31106
dmz-rvs-proxy-1	T1190		Initial Access	A web attack returned code 200 (success).	6	31106
dmz-rvs-proxy-1	T1190		Initial Access	A web attack returned code 200 (success).	6	31106

Figura 4.11: Apresentação dos alertas do Suricata, agregados pelo Wazuh, originados pela execução da exploração de vulnerabilidade do *software* Nikto.

é incapaz de estabelecer novas conexões dado que as conexões são, de fato, válidas, mas exploradas indevidamente.

Nesta proposta, foi utilizado o script desenvolvido por Yaltirakli (2015), em linguagem python, para execução do ataque Slowloris. A Figura 4.12 evidencia o ataque sendo realizado pelo atacante interno, situado na VLAN de acesso aos visitantes. Importante notar que são definidos 5000 *sockets* para realização das requisições HTTPS ao servidor *dmz-rvs-proxy*, ou seja, o *proxy* reverso NGINX da zona DMZ. Como o *proxy* realiza a intermediação entre as requisições e o servidor *web*, uma falha neste sistema consequentemente causará inacessibilidade à aplicação *web*. Para esta análise virtual em infraestrutura limitada, não foram realizados ajustes na quantidade padrão de conexões suportadas pelo NGINX, sendo então suportadas apenas 1024 conexões simultâneas. Fica evidente na Figura 4.13 que o servidor *web* DMZ encontra-se indisponível, dado o esgotamento dos descritores de arquivos do sistema operacional que manejam as conexões, indicado pelo erro "PR_END_OF_FILE_ERROR"(NELSON, 2014). Com ajustes apropriados, um servidor é capaz de lidar com até mais de 1 milhão de conexões simultâneas, mas de qualquer forma, apenas este fator não o torna invulnerável a ataques de negação de serviço.

```
┌──(osboxes㉿internal-attacker)-[~/slowloris/slowloris]
└─$ python3 slowloris.py -s 5000 -p 443 -ua 10.0.3.13
[16-04-2022 15:06:05] Attacking 10.0.3.13 with 5000 sockets.
[16-04-2022 15:06:05] Creating sockets...
[16-04-2022 15:06:10] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:06:27] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:06:44] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:07:01] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:07:18] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:07:35] Sending keep-alive headers... Socket count: 1021
[16-04-2022 15:07:53] Sending keep-alive headers... Socket count: 1021
^C[16-04-2022 15:07:56] Stopping Slowloris
```

Figura 4.12: Execução do script de ataque Slowloris pelo atacante interno com destino ao *proxy* reverso da zona DMZ.

O comportamento anômalo da rede pode ser notado através do Grafana, que registra uma

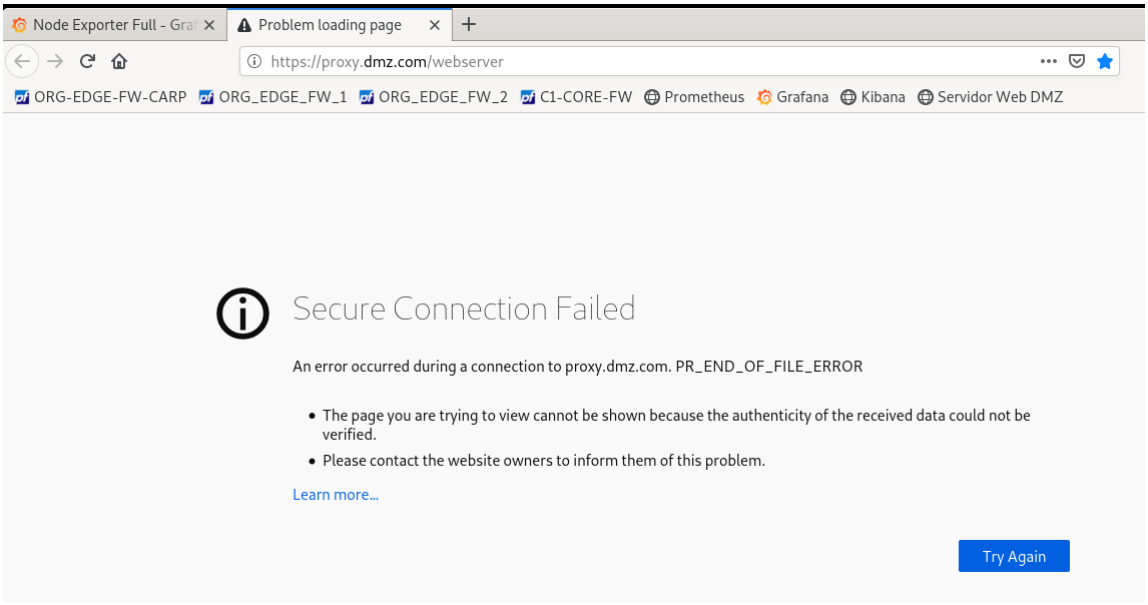


Figura 4.13: Apresentação da indisponibilidade do servidor *web* devido ao ataque Slowloris. Observa-se o esgotamento de *file descriptors* para manejo das conexões.

alta quantidade de conexões TCP estabelecidas, longe do padrão observado na topologia, sendo este registro ilustrado na Figura 4.14. Apesar de serem evidentes os indicativos do incidente, não é possível analisar com assertividade se está sendo realizado um ataque ou apenas um grande número de conexões simultâneas de usuários reais. Portanto, essa ferramenta deve ser principalmente utilizada para otimizar o tempo de resposta dos administradores da rede, facilitando a observação dos eventos atípicos.

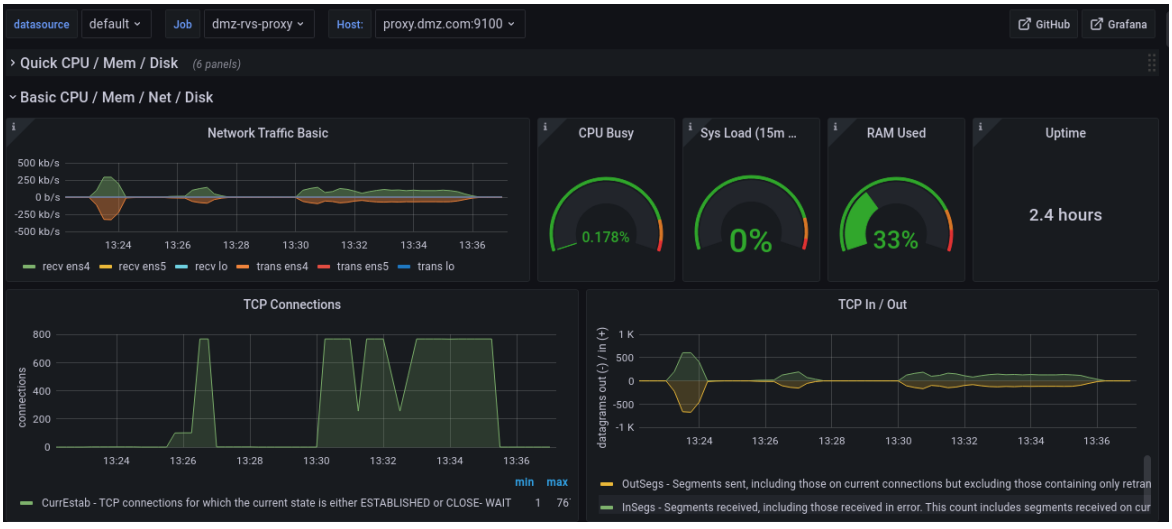


Figura 4.14: *Dashboard* do Grafana no momento do ataque Slowloris, evidenciando valores anômalos no painel de conexões TCP estabelecidas "TCP Connections".

