UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM REDES DE COMPUTADORES

ANTONIO RAFAEL PINHEIRO DANTAS

CYRM: CYBER RANGE PARA AUXILIAR O ENSINO DE DEFESA PARA ALUNOS DA DISCIPLINA DE SEGURANÇA DA INFORMAÇÃO.

QUIXADÁ

2022

ANTONIO RAFAEL PINHEIRO DANTAS

CYRM: CYBER RANGE PARA AUXILIAR O ENSINO DE DEFESA PARA ALUNOS DA DISCIPLINA DE SEGURANÇA DA INFORMAÇÃO.

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Redes de Computadores do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de tecnólogo em Redes de Computadores.

Orientador: Prof. Dr. Michel Sales Bonfim

QUIXADÁ

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

D21c Dantas, Antonio Rafael Pinheiro.
CyRM: cyber range para auxiliar o ensino de defesa para alunos da disciplina de segurança da informação / Antonio Rafael Pinheiro Dantas. – 2022.
66 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Redes de Computadores, Quixadá, 2022.
Orientação: Prof. Dr. Michel Sales Bonfim.

1. Containernet. 2. Cyber Range. 3. Docker. 4. Segurança computacional. 5. Treinamento. I. Título.

CDD 004.6

ANTONIO RAFAEL PINHEIRO DANTAS

CYRM: CYBER RANGE PARA AUXILIAR O ENSINO DE DEFESA PARA ALUNOS DA DISCIPLINA DE SEGURANÇA DA INFORMAÇÃO.

Projeto de Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Redes de Computadores do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de tecnólogo em Redes de Computadores.

Aprovada em: ___/___/____.

BANCA EXAMINADORA

Prof. Dr. Michel Sales Bonfim (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Marcos Dantas Ortiz
Universidade Federal do Ceará (UFC)

Prof. Me. Roberto Cabral Rabêlo Filho
Universidade Federal do Ceará (UFC)

A Deus. Aos meus pais, meu irmão, familiares e amigos que sempre me apoiaram e acreditaram no meu sonho.

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me dado forças para persistir e sempre ter dado certo. Não foi nada fácil realizar meu sonho, tive muitos contratempos, mas consegui! No começo da graduação percorria mais de 210 Km diariamente para assistir às aulas, perdendo uma hora de aula todo dia. Quando chegava em casa meia noite ainda ia estudar, pois durante o dia trabalhava, no começo da graduação.

Gostaria de agradecer de forma especial minha namorada, companheira, amiga Ana Cleivanete Pinheiro por está ao meu lado sempre, me incentivando. Aos meus pais César Antônio Pinheiro Dantas e Maria das Dores Pinheiro por sempre acreditarem em mim e me apoiarem em todos os momentos que precisei. Ao meu irmão Antônio Jardel Pinheiro Dantas, meus tios Paulo Mendes Pinheiro Dantas e Ana Glêda Pinheiro Dantas e todos os familiares e amigos.

Agradecer aos ótimos professores, por seus preciosos ensinamentos, a gestão do campus, pois sempre deu o suporte que precisei e a assistente social Kátia Borges por sempre está me acompanhando.

Agradeço imensamente ao meu orientador Dr. Michel Sales Bonfim por ter me orientado tão bem. Sabes que fico muito honrado em ser seu orientando. Foram muitas horas de dedicação, paciência, sempre sanando as minhas dúvidas o quanto antes, sempre me incentivou muito. O Sr. é nota 1000.

Agradeço também as pessoas que fiz amizade ao longo da jornada e sempre me ajudaram. Aos colegas de classe, de moradia... são tantos que não dá para citar.

Fico muito feliz com a banca examinadora dos professores Marcos Dantas e Roberto Cabral por ter feito parte e agradeço pelas contribuições e sugestões.

“Continue andando. Haverá a chance de você ser barrado por um obstáculo, talvez por algo que você nem espere. Mas siga, até porque eu nunca ouvi falar de ninguém que foi barrado enquanto estava parado.”

(Charles F. Kettering)

RESUMO

Com a modernização das aplicações, ataques cibernéticos estão ocorrendo em toda área da computação. A quantidade de ataques aumenta em toda parte mundo e a cada dia que passa novos ataques surgem e evoluem constantemente. No mercado de trabalho, as empresas necessitam de profissionais treinados e capacitados para manter a corporação segura desses ataques virtuais. Um treinamento com *Cyber Range* possui alto valor financeiro. A solução foi a criação de uma ferramenta gratuita, capaz de treinar e capacitar o aluno para que ele possa detectar e corrigir vulnerabilidades. O experimento foi realizado na Universidade Federal do Ceará, no Campus de Quixadá, com emulador de rede *Containernet* responsável por emular a topologia da rede e a usabilidade de containers *docker*, para prover os serviços e ataques definidos previamente. Os alunos responderam o questionário guiado no *moodle2*. A pesquisa resultou na satisfação dos participantes, com questões guiadas em ambiente emulado destinado a corporação, com todos os serviços e componentes finais. Por fim, foi preenchido um questionário no *Google Forms*, utilizado para avaliar a viabilidade do uso da ferramenta em outras disciplinas de segurança. Esse modelo de treinamento foi realizado pela primeira vez no Campus. Com público alvo para professores e estudantes da disciplina de Segurança da Informação, nosso resultado obtido foi a implementação do CyRM e validação da solução.

Palavras-chave: Containernet. Cyber Range. Docker. Segurança computacional. Treinamento.

ABSTRACT

With the modernization of applications, cyber-attacks are occurring in every area of computing. The amount of attacks is increasing all over the world and every day new attacks are emerging and constantly evolving. In the job market, companies need trained and skilled professionals to keep the corporation safe from these cyber attacks. A training with *Cyber Range* has a high financial value. The solution was to create a free tool, capable of training and capacitating the student to detect and correct vulnerabilities. The experiment was carried out at the Federal University of Ceará, at the Quixadá Campus, with a network emulator *Containernet* responsible for emulating the network topology and the usability of containers *docker*, to provide the services and attacks previously defined. The students answered the guided survey in *moodle2*. The survey resulted in the satisfaction of the participants, with guided questions in an emulated environment intended for the corporation, with all the final services and components. Finally, a questionnaire was filled in on *Google Forms* and used to evaluate the viability of using the tool in other security disciplines. This training model was held for the first time on campus. With a target audience of professors and students of the Information Security discipline, our result was the implementation of CyRM and validation of the solution.

Keywords: Containernet. Cyber Range. Docker. Computer Security. Training.

LISTA DE FIGURAS

Figura 1 – Estatísticas dos Incidentes Reportados ao CERT.br 14
Figura 2 – Taxonomia atualizada de um intervalo cibernético. 22
Figura 3 – Arquitetura Docker. 28
Figura 4 – Fluxograma do CyRM. 35
Figura 5 – *Dockerfile* do *containernet_cyrm*. 36
Figura 6 – Execursão da topologia CyRM. 36
Figura 7 – Topologia do CyRM. 38
Figura 8 – *Dockerfile* do ataque de *DDoS*. 40
Figura 9 – *Script* do ataque de *DDoS*. 40
Figura 10 – Comando para a criação da imagem de *DDoS*. 40
Figura 11 – Dockerfile do ataque de dicionário. 41
Figura 12 – Script do ataque de dicionário. 41
Figura 13 – Comando para a criação da imagem do ataque de dicionário. 42
Figura 14 – *Dockerfile* do *Blue Team*. 43
Figura 15 – Comando para a criação da imagem do *blue team*. 43
Figura 16 – *Dockerfile* do servidor *Web*. 44
Figura 17 – *Script* de inicialização servidor *Web*. 44
Figura 18 – *Dockerfile* dos usuários não atacantes. 45
Figura 19 – *Script* de preparação do cenário. 46
Figura 20 – Questão do roteiro. 47
Figura 21 – **Questão 5**: Houve lentidão/travamento na máquina durante o experimento? 50
Figura 22 – **Questão 15:** Encontrou algum bug? 50
Figura 23 – **Questão 16:** Como você avalia o tempo de resposta das ações do CyRM? . 50
Figura 24 – **Questão 6**: As questões estavam de fácil compreensão? 51
Figura 25 – **Questão 7**: As questões conseguiram lhe guiar em direção a resposta correta? 51
Figura 26 – **Questão 8**: Em que escala está a sua satisfação em realizar um experimento teórico e pratico em paralelo? 52
Figura 27 – **Questão 9**: Em que escala o CyRM conseguiu instruir, testar ou aperfeiçoar os conhecimentos a respeito da detecção, diagnóstico e mitigação de ataques? 52
Figura 28 – **Questão 10**: Em que escala o CyRM estimula a criatividade do aluno? . . . 53
Figura 29 – **Questão 14:** Qual o nível de aprendizagem? 53

Figura 30 – **Questão 11:** O CyRM conseguiu simular atividades realísticas em um ambiente corporativo? 53
Figura 31 – **Questão 13:** Qual o nível de dificuldade em utilizar o CyRM? 54
Figura 32 – **Questão 1:** Já fez ou está fazendo a disciplina de Segurança em Redes ou Segurança da Informação? 54
Figura 33 – **Questão 2:** Qual a satisfação com a ferramenta? 55

LISTA DE QUADROS

Quadro 1 – Análise comparativa entre trabalhos relacionados e esse trabalho. 33
Quadro 2 – Perguntas aplicadas no questionário de avaliação. 55

SUMÁRIO

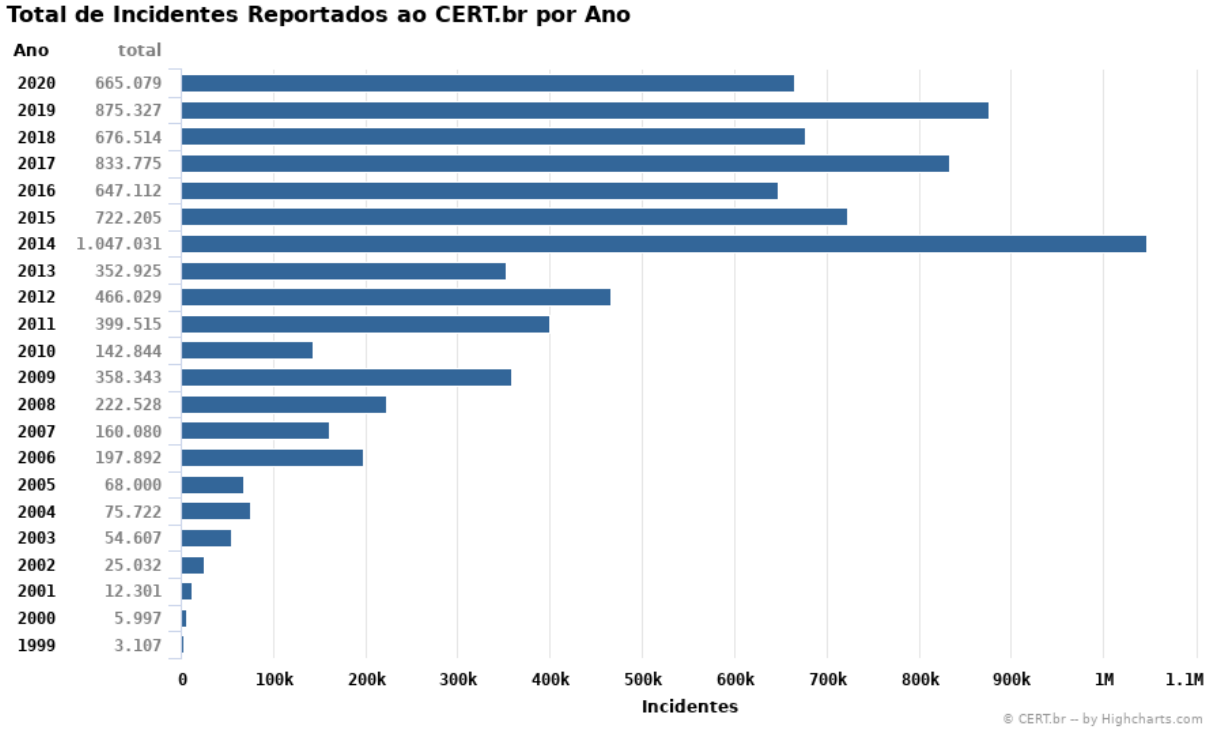
1 INTRODUÇÃO 14
1.1 Objetivos 17
1.1.1 Objetivo Geral 17
1.1.2 Objetivos Específicos 17
2 FUNDAMENTAÇÃO TEÓRICA 18
2.1 Segurança em Redes de Computadores 18
2.1.1 Tipos de ataque 18
2.1.2 Mecanismos de segurança 20
2.2 *Cyber Range* 21
2.3 Emulação com *containers Docker* 27
2.3.1 Mininet 27
2.3.2 Docker 27
2.3.3 Containernet 29
3 TRABALHOS RELACIONADOS 30
3.1 CyRIS: A Cyber Range Instantiation System for Facilitating Security Training 30
3.2 Comprehensive Cyber Arena; The Next Generation Cyber Range 31
3.3 AIT Cyber Range: Flexible Cyber Security Environment for Exercises, Training and Research 31
3.4 Gamifying ICS Security Training and Research: Design, Implementation, and Results of S3 32
3.5 Análise Comparativa 33
4 SOLUÇÃO PROPOSTA 35
4.1 Arquitetura do CyRM 35
4.2 Definição do Cenário para a Prova de Conceito 37
4.3 Implementação da Prova de Conceito do CyRM 39
4.3.1 Imagem do Ataque Distributed Denial of Service (DDoS) 39
4.3.2 Imagem do Ataque de Dicionário ou Força Bruta 41
4.3.3 Imagem do Blue Team 42
4.3.4 Imagem do Servidor Samba 43

4.3.5 *Imagem do Servidor Web* 44
4.3.6 *Imagem dos Usuários não Atacantes* 45
4.3.7 *Inicialização da Topologia* 45
4.4 **Roteiro de Estudo** 46
5 **EXPERIMENTOS** 49
5.1 **Descrição do Experimento** 49
5.2 **Avaliação dos Resultados** 49
5.3 **Discussão Final** 55
6 **CONCLUSÕES E TRABALHOS FUTUROS** 57
REFERÊNCIAS 59
APÊNDICES 60
APÊNDICE A – QUESTIONÁRIO APLICADO NO TREINAMENTO PARA OS ALUNOS 60

1 INTRODUÇÃO

De acordo com estatísticas do Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br[1]), a quantidade de ataques cibernéticos no Brasil e no mundo tem aumentado no decorrer dos anos. A Figura 1 ilustra as estatísticas de incidentes no Brasil. Somente em 2019 foram registradas 875.327 mil ataques, uma quantidade elevada e preocupante para todos os setores da sociedade.

Figura 1 – Estatísticas dos Incidentes Reportados ao CERT.br



Fonte: https://cert.br/stats/incidentes/

Temos também uma evolução e aperfeiçoamento dos métodos usados nestes ataques (LEITNER *et al.*, 2020). Portanto, há uma necessidade crescente de formar mais profissionais especializados em combater os diferentes tipos de ciberataques (operações realizadas por computadores para fins maléficos). Neste cenário, soluções de *Cyber Range* podem ser utilizadas para alavancar essa formação.

Uma plataforma de *Cyber Range* consiste em um *software* que cria um ambiente virtualizado, isolado e controlado para treinamentos voltados à segurança da informação, com exercícios práticos de ataque (*e.g., red team*) e defesa (*e.g., blue team*) em cenários realísticos (PHAM *et al.*, 2016).

[1] https://cert.br/stats/incidentes/

Existem diferentes soluções de *Cyber Range*, produzidas tanto pela academia como pela indústria. Em sua grande maioria são aplicações proprietárias com custos elevados. Dentre as aplicações desenvolvidas na academia, o CyRIS (*Cyber Range Instantiation System*) (PHAM *et al.*, 2016) provê um mecanismo para o gerenciamento de cenários de ataques cibernéticos, voltados para treinamentos de segurança. Para tanto, utiliza máquinas virtualizadas em ambiente KVM[2]. Por outro lado, o *Cyber* Arena (Karjalainen; Kokkonen, 2020) implementou uma infraestrutura para a realização do treinamento e exercícios de segurança da informação. A rede de testes é simulada em um ambiente fechado, que não possui conexão com a *Internet*, sendo controlada com sistemas operacionais e ferramentas de redes necessárias para infraestrutura virtualizada. Um problema encontrado nestas soluções é que elas consomem muito processamento e memória, o que as tornam inviáveis para implantação em ambientes acadêmicos, que geralmente possuem recursos computacionais limitados de *hardware*.

A Universidade Federal do Ceará - Campus Quixadá [3] (UFQUIXADA) é um Campus temático com seis cursos de Tecnologia da Informação e Comunicação (TIC). Neste cenário, a disciplina de Segurança da Informação tem o importante papel de capacitar o aluno na resolução de problemas que possam causar o comprometimento de serviços no local de trabalho, tais como equipamentos de defesa mal configurados. Contudo, atualmente não há um *Cyber Range* para realizar treinamento adequado nesta especialidade. Além disso, a instituição não dispõe de recursos financeiros e computacionais para contratar e implantar alguma solução de *Cyber Range* existente no mercado. Portanto, identificamos a necessidade de desenvolver uma plataforma de *Cyber Range* para a UFQUIXADA, que atendesse as necessidades e realidade da instituição, considerando principalmente, a escassez de recursos de processamento e memória.

Neste contexto, pesquisamos ferramentas que pudessem auxiliar no desenvolvimento de uma solução leve de *Cyber Range*. Identificamos duas: *Docker* (VITALINO; CASTRO, 2018) e *Containernet*[4]. O *Docker* é uma plataforma de código aberto para o gerenciamento de contêineres. O *container* é semelhante a uma máquina virtual, porém é mais rápido e leve pois compartilha o mesmo *kernel* do sistema operacional hospedeiro. Para efeito de comparação, enquanto é possível rodar dezenas de máquinas virtuais em um servidor, podemos executar centenas ou milhares de *containers* neste mesmo servidor. Usando as diferentes imagens disponíveis no repositório oficial do *Docker* (*Docker hub*), temos a flexibilidade para a criação

[2] https://www.linux-kvm.org/page/Main_Page
[3] https://www.quixada.ufc.br/
[4] https://github.com/containernet/containernet

de diferentes cenários de treinamento. Por outro lado, o *Containernet* é um emulador de rede leve, derivado do *Mininet*[5], que usa o *Docker* para a criação dos nós. Nesse caso, usamos o *Containernet* para criar uma solução de *Cyber Range* para estudantes do UFQUIXADA, pois habilita a criação de uma topologia de rede programável e leve além de usufruir da flexibilidade do *Docker*, permitindo assim a criação de diferentes cenários de defesa. Portanto, neste trabalho, consideramos a seguinte hipótese:

Hipótese: É possível desenvolver uma ferramenta de *Cyber Range* utilizando *Docker containers* e o emulador *Containernet*.

O propósito deste trabalho foi desenvolver uma ferramenta de *Cyber Range* em ambiente emulado e controlado para o treinamento de segurança de alunos da UFQUIXADA. Com esse fim, adaptamos o *Containernet* e desenvolvemos *scripts* para a criação do cenário de defesa. Geradores de tráfego de ataques foram simulados através do uso de containers personalizados com imagem *Alpine Linux*[6] para viabilizar a realização de exercícios realísticos de segurança destinados ao *blue team* (responsáveis por corrigir vulnerabilidade e defender). Finalmente, avaliamos a usabilidade da solução na turma da disciplina Segurança da Informação do curso de Redes de Computadores da UFQUIXADA. Para tanto, implementamos um estudo de caso e um roteiro que viabilizasse a condução do treinamento. Os alunos realizaram a prática e responderam o questionário guiado no *moodle2*. Em seguida, preencheram um questionário no *Google Forms*, usado para avaliar a viabilidade do uso da ferramenta em outras disciplinas de segurança.