Através das diversas funcionalidades do Grafana, também é possível configurar alertas para notificação da quebra de limites pré-estabelecidos. Nesta proposta, foi utilizado desta funcionalidade para integração com o serviço Telegram, já descrito anteriormente. Através da *query* definida na

Figura 3.32, foi estabelecido um alerta para notificação quando este painel ultrapasse o limite de 500 conexões simultâneas, ajustado para o cenário proposto. Como pode ser visualizado na Figura 4.15, são disponibilizados dois estados de alerta, o disparo e a resolução. Conforme observado, são descritas informações do painel que gerou o alerta e também o *host* de origem do mesmo, agilizando a inspeção do evento e a proposta de uma possível solução. Com a resolução deste evento e a normalização do número de conexões, também é emitido o complemento do alerta anterior, indicando que a situação foi normalizada.

Assim como pode ser visto na Figura 4.16, este ataque também aciona alertas na aplicação Wazuh provenientes da regras definidas no Suricata, indicando a recepção de um pacote TCP FIN fora da janela esperada para o fluxo. Apesar dessa categoria de regras ser geralmente tratada como falso positivo, é valido destacar que também é possível relacionar os demais resultados já descritos com o monitoramento de eventos pelo Wazuh para uma completa avaliação do acontecimento.

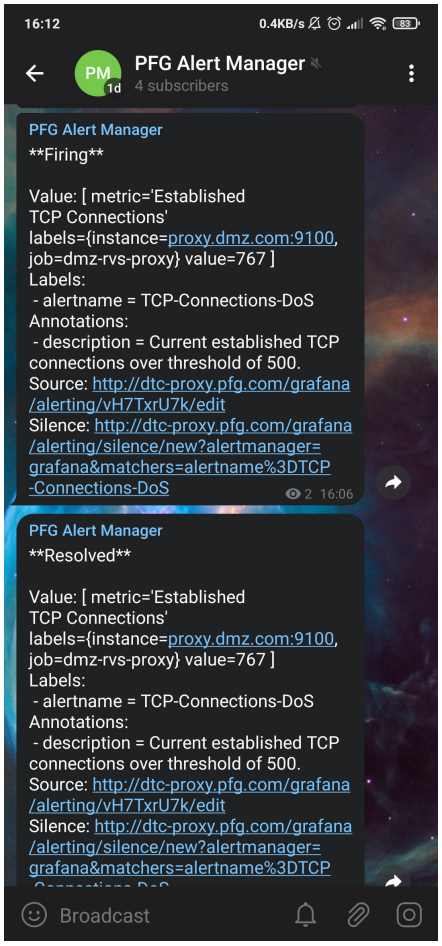


Figura 4.15: Fluxo de notificação de alertas do Grafana via Telegram, originados do ataque Slowloris. Indica-se o estabelecimento de conexões TCP acima do limiar válido e esperado.

Visto que tentativas de ataques são inevitáveis, é importante ter estabelecidas estratégias de mitigação e melhoria dos sistemas, a fim de evitar novas ocorrências similares. Tal como descreve Cloudflare (2022), umas das estratégias para este tipo de ataque é a limitação do número de conexões estabelecidas pelo mesmo usuário, sendo então o atacante bloqueado impedido de inundar

Description	Level	Rule ID
Suricata: Alert - SURICATA STREAM FIN out of window	3	86601
Suricata: Alert - SURICATA STREAM FIN out of window	3	86601
Suricata: Alert - SURICATA STREAM FIN out of window	3	86601
Suricata: Alert - SURICATA STREAM FIN out of window	3	86601
Suricata: Alert - GPL ICMP_INFO PING *NIX	3	86601

Figura 4.16: Apresentação dos alertas gerados pelo Suricata e encaminhados ao Wazuh referente ao ataque Slowloris.

o servidor *web* ou *proxy*. Após a definição do limite de 50 conexões por usuário, nas opções avançadas da regra HTTPS do *firewall c1-core-fw*, este ataque é mitigado, como pode ser visto na Figura 4.17.

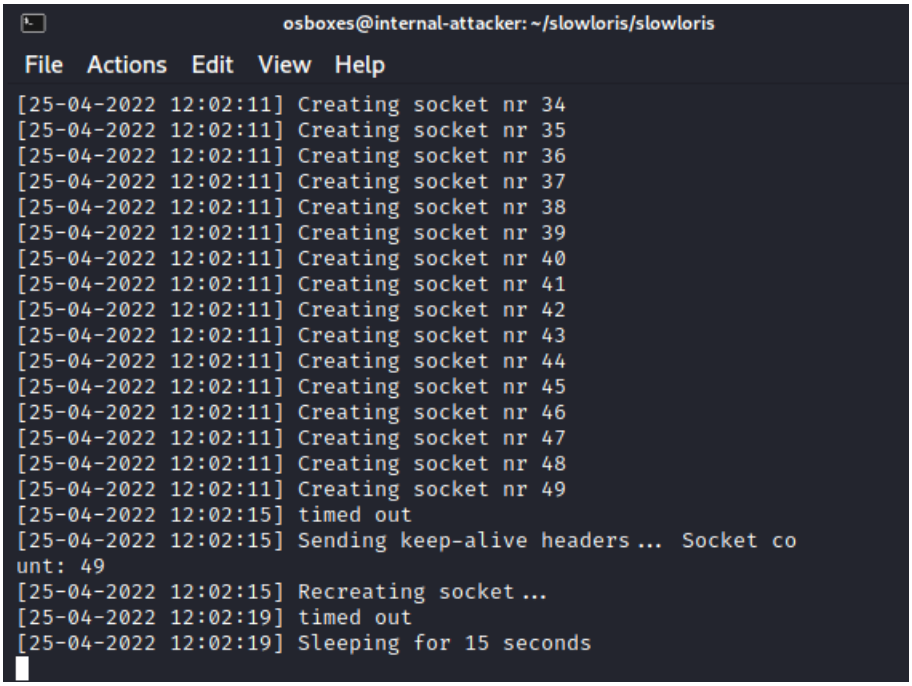


Figura 4.17: Ataque Slowloris sendo mitigado devido à definição do número máximo de possíveis conexões estabelecidas para um mesmo *host*. Ajuste definido na regra de permissão HTTPS para fora da rede LAN.