A partir da análise dos dados coletados, podemos constatar que o CyRM obteve um bom índice de satisfação, tendo em vista que é um experimento precursor em Cyber Range (CR) na UFQUIXADA, com questões guiadas em ambiente emulado com todos os serviços, componentes finais simulados e cenário realístico de um ambiente corporativo. Esse treinamento é destinado a professores e estudantes da disciplina de Segurança da Informação. Os resultados obtidos foram a implementação do CyRM e a validação da solução.

Espera-se que esta plataforma passe a ser utilizada nas disciplinas de Segurança da Informação da UFQUIXADA e de outras universidades, a fim de preparar os estudantes para o combate aos ataques cibernéticos, contribuindo para a sua formação.

[5] http://mininet.org/
[6] https://hub.docker.com/_/alpine

1.1 Objetivos

1.1.1 Objetivo Geral

Propor e avaliar uma solução leve de *Cyber Range* para auxiliar no ensino de defesa para alunos da disciplina de Segurança da Informação do Campus da UFC em Quixadá.

1.1.2 Objetivos Específicos

1. Definir a arquitetura da plataforma de *Cyber Range* a partir de soluções já existentes e identificação de ferramentas aplicáveis;
2. Implementar a solução de *Cyber Range* guiado por um caso de uso;
3. Avaliar a solução proposta na turma da disciplina Segurança da Informação do curso de Redes de Computadores da UFQUIXADA.

O restante do trabalho foi organizado da seguinte forma: no Capítulo 2, apresentamos os principais conceitos que orientam este trabalho; no Capítulo 3, descrevemos os trabalhos relacionados; no Capítulo 4, apresentamos a arquitetura, implementação e roteiro de estudo do CyRM; no Capítulo 5, apresentamos os experimentos realizados e avaliação dos resultados e no Capítulo 6 apresentamos as conclusões e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo demonstramos os principais conceitos que orientaram este trabalho.

2.1 Segurança em Redes de Computadores

A Segurança da Informação tem como propósito a proteção de um conjunto de informações que possuem valor para um indivíduo ou uma instituição. Ela é fundamental porque protege de danos ou roubos dados confidenciais, informações de identificação pessoal, propriedade intelectual, dados e sistemas de informações, dentre outros (STALLINGS, 2014).

A Segurança da Informação é baseada em cinco pilares.

- **Autenticidade:** garante que o emissor de determinada informação seja realmente quem diz ser, assegurando que a informação seja proveniente de uma fonte confiável;
- **Confidencialidade:** refere-se ao modo como ocorre a proteção dos dados de uma organização ou indivíduo a fim de evitar ataques cibernéticos. Para impedir que informações confidenciais sejam furtadas, ações preventivas devem ser tomadas, como por exemplo, reduzir o número de colaboradores que têm acesso a informações sigilosas;
- **Disponibilidade:** garante que a informação esteja à disposição dos usuários autorizados para consulta a qualquer momento, assegura acessibilidade e uso rápido e confiável da informação;
- **Integridade:** certifica que os dados sejam corretos, autênticos e confiáveis, ou seja, não sofreram alterações ou destruição imprópria de informação;
- **Legalidade:** o uso da informação precisa seguir a norma constitucional do país, ou seja, deve estar de acordo com as leis aplicáveis, regulamentos, licenças e contratos.

2.1.1 Tipos de ataque

De acordo *OWASP Top Ten* [1], os ataques abaixo foram os mais relevantes em aplicações Web.

- **Injeção:** são falhas de injeção ao *SQL (Structured Query Language)* e *NoSQL (Not Only Structured Query Language)*. Esse ataque ocorre quando o atacante envia dados não confiantes para a base de dados em comandos ou realiza consultas não permitidas;
- **Quebra de Autenticação:** comumente, as aplicações Web possuem mecanismos de

[1] https://owasp.org/www-project-top-ten/

autenticação e gerenciamento de usuários. Caso esses mecanismos não estejam bem implementados, como *tokens* de sessão ou senhas, os invasores podem invadir passando-se por usuários verdadeiros e realizando ações não permitidas, que podem causar prejuízos;

- **Exposição de Dados Sensíveis:** existem muitas aplicações Web que não protegem corretamente as informações sensíveis. O atacante pode usufruir dessa falha e roubar ou modificar esses dados. Portanto, é de extrema importância que haja proteção, utilizando criptografia no armazenamento e no ato da transferência de informações;
- **XXE(*XML eXternal Entity*):** os ataques XXE geralmente ocorrem quando os *parsers* XML (*Extensible Markup Language*) são mal configurados e não estão atualizados. O XML possui um mecanismo em que o programador configura o caminho dos dados que estão armazenados, que pode ser tanto local como remoto. Se o atacante tiver acesso a esse caminho, pode alterar e/ou remover dados existentes.
- **Quebra de Controle de Acesso:** algumas restrições que o usuário possui para realizar determinadas funcionalidades, nem sempre são configuradas de forma apropriada. Com essas más configurações, os atacantes se aproveitam e realizam investidas para acessar os dados não autorizados e contas privadas. Dessa forma, podem realizar várias ações como visualizar, modificar, transferir e deletar esses dados;
- **Configurações de Segurança Incorreta:** o problema que acontece com mais frequência é a má configuração, como realizar o armazenamento em nuvem sem qualquer limitação tal como HTTP (*Hypertext Transfer Protocol*) com uma configuração pode demonstrar algumas informações sensíveis. Demais sistemas operacionais, aplicações, servidores, *framework* e etc, devem ser configurados de modo seguro e sempre atualizados;
- **XSS (*Cross-Site Scripting*):** falhas sempre ocorrem quando as aplicações Web processam dados não confiáveis, que não possuem qualquer validação. Também ocorrem quando o atacante passa códigos maliciosos usando uma API (*Application Programming Interface*) do *browser*, a partir de formulário HTML (*Hypertext Transfer Protocol*). O XSS permite que atacantes possam inserir *scripts* maliciosos fazendo com que suas vítimas sejam redirecionadas para determinados sites inseguro, além de permitir o roubo de sessão e modificação de páginas Web;
- **Desserialização Insegura:** é uma vulnerabilidade que tem como propósito a execução de código remoto. Tem como objetivo a realização de ataques para a obtenção de privilégios, injeção e repetição de código;

- **Utilização de Componentes Vulneráveis:** alguns componentes como módulos de *software*, *framework* e demais bibliotecas, quando são executados com o mesmo privilégio da aplicação. Se algum desses componentes possui vulnerabilidade, o atacante poderá explorá-la;
- **Registro e Monitorização Insuficiente:** causam incidentes que não foram mapeados. Possibilitando aos atacantes se aproveitarem do ambiente com foco em atacar outros sistemas, fazendo a manipulação de dados, extraindo e removendo.

2.1.2 Mecanismos de segurança

Segundo o CERT.br[2], é importante que indivíduos ou empresas tomem o máximo de precauções para que não sejam vítimas de crimes cibernéticos, pois tais crimes podem quebrar um ou mais pilares da segurança da informação e com isso prejudicar as vítimas. Portanto, é necessário utilizar mecanismos de segurança para reduzir o número de vulnerabilidades, através do uso de regras de filtragem, detecção e prevenção de comportamentos anormais na rede e dispositivos mal configurados. A seguir, descrevemos os principais mecanismos de segurança:

- **Política de segurança:** nessa etapa, são definidas as políticas que decidem os deveres e direitos que os usuários possuem. Portanto pode haver algumas políticas específicas, dentre elas de senhas, *backup*, privacidade, confidencialidade e política de uso aceitável. Em alguns casos não é aceitável que não haja responsabilidade na rede, não ocorra a cópia de arquivos ou qualquer dado que não seja autorizado e não enviar informações duvidosas;
- **Criptografia:** quando se usa criptografia dificulta o tráfego cifrando a comunicação e armazenamento das informações nos serviços na internet, para manter os dados incompreensíveis para usuários não permitidos;
- ***Backups*:** definição das políticas de proteção para o armazenamento dos dados, incluindo cópias de segurança e memória para armazenamento caso ocorra falhas ou roubo de dados. Os cuidados que podem ser tomados são: salvar arquivos em lugares diferentes como HDD(*Hard Disk Drive*), SSD(*Solid State Drive*) e servidores em nuvem proporcionando espelhamento de dados, é importante também sempre manter o *backup* atualizado e criptografar os dados mais sensíveis;
- **Registro de eventos (*Logs*):** ferramenta que registra as ações realizadas por programas em arquivos de dados. Através destes registros, é possível a análise de possíveis ocorrências,

[2] https://cert.br/

como: tentativas de acesso por usuário não autorizados, ataques de força bruta, problemas de *hardware*, etc. Caso algum ataque venha a acontecer, fica registrado o IP do suposto atacante, quantidade de tentativas, horário da ocorrência, protocolos que foram testados e autenticação sucedida. Alguns pontos que devem ser vistos com atenção são: verificar quantidade de armazenamento onde será alocado para que seja tudo armazenado com eficácia, sempre ficar atento para que pessoas não autorizadas não apaguem esses arquivos de *logs* e não dá permissão para todos os usuários, somente para quem está credenciado;

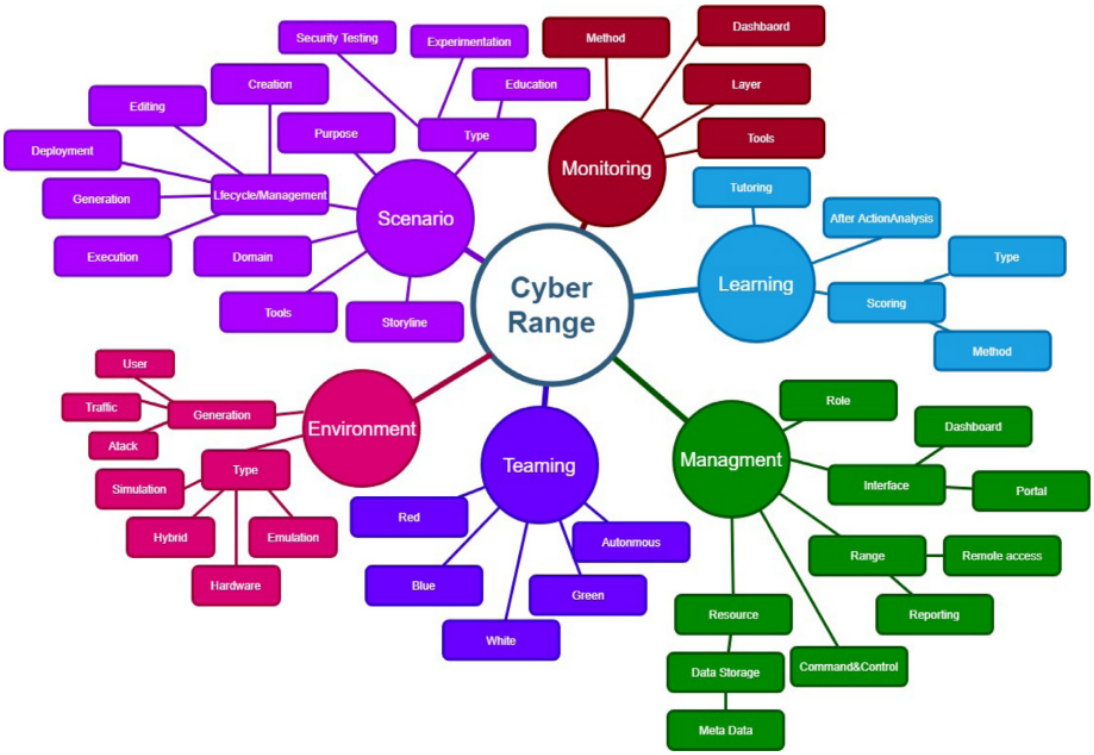
- **Serviços *antimalware*:** são ferramentas que procuram detectar *malwares*, além de neutralizar os ataques provenientes destes. Existem vários tipos de ferramentas *antimalware* como método de detecção que analisa se o código é malicioso, para obtenção desses *softwares* existem várias modalidades, gratuito, pago e versão teste. Em relação a execução pode-se usar mais de um *antimalware* em serviços conectados a *Internet*, na própria máquina e servidores. Deve-se ter alguns cuidados, como: manter a base de dados atualizada, configurar a verificação automática em dispositivos removíveis e evitar execuções de programas de *antimalware* em paralelo, de forma a evitar problemas de desempenho;
- ***Firewall*:** é uma ferramenta com a funcionalidade de controlar o tráfego passante em ativos da rede. Podendo bloquear ou permitir o tráfego de acordo com as regras definidas pelo administrador de rede.
- ***Honeypot*:** é uma ferramenta que possui as mesmas configurações do sistema operacional original e/ou de determinadas aplicações. Basicamente são iguais, mas o *honeypot* pode possuir algumas vulnerabilidades, para quando o atacante realizar o ataque ele pensar que está atuando de verdade e prejudicando a vítima. Contudo, o atacante não está causando nenhum impacto e sim, sendo abstraído de todas as informações e em determinados casos, ele pode realizar *downloads* maliciosos. Além disso, os ataques ficam registrados, o que contribui para melhorar a segurança do sistema, ver a origem do ataque e quais as senhas e *login* ele mais usou. Com essas informações pode-se prevenir novos ataques, tornando o sistema seguro;

2.2 *Cyber Range*

Cyber Range consiste em uma ferramenta destinada ao treinamento de indivíduos que pretendem seguir a carreira de analista de segurança, especializados na defesa contra ataques cibernéticos. Geralmente, um *Cyber Range* cria cenários de treinamento em ambientes

virtualizados e controlados, com emulação e/ou simulação, contendo exercícios práticos de ataque e defesa.

Figura 2 – Taxonomia atualizada de um intervalo cibernético.



Fonte: (YAMIN *et al.*, 2020).

A Figura 2 ilustra uma taxonomia que engloba todos os componentes que podem compor uma solução de *Cyber Range*. Descreveremos cada um deles a seguir.

Através do **Monitoramento** *(Monitoring)*, os participantes são monitorados ao realizar os exercícios de segurança. Dentre os itens mais importantes para realização do monitoramento, podemos citar:

- **Método** *(Method)*: são criados para monitorar as atividades e testes de segurança, que podem ser monitorados por ferramentas automatizadas ou manualmente;
- **Painel de controle** *(Dashbaord)*: Serve para controlar o ambiente com base nos *logs* capturados e armazenados, os demais *software* livres são utilizados para gerar os relatórios detalhados;
- **Camada** *(Layer)*: classifica os tipos de monitoramento que podem ser em vários tipos de camadas;
- **Ferramentas** *(Tools)*: são listadas e escolhidas para a realização da monitoração de atividades de segurança que tanto pode ser *software*, como *hardware*. Também possuem

gerenciamento de evento de percepção e invasão;

No **Aprendendo** *(Learning)* os participantes são ensinados por um tutor, para a execução do exercício de segurança da informação. Os itens para a aprendizagem e tutoria do aluno, serão destacados abaixo:

- **Tutoria** *(Tutoring)*: sistema utilizado para realizar uma análise detalhada dos participantes e de todos os exercícios realizados nos treinamentos e armazená-los;
- **Análise Pós-Ação** *(After Action Analysis)*: é necessário definir as métricas antes de iniciar a execução do experimento e aguardar a conclusão do mesmo, para que seja possível analisar os dados. Portanto, os dados experimentais são analisados em um grande conjunto de dados;
- **Pontuação** *(Scoring)*: são usados dados interligados ao sistema de monitoramento, que verificam o desempenho dos usuários no momento das atividades de segurança da informação e de determinados testes de laboratórios, ferramentas destinadas à segurança, etc. Dependendo da forma de avaliação escolhida, há vários métodos de mecanismo de pontuação que são utilizados para avaliar os estudantes ou as ferramentas para analisar o processo. Alguns desses mecanismos estão listados abaixo:
  - **Tipo** *(Type)*: destinado ao *software* e/ou *hardware* com o propósito de pontuar os testes e atividades de segurança. As ferramentas adicionam funcionalidades para analisar os *logs* e demais informações correlatas;
  - **Método** *(Method)*: são classificados de acordo com o propósito, que pode ser atingir um determinado objetivo ou analisar registros de *logs* gerados para os exercícios sobre segurança da informação, destinados aos estudantes e/ou a testes de ferramentas de segurança;

Na **Gestão** *(Management)* é realizado o gerenciamento das funcionalidades que o compõem, tais como controle dos recursos computacionais que produz as faixas cibernéticas. A seguir, exemplificaremos estas funcionalidades:

- **Função** *(Role)*: são definidas por cada equipe, e podem ter diferentes finalidades como atacar, defender, configurar o ambiente, corrigir vulnerabilidades, etc;
- **Interface** *(Interface)*: facilita o gerenciamento, pois demonstra de forma gráfica todas as ações que estão acontecendo em momentos simultâneos, como processamentos e operações de controle realizadas pelo gestor:
  - **Painel de instrumentos** *(Dashbaord)*: possui funções que permitem ao gestor

realizar instruções, de acordo com a necessidade e de modo gráfico, facilitando o manuseio das instruções a serem comandadas;

  - **Entrada** *(Portal)*: possui métodos eficazes na transmissão de dados com eficiência, acessível a todos os usuários, distribuindo recursos computacionais geograficamente, facilitando o acesso de todos;

- **Gama** *(Range)*: determina o gerenciamento de alcance que pode ser realizado com acesso remoto e relatório. Utiliza um *proxy* para realizar a comunicação remota e acessar os dispositivos físicos:
  - **Acesso remoto** *(Remote access)*: é utilizado para acessar máquinas que estão em longa distância e realizar determinadas funcionalidades, tais como inicializar um sistema, desligar, realizar tarefas de instruções e também monitorar em tempo real as atividades que estão sendo executadas;
  - **Comunicação** *(Reporting)*: possui a funcionalidade de descrever o projeto detalhado, com todas as partes que contém;
- **Controle e Comando** *(Command and Control)*: são utilizados em uma interface gráfica para facilitar o comando do sistema e realizar as funções, controlando de acordo com a necessidade, podendo visualizar o status das atividades executadas em tempo real;
- **Recurso** *(Resource)*: utilizado para o armazenamento de dados de forma autônoma e demais ferramentas utilizadas:
  - **Armazenamento de dados** *(Data Storage)* possui módulos para a realização de armazenamento de dados e configurações do sistema, cenários, ferramentas para realização de ataques e defesa, e uma série de outras partes que compõem o experimento:
    * **Meta Dados** *(Meta Data)*: são dados que referem-se a outros dados que possuem base, fazem um resumo das informações desses dados base que são referentes;

A **Equipe** *(Teaming)* é definida por cores. Portanto, cada uma das cores tem seu significado e suas funções. A equipe é formada por grupos de pessoas e cada grupo tem as suas responsabilidades. A seguir, iremos detalhar a funcionalidade das cores dos times:

- **Vermelho** *(Red)*: o time vermelho é responsável por atacar e identificar falhas e/ou vulnerabilidades em ambientes para treinamento e em ambientes de produção;
- **Azul** *(Blue)*: o time azul é responsável pela defesa do ambiente, sendo assim responsável por corrigir as vulnerabilidades encontradas pela equipe vermelha, pelas configurações dos ativos da rede, etc;