4.2.3 Ataque de penetração através da ferramenta Hydra

Um dos principais objetivos dos atacantes na área de segurança é a penetração de sistemas, que pode ser realizada de inúmeras formas distintas. Uma dos protocolos mais relevantes nesta exploração é o SSH, utilizado principalmente para a conexão em servidores remotos. Foi utilizada da ferramenta Hydra, também disponível no sistema Kali Linux, para explorar ataques de força bruta sobre as credenciais do servidor SSH do proxy reverso da zona DMZ. Como já configurado anteriormente, as regras do *firewall c1-core-fw* não permitem a tentativa de conexão SSH do atacante interno, como pode ser verificado na decodificação dos *logs* do *firewall* realizado pelo Wazuh na Figura 4.18. Dado que o *firewall* foi capaz de bloquear tal fluxo, não é gerado automaticamente qualquer forma de alerta pelo Wazuh, pois este evento não apresenta prováveis perigos à organização. Contudo, está definida uma regra que realizará o alerta caso sejam realizadas rápidas e sucessivas tentativas de conexões, que pode também ser alterada de acordo com os requisitos da organização. Embora o *log* descrito não gere um alerta automático, o Wazuh ainda é capaz de aprimorar tais informações manualmente, facilitando a análise do evento. Como já descrito, quaisquer regras implementadas pelo Wazuh também podem facilmente ser alteradas pela própria interface gráfica do Kibana.

```
**Phase 2: Completed decoding.
        name: 'json'
        agent.id: '011'
        agent.name: 'c1-core-fw'
        data.action: 'block'
        data.dstip: '10.0.3.13'
        data.dstport: '22'
        data.id: '1650131152'
        data.length: '0'
        data.protocol: 'tcp'
        data.srcip: '10.1.2.50'
        data.srcport: '39414'
        decoder.name: 'pf'
        full_log: 'Apr 25 22:04:06 c1-core-fw filterlog[14911]: 103,,,1650131152,
em1.102,match,block,in,4,0x0,,64,18864,0,DF,6,tcp,60,10.1.2.50,10.0.3.13,39414,22
,0,S,2800748016,,64240,,mss;sackOK;TS;nop;wscale'
        id: '1650924247.5054255'
        location: '/var/log/filter.log'
        manager.name: 'dtc-server-2'
        predecoder.hostname: 'c1-core-fw'
        predecoder.program_name: 'filterlog'
        predecoder.timestamp: 'Apr 25 22:04:06'
        rule.description: 'pfSense firewall drop event.'
        rule.firedtimes: '3'
        rule.gpg13: '['4.12']'
        rule.groups: '['pfsense', 'firewall_block']'
        rule.hipaa: '['164.312.a.1']'
        rule.id: '87701'
        rule.level: '5'
        rule.mail: 'false'
        rule.nist_800_53: '['SC.7']'
        rule.pci_dss: '['1.4']'
        rule.tsc: '['CC6.7', 'CC6.8']'
        timestamp: '2022-04-25T18:04:07.219-0400'
```

Figura 4.18: Apresentação do *log* do Suricata, implementado *firewall* c1-core-fw, decodificado pelo Wazuh. Este descreve o evento de bloqueio da tentativa de conexão SSH pelo atacante interno.

Contudo, por motivos de análise dos resultados de monitoramento, essa regra foi desativada

momentaneamente. Assim como apresentado na Figura 4.19, foram realizadas mais de 100 tentativas de conexão SSH com o servidor *dmz-rvs-proxy-1* em menos de 5 minutos e, como esperado, foi obtido *login* e senha de conexão. A Figura 4.20 apresenta a agregação dos *logs firewall c1-core-fw* e do servidor *dmz-rvs-proxy-1* agregados pelo Wazuh, que apresenta total visibilidade sobre as tentativas de conexão SSH realizadas pelo atacante interno. Nesta imagem são visualizadas, em sequência, alertas de escaneamneto da porta SSH, diversas tentativas falhas de conexão SSH (agregadas em apenas um alerta pelo Wazuh) e por, fim, o sucesso de autenticação que afirma que o atacante conseguiu acesso ao servidor de destino.

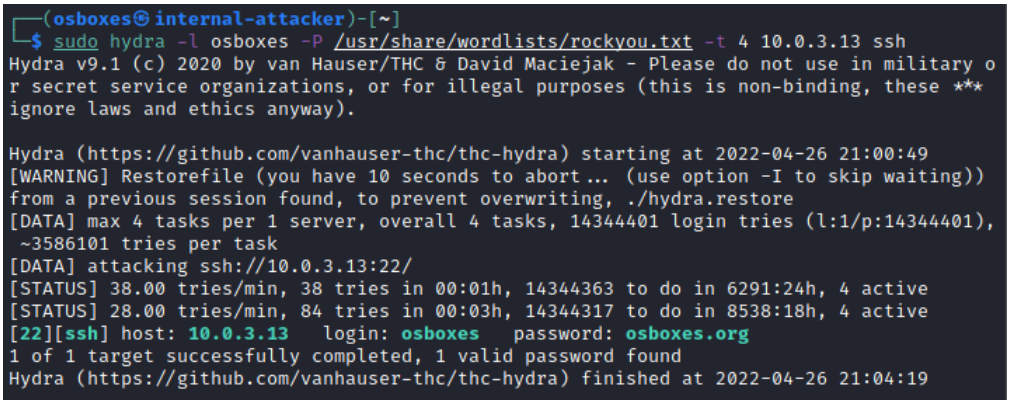


Figura 4.19: Teste do resultado do ataque de penetração SSH por força bruta, realizado pelo atacante interno com a ferramenta Hydra, sobre o servidor *dmz-rvs-proxy-1*. Reitera-se que foram desabilitadas regras de firewll momentaneamente para análise de cenário.

dmz-rvs-proxy-1	T1078	T1021	Defense Evasion, Initial Access, Persistence, Privilege Escalation, Lateral Movement	sshd: authentication success.	3	5715
dmz-rvs-proxy-1	T1110		Credential Access	sshd: authentication failed.	5	5716
dmz-rvs-proxy-1	T1110		Credential Access	sshd: authentication failed.	5	5716
dmz-rvs-proxy-1	T1110		Credential Access	sshd: authentication failed.	5	5716
dmz-rvs-proxy-1	T1110		Credential Access	sshd: authentication failed.	5	5716
c1-core-fw				Suricata: Alert - ET SCAN Potential SSH Scan OUTBOUND	3	86601
c1-core-fw				Suricata: Alert - ET SCAN Potential SSH Scan OUTBOUND	3	86601

Figura 4.20: Apresentação do monitoramento dos eventos de tentativa de conexão SSH por força bruta. Resultados dos *logs* do *firewall c1-core-fw* e do servidor *dmz-rvs-proxy-1* coletados e aprimorados pelo Wazuh.

Como já foi validado, a melhor prática de segurança para impedir ataques mais agressivos, como ataques de força bruta, é a definição de regras claras e diretas no *firewall*, impedindo sequer a tentativa de conexão ao servidor. Além disso, devem ser empregadas restrições na própria aplicação, como neste caso do servidor SSH, deve ser configurado o bloqueio de autenticação por senha, número máximo de tentativas de conexão antes de um *timeout*, lista de endereços IP permitidos, entre outros.

4.2.4 Ataque man-in-the-middle (MITM) através da técnica de ARP Cache Poisoning

A técnica de ataque ARP Cache Poisoning, ou ARP Spoofing, consiste no transmissão de pacotes ARP Reply ao *gateway* da rede LAN e à vítima do ataque com a intenção de provocar uma mudança na associação entre o seu endereço IP e MAC na tabela ARP. Após esta alteração na tabela, o atacante é compreendido como sendo o *gateway* da rede para a vítima e como sendo a vítima para o *gateway* da rede. Dessa forma, todo o tráfego transmitido pelo cliente é enviado inicialmente para o atacante, que o redireciona para o *gateway* original após executada suas atividades, sendo o fluxo contrário também válido. Dada a priorização da performance em detrimento da segurança, o protocolo ARP disponibiliza o acesso a tal vulnerabilidade de forma extremamente simples, dado que as respostas ARP (ARP Replies) podem ser enviadas sem nem mesmo a exigência de uma requisição ARP (ARP Request) preliminar. Portanto, é importante enfatizar que essa vulnerabilidade permite o controle total sobre os pacotes gerados e transmitidos pela vítima, apresentando assim uma grande falha de segurança. Este ataque caracteriza um exemplo da classe de técnicas *man-in-the-middle* (RADWARE, 2022).

Para esta análise prática foi utilizado o software Ettercap, disponível por padrão na distribuição do Kali Linux (ETTERCAP, [s.d.]). Como já mencionado, a execução deste ataque é simples, bastando identificar a vítima e o *gateway* como alvos para transmissão dos pacotes ARP. Após a inicialização do ataque, a Figura 4.21 ilustra o tráfego ICMP iniciado pela vítima (PC3) para o endereço IP do servidor *proxy* DMZ e, estando a vítima conectada na mesma VLAN que o atacante, este atua como *man-in-the-middle* desta conexão. À direita, são evidentes os pacotes de *echo reply* sendo recebidos pela vítima, enquanto, à esquerda, é visível o atacante interceptando tanto os pacotes *echo request* como *echo reply*. De acordo com a Figura 4.22, são apresentadas as tabelas ARP da vítima durante e depois do ataque, em que se observa a igualdade entre o endereço MAC do *gateway* original com o *gateway* falsificado pelo atacante. Da maneira com que esse *switch* e VLAN foram configurados até este momento, todo tráfego proveniente do usuário PC3, ou qualquer outro dispositivo que venha estar conectado a esta VLAN, está passível de interceptação pelo atacante.

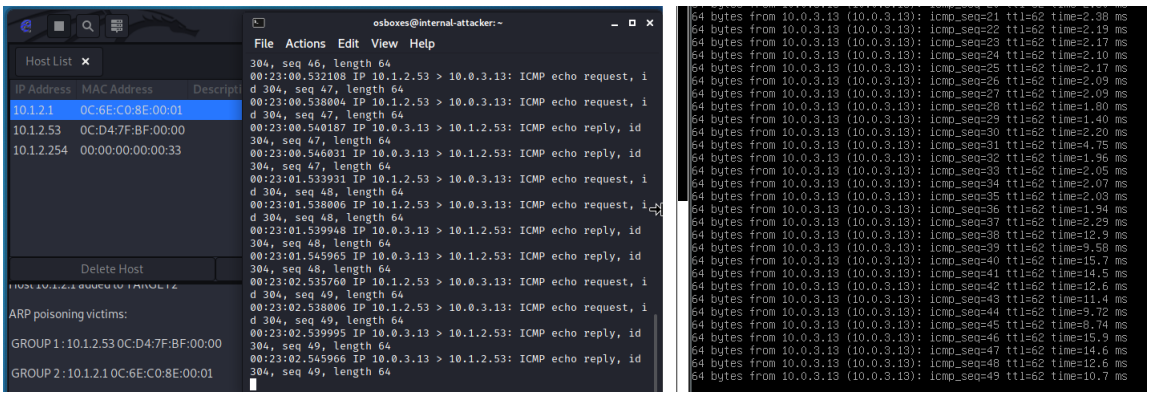


Figura 4.21: Apresentação à direita do tráfego ICMP iniciado pelo PC3 e à esquerda a interceptação deste tráfego pelo atacante interno através do ataque ARP Cache Poisoning.

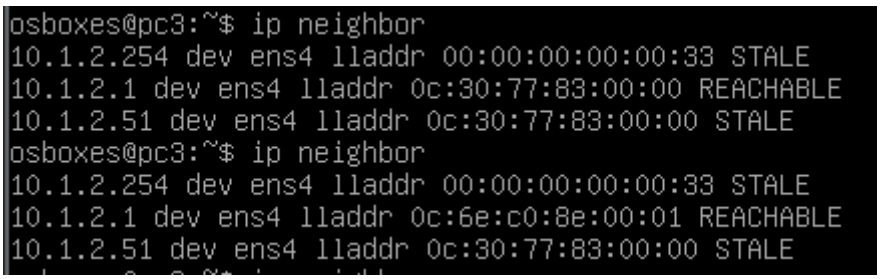


Figura 4.22: Apresentação da tabela ARP da vítima (PC3) durante e depois do ataque de ARP Cache Poisoning.

O ARP Poisoning é considerado um ataque de baixo nível, o que torna praticamente impossível sua detecção pelas ferramentas disponíveis na topologia, ainda mais em tempo real. Portanto, essa análise apresenta um resultado importante de que, mesmo com as melhores ferramentas, não é possível monitorar todos os possíveis casos de incidentes de segurança, principalmente para uma sub-rede aberta para visitantes. Contudo, é possível e importante se prevenir dos ataques já explorados e conhecidos na área de segurança. Em redes convencionais, as principais soluções implementadas são a implementação de uma tabela ARP estática e a filtração de pacotes ARP suspeitos através da técnica DAI (*Dynamic Arp Inspection*), por exemplo (RADWARE, 2022). Contudo, foi proposta a mitigação deste ataque através da definição de uma regra de ACL, pela controladora SDN, responsável pelo bloqueio de servidores DNS e também pelo encaminhamento forçado de todo o tráfego dos usuários finais pelo enlace de conexão com o *gateway*, (*firewall c1-core-fw*). Essa proposta é válida para a sub-rede de visitantes, dado que estes não necessitam manter uma conexão entre si. A Figura 4.23 apresenta a implementação real da regra de ACL descrita e a Figura 4.24 apresenta a definição em quais portas do *switch* serão tratadas essas regras, valendo ressaltar a definição da interface "trusted-port"como interface segura definida na regra da Figura 4.23. Através destas implementações de controle de acesso a cada porta, o software *Ettercap*, ou qualquer outro *sniffer*, não consegue mais descobrir os demais *hosts* da sub-rede, dado que todos os pacotes são forçados para o *firewall c1-core-fw*, como pode ser visto na Figura 4.25 pela ausência do PC3 na lista de endereços IP descobertas pelo atacante interno.

```
acls:
   access-port-protect:
       - rule: # Drop DHCP Servers
           dl_type: 0x800 # IPv4
           nw_proto: 17 # UDP
           udp_src: 67
           udp_dst: 68
           actions:
               allow: 0 # Drop
       - rule: # Forward all Traffic to Firewall LAN
           actions:
               output:
                   port: "trusted-port"
```

Figura 4.23: Definição das regras de ACL pela controladora Faucet para bloqueio de servidores DHCP e redirecionamento de todo o tráfego para o *gateway* seguro de nome "trusted-port".

```
c1-acc-sw-3:
    dp_id: 0x12
    hardware: "Open vSwitch"
    interfaces:
        1:
            name: "trusted-port"
            description: "stack link to c1-dst-sw-3"
            stack:
               dp: c1-dst-sw-3
               port: 2
        2:
            description: "iface to PC3"
            native_vlan: c1-int-bldb-users
            acls_in: [access-port-protect]
        3:
            description: "iface to INTERNAL_ATTACKER"
            native_vlan: c1-int-bldb-users
            acls_in: [access-port-protect]
```

Figura 4.24: Definição das regras de ACL nas portas do *switch* conectadas aos *hosts* PC3 e ao atacante interno. Através dessa proteção o atacante interno é impossibilitado de se conectar com o PC3.