- **Branco** *(White)*: o time branco é responsável pela elaboração do ambiente, tanto da equipe que é responsável por realizar o ataque, como da equipe encarregada por executar a defesa, sendo assim, define regras, funcionalidades, configurações, avaliações, cria vulnerabilidades para que os membros encontre-as e corrija-as e dá instruções aos participantes;
- **Verde** *(Green)*: o time verde é responsável pela a implementação, monitoramento, manutenção projetada por ele mesmo, além de resolver *bugs* e travamentos durante o exercício;
- **Autônomo** *(Autonmous)*: a função dessa equipe é automatizar as ferramentas, cenário, dentre outras funcionalidades para economizar tempo e trabalho;

O **Ambiente** *(Environment)* executa o cenário com a definição de ferramentas. Os *softwares* utilizados para a implementação do ambiente podem ser virtualizados ou físicos, assim como o uso de ferramentas para gerar tráfego e/ou ataque. A seguir, descreveremos os elementos que compõem o ambiente:

- **Geração** *(Generation)*: é usado para gerar tráfegos e determinados tipos de ataques ou até mesmo determinados tipos de utilizadores;
  - **Utilizador** *(User)*: o usuário pode determinar as ferramentas para utilizar no momento de gerar os dados;
  - **Tráfego** *(Traffic)*: são simulados em ambientes emulados para que se aproximem o máximo da realidade e sejam mais seguros na realização dos testes;
  - **Ataque** *(Atack)*: são gerados em ambientes controlados e com o propósito de aumentar o nível de realidade e a segurança;
- **Tipo** *(Type)*: determina o tipo mais apropriado para o ambiente em que serão realizados os testes. Abaixo, mencionaremos os Tipos que podem ser utilizados no ambiente:
  - **Simulação** *(Simulation)*: é importante para a realização dos testes de aplicações, desempenho, etc. Sem causar nenhum prejuízo ao *hardware* ou ao próprio sistema com as funcionalidades;
  - **Híbrido** *(Hybrid)*: é utilizado com dois tipos de ambientes iguais simultaneamente, buscando a vantagem de um e o complemento do outro;
  - **Hardware** *(Hardware)*: é uma opção que permite que toda aplicação seja executada no próprio *hardware*, podendo ser mais preciso, porém pode haver uma maior probabilidade da máquina não responder a todos os comandos e gerar prejuízo no desempenho;

  - **Emulação** *(Emulation)*: tem o propósito de ser o mais parecido possível do sistema e do *hardware* que está sendo utilizado, sendo assim, causa mais produtividade e economia em geral;

No **Cenário** *(Scenario)* são definidas as etapas do treinamento, laboratórios de testes e o ambiente completo, como a definição do sistema operacional, requisitos de treinamentos, ferramentas a serem usadas e os objetivos. Adiante, apresentaremos os elementos que integram o cenário:

- **Tipo** *(Type)*: mostra que os cenários são dinâmicos ou estáticos. Quando definido o cenário estático, a aplicação não pode ser alterada enquanto a atividade estiver sendo executada. Porém, o enredo não pode incluir nenhum componente no cenário estático, somente no dinâmico, com um determinado tempo. O cenário dinâmico realiza a inclusão do estático, podendo ser alterado, ao incluir componentes durante a execução de um cenário. Um exemplo é quando um gerador de tráfego é injetado no momento em que o exercício está ativo. Abaixo, listamos os tipos de cenário do *Cyber Range*:
  - **Educação** *(Education)*: são definidos exercícios para que o usuário possa aprender sobre segurança da informação, ou até mesmo, se aperfeiçoar;
  - **Teste de Segurança** *(Security Testing)*: tem o propósito de testar a segurança do *hardware* e/ou *software* em ambiente controlado;
  - **Experimentação** *(Experimentation)*: realiza vários testes para a validação de um experimento, para comprovar se atinge o objetivo, e/ou aperfeiçoá-lo;
- **Propósito** *(Purpose)*: são os objetivos do cenário para validar novas ferramentas e/ou treinamentos de exercícios para a segurança da informação.
- **Ciclo/Gestão** *(Lfecycle/Management)*: tem como objetivo detalhar as partes do ciclo de gestão, do ponto inicial até a conclusão. A seguir, especificaremos as partes do Ciclo:
  - **Criação** *(Creation)*: ponto de partida em que é realizada a criação do ambiente e das funcionalidades;
  - **Edição** *(Editing)*: caso haja necessidade, ou não siga o escopo, poderá ser editado;
  - **Implementação** *(Deployment)*: realiza as funcionalidades que foram criadas e coloca-as em execução;
  - **Geração** *(Generation)*: gera dados da implementação que está sendo executada e observa o comportamento geral, para tomar decisões;
  - **Execução** *(Execution)*: executa a aplicação do ambiente e as funcionalidades, dessa

forma, é esperado que haja o funcionamento adequado da aplicação;

- **Domínio** *(Domain)*: mostra qual aplicação deve ser usada, como *Internet of things* (IoT), nuvem, rede, dentre outros;
- **Ferramentas** *(Tools)*: são destinadas para realizar a implantação do cenário, que tanto pode ser para a criação do cenário, como do enredo.
- **Enredo** *(Storyline)*: são relatos de uma história ou mais histórias relacionadas aos cenários dos exercícios. Inclui os pontos mais importantes que compõem os cenários, que estão interconectados com a narrativa do cenário. Com isso, pode-se experimentar novas tecnologias e realizar testes de ferramentas a serem pesquisadas.

2.3 Emulação com *containers Docker*

2.3.1 Mininet

Emulador é um *software* que cria funções iguais de ambientes desejáveis com o propósito de executar alguns programas, ou até mesmo, realizar a simulação de uma determinada aplicação.

O *Mininet*[3] é um emulador de redes de código aberto e implementado na linguagem de programação *python* do tipo CBE(*Container-Based Emulation*). Portanto, é um emulador leve com muitos recursos, tais como *host*, *switch* e controlador baseado no protocolo *OpenFlow*. Dentre cada componente emulado o *host* tem como funcionalidade mandar e receber tráfego, o *switch* é responsável por determinar as rotas aos seus destinos e o controlador determina a funcionalidade da rede, de controle e gerenciamento (HANDIGOL *et al.*, 2012).

2.3.2 Docker

De acordo com a documentação oficial, o **Docker**[4] é uma plataforma de código aberto, com o propósito de melhorar a performance do desenvolvimento, que permite separar a aplicação da infraestrutura e melhorar o gerenciamento da aplicação. Dessa forma, o *Docker* possui aspectos importantes, contém funcionalidades que o tornam vantajoso, tais como realizar a implantação e os testes em tempo real, ser executado em ambiente de produção e executar vários *containers* em um só momento. Além disso, possui outras funcionalidades benéficas,

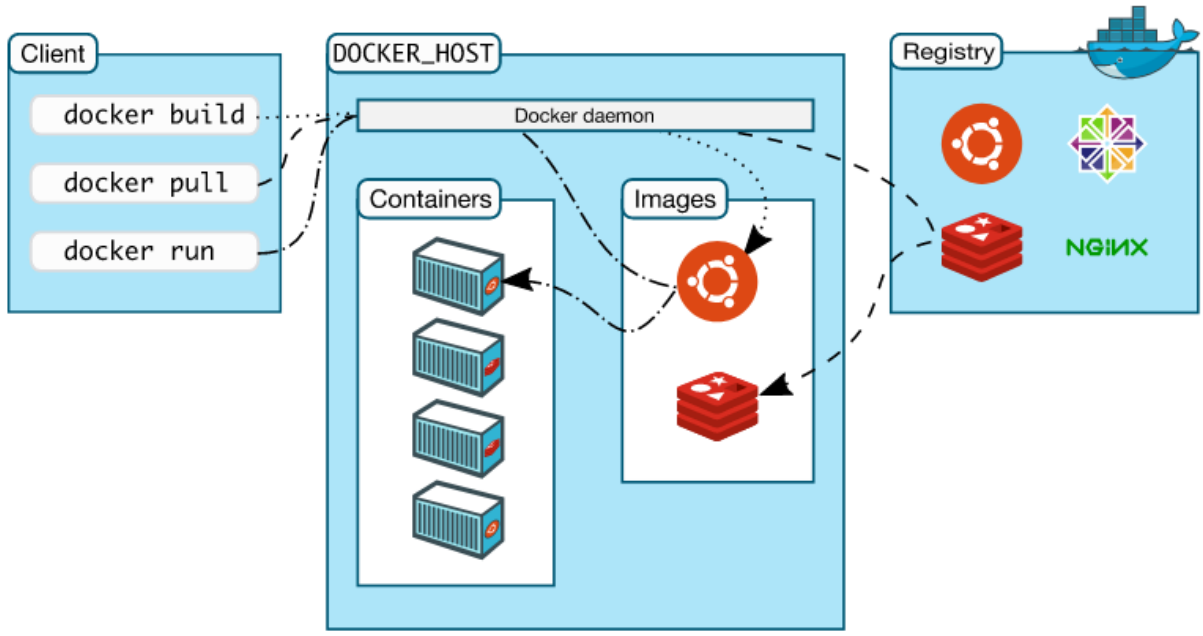
[3] http://mininet.org/
[4] https://docs.docker.com/get-started/overview/

como isolamento e segurança, que utilizam menos recursos do *hardware*, por isso, pode-se utilizar várias tarefas na mesma máquina sem causar danos.

O *Docker* pode ser utilizado em várias situações, por exemplo, programadores podem desenvolver aplicações em ambiente local, e posteriormente, compartilhar com outros colegas de trabalho, usando o *Container Docker*. Pode ser usado também para a realização dos testes, que tanto podem ser manuais, como automatizados, no momento em que os desenvolvedores estão realizando os testes e são encontrados os erros, para corrigi-los, basta atualizar a imagem e enviar para o ambiente.

A Figura 3 ilustra a arquitetura do *Docker*. O cliente *Docker* se comunica com o *deamon*, que realiza as principais tarefas de construir, executar e distribuir os *Container Docker*. Portanto, o cliente e o *deamon* podem ser executados ao mesmo tempo ou separadamente com um cliente ao *daemon* remoto. A comunicação é realizada pela *API REST* por soquetes *UNIX* ou até mesmo por uma interface de rede. O cliente *Docker* possui outro modelo que é o *Docker Compose*, que possibilita trabalhar em conjunto com todas as aplicações utilizando os *containers*. Abaixo apresentamos uma figura que demonstra a arquitetura do *Docker*.