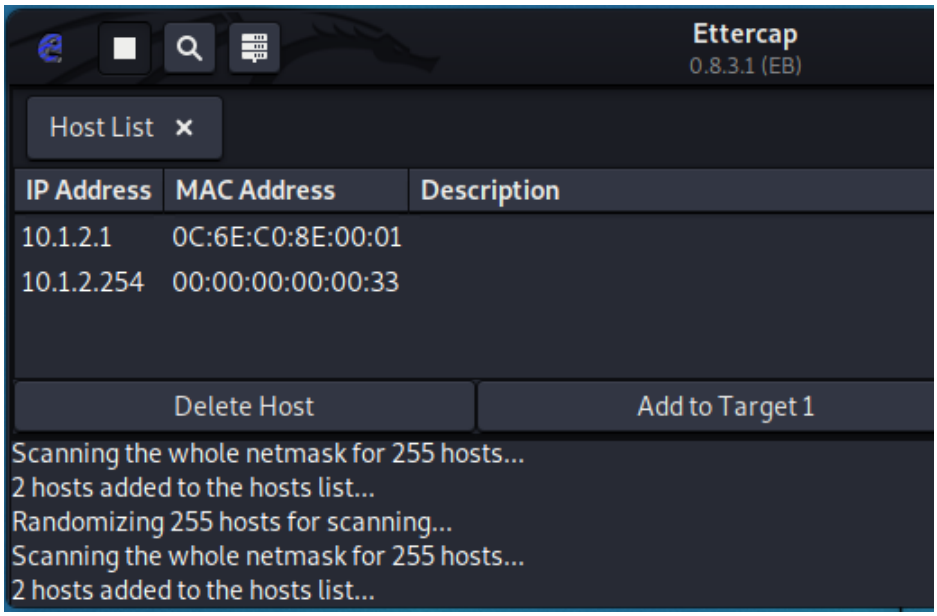


Figura 4.25: Apresentação do resultado do *software* Ettercap após implementação das regras de ACL para bloqueio do *sniffing* e *ARP poisoning*. Verifica-se a ausência do PC3 da lista de endereços IP descobertos na sub-rede.

4.2.5 Resultados gerais

Ao final das análises realizadas, utilizando ferramentas comuns aos profissionais de segurança da informação, verifica-se a efetividade tanto do sistema de gerenciamento proposto como de algumas boas práticas e técnicas de segurança implementadas. Para complementar, disponibiliza-se um resumo geral e agregado, disponibilizado pelo Wazuh no *dashboard* do Kibana, dos resultados obtidos durante as análises aqui desenvolvidas. Essa observação pode ser visualizada na Figura 4.26.

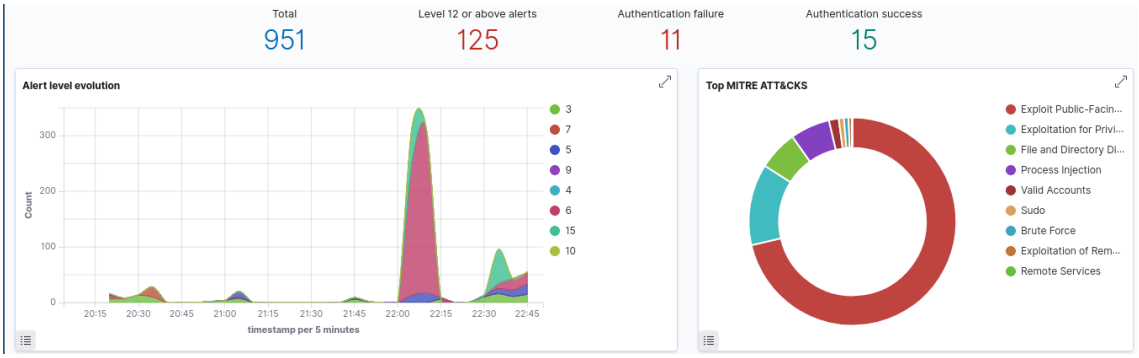


Figura 4.26: Apresentação de um trecho dos resultados coletados e agregados pelo Wazuh após todas as análises realizadas nesta seção. Visualização pelos painéis do Kibana.

Capítulo 5

Conclusões

A arquitetura implementada nesta proposta agrega sistemas, ferramentas e aplicações capazes de simular o ambiente de uma organização real em um laboratório virtual. Através das configurações de redundância, foi implementado um ambiente de alta disponibilidade que mantém o funcionamento das aplicações servidas e o ininterrupto tráfego dos usuários e clientes mesmo em situações adversas, como a falha nos dispositivos. Além do foco no desenvolvimento de uma infraestrutura robusta, também foram empregados grandes esforços para o monitoramento de métricas de desempenho, eventos e informações de segurança com soluções *open-source* que podem ser utilizadas por qualquer profissional ou organização.

O software de emulação empregado tornou possível a orquestração de múltiplos sistemas operacionais em conjunto em um único ambiente virtualizado, habilitando a implementação prática e simultânea da topologia mesmo com a grande distância física entre os autores. Além disso, a disponibilização de soluções *open-source* para todos os sistemas e ferramentas utilizadas foi indubitavelmente uma das características principais que permitiu o desenvolvimento deste estudo.

Apesar de não ter sido o foco principal deste estudo, as decisões e definições descritas sobre endereçamento IP, protocolos de roteamento, tradução NAT, configuração de VLANs, entre outros, apresentam uma lista de proposições que podem ser utilizadas para estudo técnico e comparação. Complementando, a implementação prática de uma arquitetura SDN foi apresentada para, além de facilitar a configuração de redundância, VLANs e regras de ACL, disponibilizar uma revisão acerca do estado atual desta tecnologia, cujos resultados foram excepcionais.

Por fim, através das análises dos ataques realizados, foi possível comprovar, além da perfeita integração entre as ferramentas utilizadas como solução de monitoramento de eventos e informações de segurança em tempo real, resultados práticos que indicam e propõem ajustes sobre as configurações implementadas e as boas práticas a serem seguidas. Conclui-se que o estudo proposto foi capaz de cumprir com seus objetivos e também oferecer uma notável fonte de estudo para profissionais da área, especialmente para os principiantes.

5.1 Trabalhos Futuros

Devido ao extenso escopo dessa proposta e ao tempo limitado para desenvolvimento, não foi possível incorporar mais ferramentas para uso e mais cenários de teste para as implementações realizadas. Portanto, sugere-se a evolução deste estudo baseado na execução de ataques mais complexos e sofisticados, aprimorando também todo o ecossistema para adequação a esta nova situação.

Através das melhorias do que já foi descrito e implementado, indica-se, por exemplo, a operação do atacante externo, já definido mas não utilizado. Indica-se também, a análise aprofundada dos protocolos de roteamento e formas de explorar as vulnerabilidades neles contidas, como a injeção de rotas. Além disso, como já mencionado no decorrer deste relatório, é um fato de que a maioria dos incidentes de segurança são realizados por agentes internos à organização, logo implementações como filtros de domínio DNS ou filtros de conteúdos web, por exemplo, são capazes de reduzir a taxa de casos de *phishing* e, consequentemente, acessos indevidos. Um outro ponto é a implementação de aplicações mais complexas, como bancos de dados que armazenam informações sensíveis e pessoais, tornando-se um ponto de fragilidade para organização e seus clientes. Por fim, indica-se a exploração das demais funcionalidades do Wazuh e dos demais sistemas de gerenciamento, garantindo maior visibilidade de eventos e agregação rápida de informações valiosas.

Planeja-se, portanto, o crescimento deste estudo como uma forma de agregar, cada vez mais, conhecimento acessível e completo em relação às áreas de administração de redes e segurança da informação.

REFERÊNCIAS BIBLIOGRÁFICAS

AHMAD, I.; NAMAL, S.; YLIANTTILA, M.; GURTOV, A. Security in software defined networks: A survey. **IEEE Communications Surveys & Tutorials**, v. 17, p. 2317–2340, 01 2015.

AWATI, R. **Network operations center (NOC)**. 2021. Disponível em: <https://www.techtarget.com/searchnetworking/definition/network-operations-center>. Acesso em: 23 out. 2021.

BARANES, H. **5 Reasons Why You Need an Event Management Tool**. 2019. Disponível em: <https://www.sysaid.com/blog/itsm/5-reasons-why-you-need-an-event-management-tool>. Acesso em: 28 out. 2021.

BIDOU, R. Security operation center concepts & implementation. 01 2005.

BREZULA, R. **Enterprise Network on GNS3**. 2017. Disponível em: <https://brezular.com/2017/09/07/enterprise-network-on-gns3-part-1-introduction/>. Acesso em: 24 out. 2021.

CIRT.NET. **Nikto2**. c2022. Disponível em: <https://cirt.net/Nikto2>. Acesso em: 26 abr. 2022.

CISCO. **Enterprise Campus 3.0 Architecture: Overview and Framework**. 2008. Disponível em: <https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Campus/campover.html>. Acesso em: 02 mar. 2022.

_____. **Estudos de caso de BGP**. 2008. Disponível em: <https://www.cisco.com/c/pt_br/support/docs/ip/border-gateway-protocol-bgp/26634-bgp-toc.html>. Acesso em: 18 abr. 2022.

_____. **Guia de projeto de OSPF**. 2013. Disponível em: <https://www.cisco.com/c/pt_br/support/docs/ip/open-shortest-path-first-ospf/7039-1.html>. Acesso em: 18 abr. 2022.

CISCO PRESS. **Overview of Security Operations Center Technologies**. 2015. Disponível em: <https://www.ciscopress.com/articles/article.asp?p=2455014>. Acesso em: 28 out. 2021.

CLOUDFLARE. **Ataque DDoS Slowris**. 2022. Disponível em: <https://www.cloudflare.com/pt-br/learning/ddos/ddos-attack-tools/slowloris/>. Acesso em: 25 mar. 2022.

CROWLEY, C.; PESCATORE, J. Common and best practices for security operations centers: Results of the 2019 soc survey. **Information Security Reading Room, SANS Institute**, 2019.

CóZAR, J. P. **Implementación de Wazuh en una organización pública**. Dissertação (Mestrado) — Universitat Oberta de Catalunya (UOC), Junho 2020.

DEBIAN. **Securing Debian Manual**. 2022. Disponível em: <https://www.debian.org/doc/manuals/securing-debian-manual/sec-bind.en.html>. Acesso em: 13 mar. 2022.

ETTERCAP. **Ettercap**. [s.d.]. Disponível em: <https://www.ettercap-project.org/>. Acesso em: 26 abr. 2022.

FAUCET. **Architecture**. 2021. Disponível em: <https://docs.faucet.nz/en/latest/architecture.html>. Acesso em: 15 mar. 2022.

_____. **Configuration**. 2021. Disponível em: <https://docs.faucet.nz/en/latest/configuration.html>. Acesso em: 20 abr. 2022.

_____. **Introduction to Faucet**. 2021. Disponível em: <https://docs.faucet.nz/en/latest/intro.html>. Acesso em: 16 abr. 2022.

FERGUSON, A. D.; GRIBBLE, S.; HONG, C.-Y. et al. Orion: Google's Software-Defined networking control plane. In: **18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)**. USENIX Association, 2021. p. 83–98. ISBN 978-1-939133-21-2. Disponível em: <https://www.usenix.org/conference/nsdi21/presentation/ferguson>.

FORCEPOINT. **What is the CIA Triad?** s.d. Disponível em: <https://www.forcepoint.com/pt-br/cyber-edu/cia-triad>. Acesso em: 28 out. 2021.

GEORGIADOU, A.; MOUZAKITIS, S.; ASKOUNIS, D. Assessing mitre att&ck risk using a cyber-security culture framework. **Sensors**, v. 21, n. 9, 2021. ISSN 1424-8220.

GOVERNO FEDERAL. **Proteção de Dados - LGPD**. 2020. Disponível em: <https://www.gov.br/defesa/pt-br/acesso-a-informacao/lei-geral-de-protecao-de-dados-pessoais-lgpd>. Acesso em: 26 out. 2021.

HAE, T. S.; THONG, L. K.; MATTHEW, S. N.; HOW, T. C. Smart network and security operations centre. **DSTA Horizons**, 2016.

HARRINGTON, D. **Data Security: Importance, Types, and Solutions**. 2021. Disponível em: <https://www.varonis.com/blog/data-security/>. Acesso em: 28 out. 2021.

HONG, D. K.; MA, Y.; BANERJEE, S.; MAO, Z. M. Incremental deployment of sdn in hybrid enterprise and isp networks. In: **Proceedings of the Symposium on SDN Research**. New York, NY, USA: Association for Computing Machinery, 2016. (SOSR '16). ISBN 9781450342117. Disponível em: <https://doi.org/10.1145/2890955.2890959>.

HUMAYUN, M.; NIAZI, M.; JHANJHI, N. et al. Cyber security threats and vulnerabilities: A systematic mapping study. **Arabian Journal for Science and Engineering**, v. 45, n. 4, 2020.

ISC. **BIND 9**. c2022. Disponível em: <https://www.isc.org/bind/>. Acesso em: 25 abr. 2022.

JING, X.; YAN, Z.; PEDRYCZ, W. Security data collection and data analytics in the internet: A survey. **IEEE Communications Surveys & Tutorials**, v. 21, n. 1, p. 586–618, 2019.

KHAN, N.; BROHI, S.; ZAMAN, N. Ten deadly cyber security threats amid covid-19 pandemic. 05 2020.

KIM, S.; YOON, S.; NARANTUYA, J.; LIM, H. Secure collecting, optimizing, and deploying of firewall rules in software-defined networks. **IEEE Access**, v. 8, p. 15166–15177, 2020.

MAGANE ENGINE. **Fault and Event Management**. 2021. Disponível em: <https://www.manageengine.com/network-monitoring/event-and-fault-management.html>. Acesso em: 28 out. 2021.

MANAGE ENGINE. **Network monitor's Event Log monitoring**. 2021. Disponível em: <https://www.manageengine.com/network-monitoring/Eventlog_Tutorial_Part_II.html>. Acesso em: 28 out. 2021.