Figura 3 – Arquitetura Docker.



Fonte: https://docs.docker.com/get-started/overview/

O *Docker Hub*[5] é um repositório gratuito com imagens *container* pré-compiladas, que pessoas ou empresas armazenam com o intuito de compartilhar ferramentas, possui uma

[5] https://hub.docker.com/

variedade imensa dessas imagens, tais como *python, ubuntu, postgres, kali, alpine*, entre outros.

2.3.3 Containernet

Containernet é um *fork* do *Mininet* que permite a criação de nós a partir de *Containers Docker* em topologias de rede emuladas, o que o torna mais flexível do que o *Mininet*. Em (PEUSTER *et al.*, 2016), os autores relatam que uma das vantagens do *Containernet* é a possibilidade de adicionar e remover *containers* da rede emulada, em tempo real, o que não é possível no *Mininet*, além da possibilidade de alterar a limitação de recursos como memória RAM e processamento para um único *container* em tempo real, não somente durante a inicialização.

As funcionalidades acima permitem que o *Containernet* seja utilizado como uma ferramenta para a emulação de infraestruturas de Nuvem, onde é possível iniciar e parar instâncias computacionais durante o experimento. Por esse motivo, ele tem sido aplicado em experimentos no campo da Computação em Nuvem, *Fog Computing*, *Network Function Virtualization* (NFV) e *Multi-access Edge Computing* (MEC).

Neste trabalho, utilizamos o *Containernet* para a criação do cenário de *Cyber Range* utilizando topologia emulada de rede com *Containers Docker* simulando serviços, dispositivos finais, ataques e vulnerabilidades. Geradores de tráfego de ataques foram simulados através do uso de *containers* **Alpine Linux**[6] para viabilizar a realização de exercícios realísticos de segurança destinado ao *blue team*.

[6] https://hub.docker.com/$_{/alpine}$

3 TRABALHOS RELACIONADOS

Nesta seção descreveremos os trabalhos relacionados ao nosso projeto. Vale ressaltar que orientamos a nossa busca apenas para soluções de *Cyber Range* que fizeram uso de ferramentas *open source*.

3.1 CyRIS: A Cyber Range Instantiation System for Facilitating Security Training

Em (PHAM *et al.*, 2016), os autores propuseram um *Cyber Range*, chamado ***CyRIS***[1], composto por ambientes virtualizados e controlados, sendo usado para treinamentos de segurança da informação através da aplicação de atividades práticas para a obtenção de conhecimentos. Para o treinamento foi necessário criar uma faixa cibernética proposital para os alunos aprenderem a identificar vulnerabilidades que continham no sistema, sendo emulados dois tipos de ataque. O ataque estático foi introduzido antes de começar o treinamento com o objetivo dos participantes descobrirem os ataques que foram realizados, tais como *DDoS* e *brute force SSH*. E o ataque dinâmico que acontece em tempo real com o intuito dos alunos tomarem alguma atitude para se defender. Foram usadas imagens de virtualização do *KVM* utilizando o *framework CyTrONE*[2] para automatizar todas as tarefas e definir algumas configurações necessárias como endereço *IP*, nome do *host*, etc. Para a preparação do ambiente, foram criados vários *hosts* em uma mesma rede *LAN*. Em seguida, foi configurada a emulação de ataques, captura dos pacotes e emulação de *malware*, sendo uma das mais importantes atividades realizadas no *Cyber Range*. Vale ressaltar que o *CyRIS* isola a rede emulada do ambiente externo para que não ocorra risco de vazamento dos ataques.

Semelhante a (PHAM *et al.*, 2016), o nosso trabalho também foca em ambientes virtualizados e controlados. Contudo, os autores em (PHAM *et al.*, 2016) utilizaram o *KVM*, um hipervisor que virtualiza toda a pilha de *hardware* e *software*. Em nosso trabalho, para a criação das topologias de rede emuladas, usamos o *Containernet* que permite uma emulação mais leve e eficiente através do uso de *Containers Docker*. Além disso, os autores em (PHAM *et al.*, 2016) realizaram apenas treinamento de defesa contra ataques cibernéticos. Em nosso trabalho, construímos cenários de treinamento para defesa, em um modelo de *blue team*.

[1] https://github.com/crond-jaist/cyris/
[2] https://github.com/crond-jaist/cytrone

3.2 Comprehensive Cyber Arena; The Next Generation Cyber Range

Em (Karjalainen; Kokkonen, 2020), os autores propuseram um modelo conceitual para uma solução de *Cyber Arena*. *Cyber Arena* consiste em um ambiente de treinamento em segurança cibernética complexo e de máximo realismo. Para tanto, ele orienta o uso de *Cyber Range*, com suas tecnologias e fenômenos de cibersegurança, em ambientes heterogêneos de larga escala. Neste trabalho, os autores apresentaram os requisitos de alto nível para a implementação de soluções *Cyber Arena*, como por exemplo: ambientes realísticos, controlados e isolados; simulação da Internet com suas estruturas e serviços; simulação de cenários organizacionais (e.g. provedores de serviço), conectados à Internet; capacidade para a simulação de tráfego de rede proveniente de usuário e aplicações; simulação e execução de ataques; colaboração e cooperação com outras plataformas de treinamento. Finalmente, um *Cyber Arena* deve permitir o planejamento, execução, monitoramento e análise de exercícios, através de ferramentas, de forma que os instrutores possam avaliar o desempenho dos alunos, bem como os alunos possam se auto-avaliar, exercitando um dos elementos de aprendizado mais importantes, a reflexão.

Semelhante a (Karjalainen; Kokkonen, 2020), o nosso trabalho focou em soluções de *Cyber Range*. Contudo, enquanto os autores em (Karjalainen; Kokkonen, 2020) definiram apenas um modelo conceitual, nós implementamos e avaliamos a solução de *Cyber Range*, usando o *Containernet* e simulando cenários de treinamento para defesa, em um modelo de *blue team*. Entretanto, usamos os requisitos definidos acima para guiar o desenvolvimento da nossa solução.

3.3 AIT Cyber Range: Flexible Cyber Security Environment for Exercises, Training and Research

No (LEITNER *et al.*, 2020), os autores descreveram a arquitetura, a implementação e os casos de uso do *AIT Cyber Range* com o objetivo de fornecer uma arquitetura flexível e escalável dividida em quatro blocos de construção: plataforma de computação, provisionamento de infraestrutura, provisionamento de *software* e mecanismo de cenário. Com o intuito de realizar exercícios de segurança, treinamentos e pesquisas relacionadas à segurança da informação. Todas as ferramentas utilizadas são de código aberto, a ferramenta *Open Stack* foi utilizada para facilitar a criação da infraestrutura. Na realização dos exercícios, cada equipe possuía um domínio, ficando isolada das outras equipes, porém, com acesso às funcionalidades da plataforma. O *VMware* foi usado para a realização da virtualização, os ataques foram realizados

de forma automatizada para que o usuário melhorasse seu desempenho na defesa. Nesse trabalho, utilizaram ainda IDS(Sistema de Detecção de Intrusão), que possui um *firewall* atualizado para melhorar o mecanismo de defesa.

Similar a (LEITNER *et al.*, 2020), nosso trabalho focou em *Cyber Range*, e realizou exercícios de segurança da informação. É composto por uma arquitetura que consome menos recursos das máquinas. Diferentemente do (LEITNER *et al.*, 2020), usamos o *Containernet* e simulação, o cenário não teve acesso a internet, inviabilizando vulnerabilidades e ataques cibernéticos externos. Dessa forma, os alunos aprenderam a se defender, como no modelo *bluem team*.

3.4 Gamifying ICS Security Training and Research: Design, Implementation, and Results of S3

O (ANTONIOLI *et al.*, 2017) se propõe a encarar os desafios enfrentados na educação e pesquisa sobre a segurança dos Sistemas de Controle Industrial (ICS). A pesquisa foi dividida em duas etapas, a fase online foi de treinamento e apresentação de novas categorias encontradas em CTFs (*Capture The Flag*) em que é composto o modo ataque e defesa o CTF (*Capture The Flag*), são separados em dois times, o vermelho, que é responsável por atacar e o azul que é responsável por defender.

As duas equipes têm a mesma máquina virtual para corrigir vulnerabilidades e atacar a máquina da outra equipe, ambas alocadas em uma mesma rede LAN (*Local area network*). No estilo *Jeopardy* (são criados vários tipos de categorias, com a realização de diferentes tipos de exercícios e demais segmentos como exploração, criptografia e engenharia reversa, de acordo com cada atividade, possui os exercícios relacionados e com uma pontuação de acordo com cada tarefa concluída.) Na segunda fase, ao vivo, as equipes de ataque e equipes de defesa tiveram acesso à bancada de teste de distribuição de água (SWaT), fazendo com que o procedimento estivesse mais próximo da realidade, como em uma empresa de verdade. Implantaram uma ampla gama de ataques, enquanto dois acadêmicos executavam sistemas de detecção de ataques. Para a implantação do trabalho, foram criadas competições de segurança gamificadas, destinadas a profissionais industriais e acadêmicos da área de segurança.

Equivalente ao (ANTONIOLI *et al.*, 2017), no nosso trabalho foi desenvolvido um *cyber range* que consome menos recursos da máquina e utilizamos a topologia de rede *Containernet*, o (ANTONIOLI *et al.*, 2017) utilizou o *mininet*, que possui menos funcionalidades

e consome mais memória, processamento, etc.

3.5 Análise Comparativa

O quadro 1 compara os trabalhos relacionados com o nosso. De modo a facilitar o entendimento dos trabalhos relacionados, em analogia ao nosso trabalho, o quadro abaixo apresenta um resumo comparativo. Neste resumo, apresentamos as diferenças e semelhanças do nosso trabalho e dos trabalhos relacionados, segundo suas propostas de implementação, ferramentas de virtualização, plataforma, tipos de exercícios e cenário.