MCAFEE. **What Is a Security Operations Center (SOC)?** 2021. Disponível em: <https://www.mcafee.com/enterprise/en-us/security-awareness/operations/what-is-soc.html>. Acesso em: 24 out. 2021.

MCKEOWN, N.; ANDERSON, T.; BALAKRISHNAN, H. et al. **OpenFlow: Enabling Innovation in Campus Networks**. 2008. Disponível em: <http://ccr.sigcomm.org/online/files/p69-v38n2n-mckeown.pdf>. Acesso em: 24 out. 2021.

MITRE. **MITRE ATT&CK**. 2021. Disponível em: <https://attack.mitre.org>. Acesso em: 24 out. 2021.

NELSON, R. **Tuning NGINX for Performance**. 2014. Disponível em: <https://www.nginx.com/blog/tuning-nginx/>. Acesso em: 25 abr. 2022.

NETGATE. **High Availability**. 2021. Disponível em: <https://docs.netgate.com/pfsense/en/latest/highavailability/index.html>. Acesso em: 20 mar. 2022.

_____. **Rule Methodology**. 2021. Disponível em: <https://docs.netgate.com/pfsense/en/latest/firewall/rule-methodology.html>. Acesso em: 20 mar. 2022.

NGINX. **Advanced Load Balancer, Web Server Reverse Proxy - NGINX**. c2022. Disponível em: <https://www.nginx.com/>. Acesso em: 25 abr. 2022.

_____. **SSL Termination for TCP Upstream Servers**. c2022. Disponível em: <https://docs.nginx.com/nginx/admin-guide/security-controls/terminating-ssl-tcp>. Acesso em: 25 abr. 2022.

NGUYEN, V.-G.; KIM, Y.-H. Sdn-based enterprise and campus networks: A case of vlan management. **Journal of Information Processing Systems**, v. 12, n. 3, p. 511–524, 2016.

NMAP. **Nmap: the Network Mapper**. [s.d.]. Disponível em: <https://nmap.org/>. Acesso em: 26 abr. 2022.

OISF. **What is Suricata**. c2019. Disponível em: <https://suricata.readthedocs.io/en/suricata-6.0.5/what-is-suricata.html>. Acesso em: 16 abr. 2022.

OPEN VSWITCH. **Open vSwitch**. 2016. Disponível em: <https://www.openvswitch.org/>. Acesso em: 29 out. 2021.

PESCATORE, J.; FILKINS, B. Closing the critical skills gap for modern and effective security operations centers (socs). **SANS Institute**, 2020.

PFSENSE. **pfSense - World's Most Trusted Open Source Firewall**. 2021. Disponível em: <https://www.pfsense.org/>. Acesso em: 29 out. 2021.

PROMETHEUS. **Exporters and Integrations**. c2022. Disponível em: <https://prometheus.io/docs/introduction/overview/>. Acesso em: 10 abr. 2022.

_____. **Overview: What is Prometheus?** c2022. Disponível em: <https://prometheus.io/docs/introduction/overview/>. Acesso em: 13 mar. 2022.

RADWARE. **ARP Poisoning**. 2022. Disponível em: <https://www.radware.com/security/ddos-knowledge-center/ddospedia/arp-poisoning/>. Acesso em: 25 mar. 2022.

REBACK, G. **O que é SIEM e quais suas principais funcionalidades?** 2020. Disponível em: <https://seginfo.com.br/2020/09/03/o-que-e-siem-e-quais-suas-principais-funcionalidades/>. Acesso em: 28 out. 2021.

ROSENCRANCE, L. **Security information and event management (SIEM)**. 2020. Disponível em: <https://searchsecurity.techtarget.com/definition/security-information-and-event-management-SIEM>. Acesso em: 28 out. 2021.

ROSS, E.; LIANG, H.; SHARKEY, K. **DHCP (Dynamic Host Configuration Protocol) Basics**. 2021. Disponível em: <https://docs.microsoft.com/en-us/windows-server/troubleshoot/dynamic-host-configuration-protocol-basics>. Acesso em: 17 abr. 2022.

SCARFONE, K.; HOFFMAN, P. **Guidelines on Firewalls and Firewall Policy**. Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2009. Disponível em: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-41r1.pdf>.

SHIVANG. **What is Grafana? Why Use It? Everything You Should Know About It**. 2022. Disponível em: <https://www.scaleyourapp.com/what-is-grafana-why-use-it-everything-you-should-know-about-it/>. Acesso em: 19 mar. 2022.

VAZãO, A.; SANTOS, L.; PIEDADE, M.; RABADãO, C. Siem open source solutions: A comparative study. In: **14th Iberian Conference on Information Systems and Technologies (CISTI)**. Coimbra, Portugal: IEEE, 2019.

VYOS. **VyOS - Open Source Router and Firewall Platform**. 2021. Disponível em: <https://vyos.io/>. Acesso em: 29 out. 2021.

WANG, B.; ZHENG, Y.; LOU, W.; HOU, Y. T. Ddos attack protection in the era of cloud computing and software-defined networking. **Computer Networks**, v. 81, p. 308–319, 2015. ISSN 1389-1286. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128615000742>. Acesso em: 9 mar. 2021.

WAZUH. **CIS-CAT integration**. c2022. Disponível em: <https://documentation.wazuh.com/current/user-manual/capabilities/policy-monitoring/ciscat/ciscat.html?highlight=benchmarks>. Acesso em: 24 abr. 2022.

_____. **Components**. c2022. Disponível em: <https://documentation.wazuh.com/current/getting-started/components/index.html>. Acesso em: 24 abr. 2022.

_____. **The Open Source Security Platform**. c2022. Disponível em: <https://wazuh.com/>. Acesso em: 17 mar. 2022.

WIDUP, S.; PINTO, A.; HYLENDER, D. et al. **2021 Verizon Data Breach Investigations Report**. 2021. Disponível em: <https://www.verizon.com/business/resources/reports/2021/2021-data-breach-investigations-report.pdf>. Acesso em: 9 mar. 2021.

YALTIRAKLI, G. Slowloris. **github.com**, 2015. Disponível em: <https://github.com/gkbrk/slowloris>.

ŽGELA, M.; PENGA, I. Security information and event management – capabilities, challenges and event analysis in the complex it system. In: . [S.l.: s.n.], 2019. p. 259–266. ISSN 18472001.

APÊNDICES

I. DIAGRAMAS ESQUEMÁTICOS

Nesta seção, descrevem-se os dois principais estágios do desenvolvimento da topologia aqui proposta e implementada. Inicialmente, a Figura I.1 foi proposta como uma rede legado, com a definição do roteamento dinâmico OSPF entre as áreas da topologia e, além de tudo, definindo apenas alguns pontos de redundância. Contudo, já evidencia-se a proposta de ir além de uma topologia de campus comum, implementando também o *backbone* da organização que o gerencia.

Seguindo para a topologia final, uma evolução progressiva da topologia inicial, evidenciam-se características de complexidade e redundância muito mais elevadas. A principal grande mudança foi a troca completa do roteamento dinâmico OSPF por uma arquitetura SDN, completamente definida por *software*. Como o escopo da proposta foi bem extenso, dada a implementação de uma topologia corporativa completa e também o monitoramento da mesma, optou-se por realizar uma configuração de redundância que fosse suficiente para garantir um ambiente de alta disponibilidade, de modo que as ferramentas de gerência fossem utilizadas para auxiliar nesse processo. A configuração de uma rede complexa, através de definições do protocolo OpenFlow nos *switches*, se mostrou excepcionalmente performática dado o facilitado ajuste de parâmetros e dinamicidade da rede.