Quadro 1 – Análise comparativa entre trabalhos relacionados e esse trabalho.

Trabalho	**Implementação**	**Ferramenta de Virtualização**	**Plataforma**	**Tipo de Exercício**	**Cenário**
(PHAM *et al.*, 2016)	Sim	KVM	CyTrONE	Defesa	Redes Organizacionais
(Karjalainen; Kokkonen, 2020)	Não	N/A	N/A	Ataque	Redes Organizacionais
(LEITNER *et al.*, 2020)	Sim	VMware	OpenStack	Defesa	Internet
(ANTONIOLI *et al.*, 2017)	Sim	Linux Containers	Mininet	Ataque e Defesa	Sistema de Controle Industrial
Este trabalho	Sim	Docker	Containernet	Defesa	Redes Organizacionais

Fonte: elaborado pelo autor

Conforme podemos observar, somente no trabalho (Karjalainen; Kokkonen, 2020) não há implementação do *Cyber Range*, no entanto, foram descritos os conceitos mais importantes que a plataforma deve alcançar. Os demais foram implementados.

Os trabalhos (Karjalainen; Kokkonen, 2020) e (ANTONIOLI *et al.*, 2017) não demonstraram as ferramentas de virtualização usadas, já o (PHAM *et al.*, 2016) e o (LEITNER *et al.*, 2020) utilizaram o *KVM* e o *VMware*, respectivamente. Em contrapartida, iremos utilizar a ferramenta *Docker*, visto que, consome menos recurso do *hardware*.

Na topologia de rede, utilizamos o *Containernet*, que também consome menos recursos computacionais, pois, nosso foco é um ambiente fechado, sem conexão externa. Já o trabalho (LEITNER *et al.*, 2020) é implementado em nuvem, ocasionando menos segurança, visto que, há mais possibilidade da plataforma ser comprometida, pois possui um alcance maior. No trabalho (ANTONIOLI *et al.*, 2017) é implementado o *mininet*, para a realização da topologia, não havendo muitas funcionalidades. Já os trabalhos (PHAM *et al.*, 2016) e (Karjalainen; Kokkonen, 2020), não apresentam topologia de rede.

Nossos exercícios têm o propósito de fazer com que os alunos aprendam a se defender, semelhante aos trabalhos (PHAM *et al.*, 2016) e (LEITNER *et al.*, 2020) em que os ataques

são automatizados e os participantes criaram estratégias para se proteger. O nosso *Cyber Range* também possui o mesmo propósito, porém os ataques inicializaram a partir da inicialização da topologia de forma automatizada. No trabalho (ANTONIOLI *et al.*, 2017) os usuários realizam os ataques e a defesa. Já o trabalho (Karjalainen; Kokkonen, 2020), não possui nenhum tipo de exercício porque não foi implementado.

Nos cenários utilizamos redes organizacionais, pois o nosso objetivo foi manter o isolamento e o controle. Da mesma forma, os trabalhos (PHAM *et al.*, 2016) e (Karjalainen; Kokkonen, 2020) foram em redes *LAN* de organização, já o trabalho (LEITNER *et al.*, 2020) foi implementado diretamente na *Internet*, tendo sua infraestrutura em nuvem. Sob outra perspectiva, o trabalho (ANTONIOLI *et al.*, 2017) foi realizado com o propósito de controle industrial.

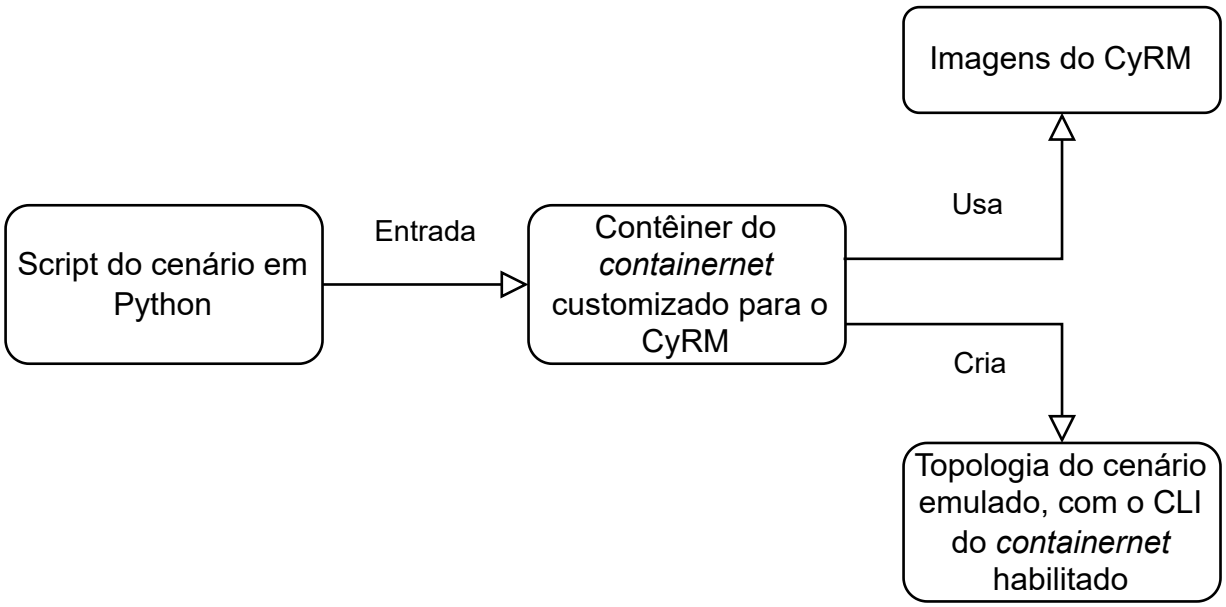
4 SOLUÇÃO PROPOSTA

Neste capítulo, definimos a arquitetura do CyRM, bem como apresentamos o cenário desenvolvido para a implementação de uma prova de conceito.

4.1 Arquitetura do CyRM

A Figura 4 ilustra a arquitetura da nossa solução de *Cyber Range*.

Figura 4 – Fluxograma do CyRM.



Fonte: Elaborado pelo autor.

O CyRM executa cenários controlados voltados para o treinamento de defesa. Tais cenários consistem em topologias emuladas, que utilizam o *Containernet* como ferramenta de emulação. Este emulador, permite a execução de nós como *Containers Docker*, o que tanto reduz o consumo de recursos computacionais como possibilita uma maior flexibilidade para a criação de cenários, dado a quantidade de imagens disponíveis em repositórios públicos e a liberdade para criação de novas imagens. Portanto, um dos pré-requisitos da ferramenta é ter o *Docker Engine* instalado.

Neste sentido, todos os cenários de emulação devem ser implementados como *scripts Python*, utilizando as bibliotecas do *Containernet*. Para executar estes *scripts*, utilizamos um *Contêiner Docker* baseado na imagem ***containernet_cyrm***. Esta imagem tem como base a imagem do *containernet*[1] (Ubuntu 18.04) com algumas customizações. A Figura 5 ilustra o

[1] https://hub.docker.com/r/containernet/containernet/

Dockerfile do *containernet_cyrm*.

Figura 5 – *Dockerfile* do *containernet_cyrm*.

```
rafael@CyRM:~/cyrm/images/containernet_cyrm_image$ cat -n Dockerfile
     1  FROM containernet/containernet
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apt update -y
     4  RUN echo "wireshark-common wireshark-common/install-setuid boolean true" | debconf-set-selections
     5  RUN DEBIAN_FRONTEND=noninteractive apt-get -y install tshark
     6  CMD ["/bin/sh"]
```

Fonte: Elaborado pelo autor.

Com o uso desta imagem, o usuário não precisa instalar o *Containernet* localmente, basta ter o *Docker Engine* instalado. Vale ressaltar que, para a emulação de roteadores no cenários, utilizamos a tabela de rotas do próprio *host* que executa a topologia (*contêiner* do *containernet_cyrm*). Isso se deve pelo fato de não termos encontrado imagens *Docker* estáveis que implementem roteadores. Portanto, todos os roteadores virtuais são baseados em *Linux (Ubuntu 18.04)*. Neste caso, todos os comandos que precisam ser executados no roteador, devem ser instalados na imagem do *containernet_cyrm*. Na Figura 5 é possível visualizar que o *Tshark* foi instalado para futuras análises de tráfego que precisem ser feitas em roteadores virtuais.

O cenário é executado pelo *contêiner containernet_cyrm*, este último recupera as imagens do CyRM (previamente construídas e armazenadas no repositório local de imagens) e constrói a topologia emulada. Ao final da construção, o *command-line interface* (CLI) do *Containernet* é mostrado, permitindo que o usuário interaja com os nós do cenário, conforme ilustrado na Figura 6.