Os demais pontos da tomada de decisão estão descritos no decorrer deste relatório.

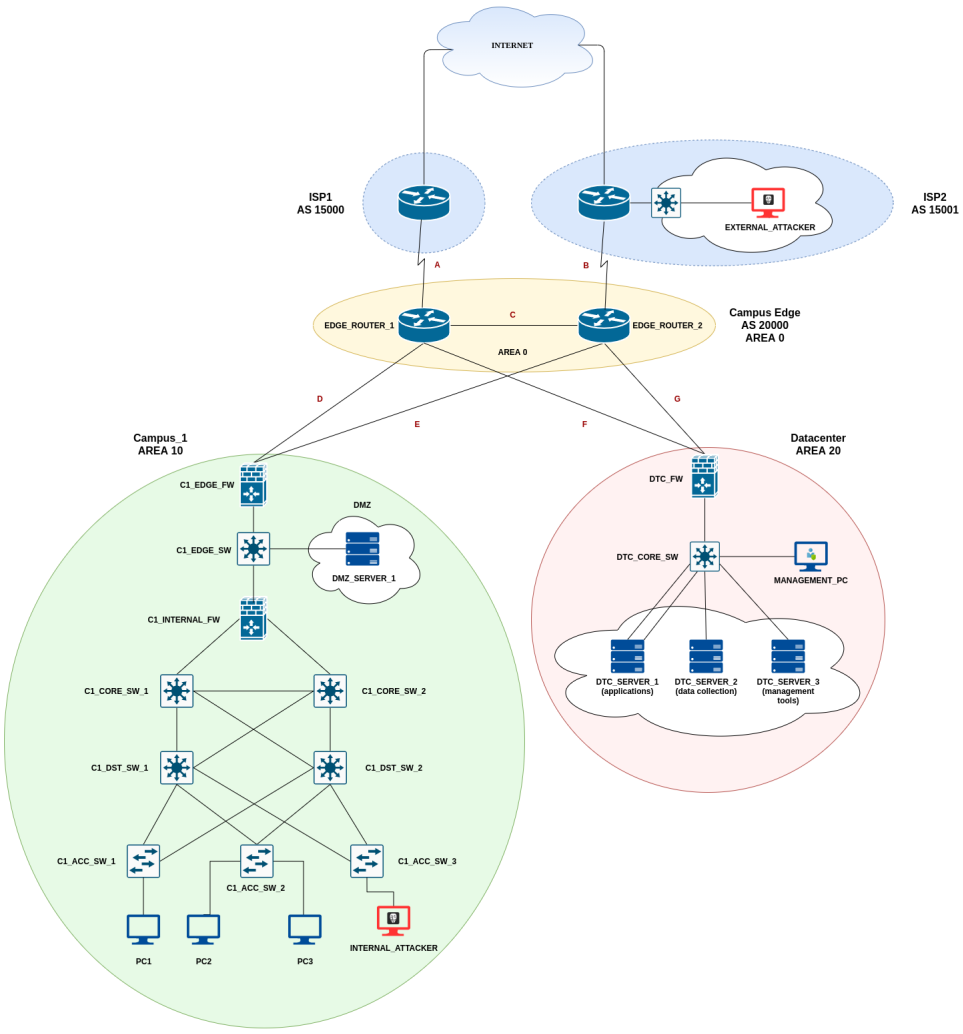


Figura I.1: Topologia inicialmente proposta para implementação.

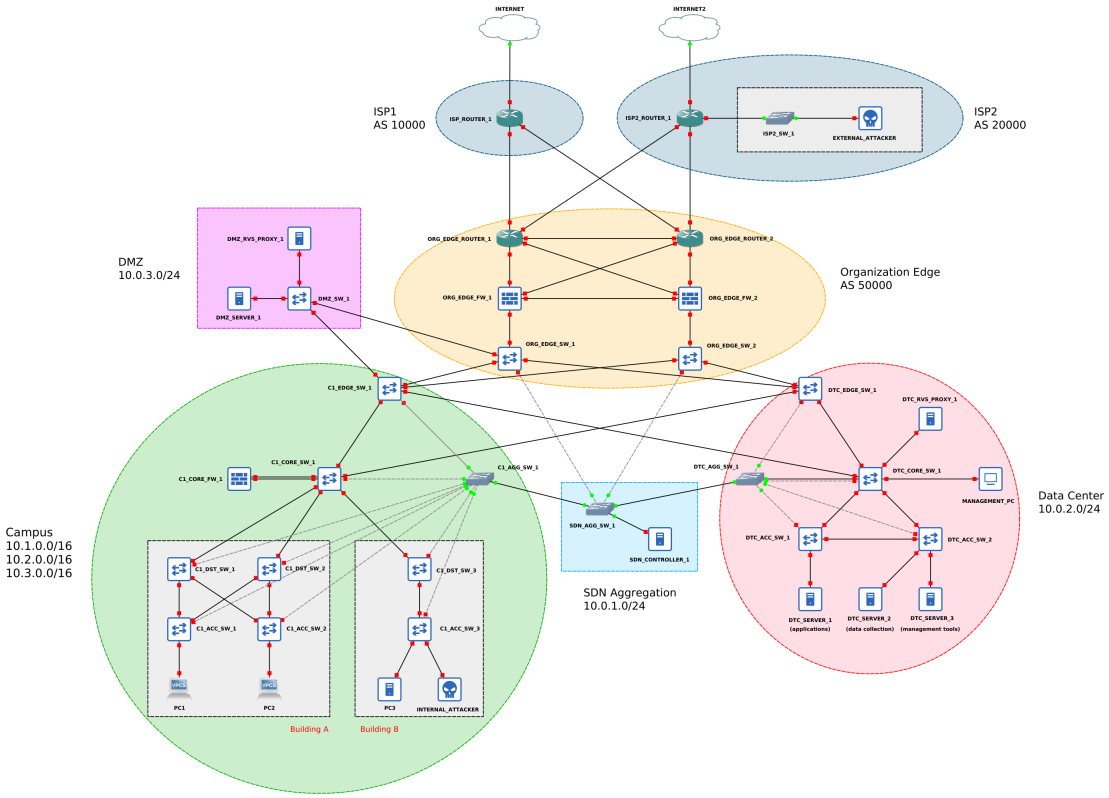


Figura I.2: Topologia final proposta para implementação.

II. ARQUIVOS DE CONFIGURAÇÃO

No desenvolvimento desta proposta foram implementados e configurados um grande número de sistemas e ferramentas. Para facilitar a visualização das decisões tomadas e, consequentemente, dos resultados apresentados, optou-se por não sobrecarregar este relatório de arquivos extensos de configuração, apenas os trechos essenciais utilizados para exemplificação de conceitos. Contudo, a disponibilização dos arquivos de configuração completos se fazem necessários para replicação e estudo deste laboratório proposto, por isso será indicado o acesso ao repositório público de códigos pelo endereço *https://github.com/pedronogs/PFG*. Os principais passos descritivos para utilização destes arquivos também estarão descritos neste repositório.