Figura 6 – Execursão da topologia CyRM.

```
gabriele: kwargs {'ip': '10.0.20.102/24', 'mac': '00:00:00:00:00:60', 'defaultRoute': 'via 10.0.20.254'}
gabriele: update resources {'cpu_quota': -1}
1:
henrique: kwargs {'ip': '10.0.20.103/24', 'mac': '00:00:00:00:00:70', 'environment': {'TARGET_IP': '10.0.30.100'}, 'defaultRoute': 'via 10.0.2
0.254'}
henrique: update resources {'cpu_quota': -1}
1:
joyce: kwargs {'ip': '10.0.20.104/24', 'mac': '00:00:00:00:00:80', 'environment': {'TARGET_IP': '10.0.30.100'}, 'defaultRoute': 'via 10.0.20.2
54'}
joyce: update resources {'cpu_quota': -1}
1:
dic-attack-ssh; latest; None; sha256:e31b65f2f754be6f0cfe9cfcee0a85deb18121b32daceb0a8066bce74a246349
dic-attack-ssh; latest; None; sha256:9cf1c903b74b6a7630b998c3703144398f61239a3214a6fac5041539c5784c52
internet: kwargs {'ip': '1.178.218.56/24', 'mac': '00:00:00:00:00:90', 'environment': {'TARGET_IP': '100.0.0.101'}, 'defaultRoute': 'via 1.178
.218.254'}
internet: update resources {'cpu_quota': -1}
*** Adding switches
*** Adding routers
*** Associating and Creating links
 *** Starting networket-eth0 up:
*** Configuring hosts
suporte servSamba servWeb joao gustavo gabriele henrique joyce internet r0
*** Starting controller
c1
*** Starting 10 switches
s1 s2 s3 s4 s5 port1 port2 port3 port4 port5 ...
*** Routing Table on Router:
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
1.178.218.0     0.0.0.0         255.255.255.0   U     0      0        0 r0-eth5
10.0.10.0       0.0.0.0         255.255.255.0   U     0      0        0 r0-eth1
10.0.20.0       0.0.0.0         255.255.255.0   U     0      0        0 r0-eth2
10.0.30.0       0.0.0.0         255.255.255.0   U     0      0        0 r0-eth3
100.0.0.0       0.0.0.0         255.255.255.0   U     0      0        0 r0-eth4

*** Running CLI
*** Starting CLI:
containernet>
```

Fonte: Elaborado pelo autor.

Finalmente, um roteiro de estudo foi montado para orientar o usuário no processo de interação com a topologia. Tal roteiro foi construído como um questionário no *Moodle*[2]. A criação desse roteiro fica a cargo de quem implementa o cenário. A seguir, apresentamos o cenário que foi implementado como prova de conceito do CyRM.

4.2 Definição do Cenário para a Prova de Conceito

Para construirmos uma prova de conceito para o CyRM, definimos e implementamos um cenário onde realizamos um treinamento de defesa contra ataques. A Figura 7 ilustra a topologia do cenário.

A topologia consiste em uma rede corporativa com topologia estrela. A infraestrutura de rede permite acesso a Internet simulada e contém 4 *VLANs (Virtual Local Area Network)*: *DMZ* 01 *(Demilitarized Zone)*, *DMZ* 02, USUÁRIOS e *BLUE TEAM* (suporte).

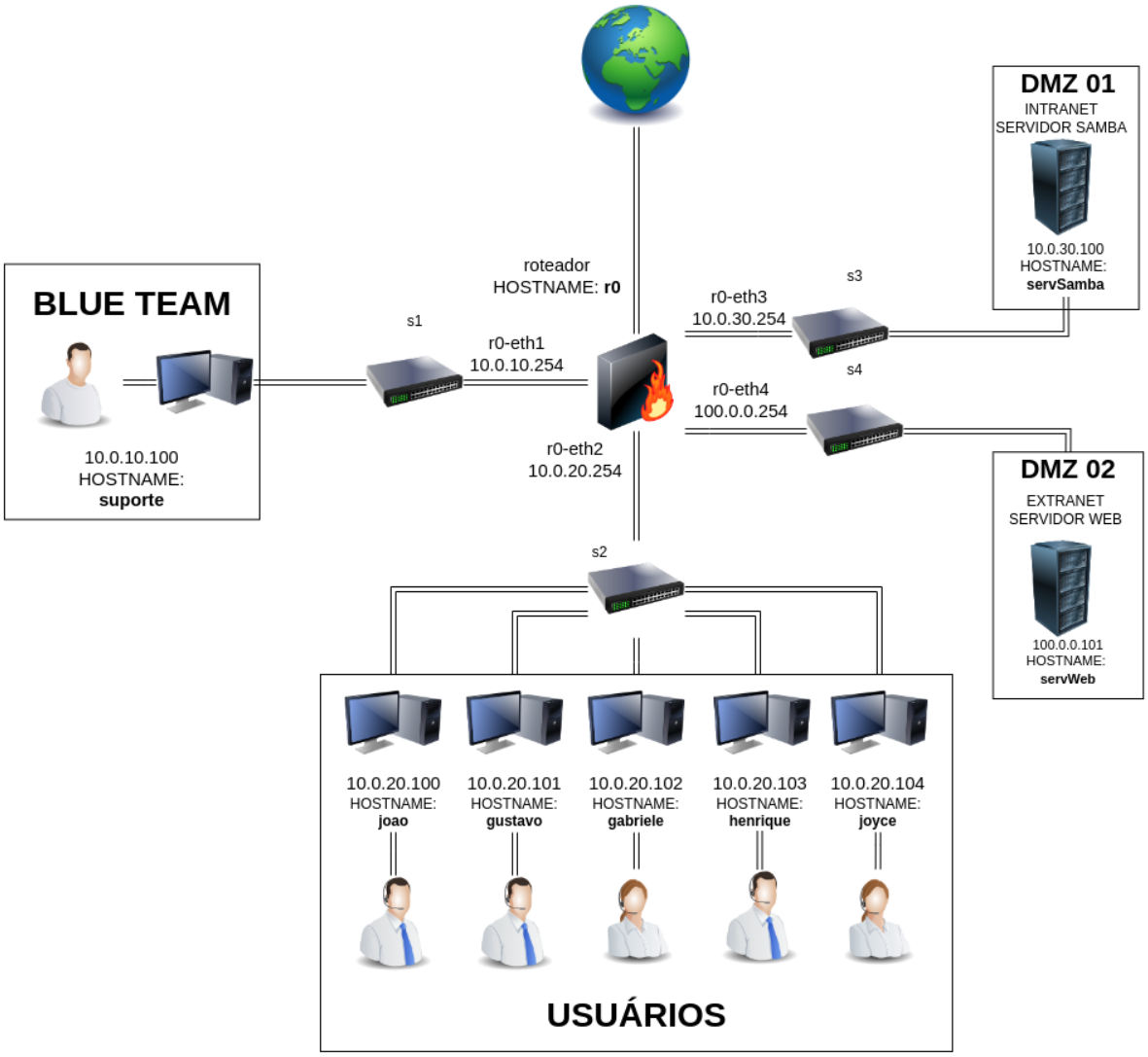
A ***DMZ* 01** consiste em uma *VLAN Intranet*, ou seja, é acessível apenas na rede interna, com a sub-rede 10.0.30.0/24. Nela, existe apenas um servidor de arquivos *Samba* que é responsável pelo compartilhamento interno de arquivos, diretórios entre os funcionários da empresa. Seu endereço IP é o 10.0.30.100 *(servSamba)* e está conectado ao *switch* s3.

A ***DMZ* 02** consiste em uma *VLAN Extranet*, com a sub-rede 100.0.0.0/24 (IPs válidos). Nela, existe apenas um servidor *Web* que tem por finalidade fornecer os serviços de armazenamento da empresa na *internet*, de forma que os clientes possam acessar de qualquer lugar. Também possui um serviço *SSH* disponível para o usuário do suporte se conectar e realizar manutenções necessárias. Outra função que o servidor contém é o *syslog-ng* que possui funcionalidade de armazenar e gerenciar todos os registros de *logs* que acontecem no serviço, permitindo ao suporte a identificação das tentativas de acesso dos usuários não autorizados. Portanto o *SSH* possui uma configuração simples, tais como porta padrão, nome de usuário e senha fácil, que facilita o acesso para o atacante. O endereço do servidor é 100.0.0.101 *(servWeb)* e está conectado ao *switch* s4.

A ***VLAN* USUÁRIOS** consiste em uma rede interna privada que engloba os funcionários da empresa responsável por tratar de assuntos com os clientes, dando suporte e solucionando as pendências. Com isso, dois usuários atacam o servidor *Samba*, enviam vários pacotes simultaneamente, ocasionando travamento no serviço, isso caracteriza um ataque *DDoS (Distributed Denial of Service)*, utilizamos a ferramenta *hping3* para a realização do ataque, alocamos a

[2] https://moodle.org/?lang=pt_br

Figura 7 – Topologia do CyRM.



Fonte: Elaborado pelo autor.

sub-rede 10.0.20.0/24 e as máquinas (5 PCs) foram conectadas ao *switch* s2.

A ***VLAN BLUE TEAM*** consiste em uma rede interna privada que engloba os funcionários do suporte técnico. Tais funcionários são responsáveis pela administração e gerência de toda a infraestrutura da rede, servidores e terminais. São responsáveis por deixar o serviço das aplicações disponíveis e resolver qualquer problema que aconteça na infraestrutura. As principais ferramentas instaladas na máquina do suporte são *smbclient* que possui a finalidade de conectar com o servidor *Samba* para realização de manutenção e tarefas afins; o *tshark* é um *software* baseado no *Wireshark* destinado a captura e análise do tráfego, possuindo a interface *CLI* para utilizar em terminal *linux* e consumir menos recursos; o *SSH Client* para conectar ao servidor *Web* e resolver qualquer pendência; e o *software* instalado é *geoiplookup* responsável por identificar de qual país de origem é o atacante, introduzindo o endereço *IP*. Para tanto, foi

alocada a sub-rede 10.0.10.0/24 e as máquinas foram conectadas ao *switch* s1. Nesta *VLAN*, existe apenas uma estação de trabalho com endereço 10.0.10.100 (suporte).

Por outro lado, o roteador r0 tem por objetivo prover a interconectividade entre todos as *VLANs* e a *Internet*. O r0 consiste em um roteador baseado em *linux* com um firewall integrado *(netfilter/iptables)*, destinado a filtrar o tráfego passante, também instalado o *Tshark* para capturar e analisar o tráfego.

A ***INTERNET*** consiste em uma rede externa da corporação, onde foi simulado o tráfego e o surgimento do ataque de dicionário ao servidor web com a ferramenta *THC Hydra* para tentar descobrir qual login e senha estavam corretos.

Para a emulação da topologia utilizamos o *containernet*, que possui a funcionalidade de emular toda a rede e a inicialização de todos os *hosts*, serviços e servidores. Cada componente era um *Contêiner Docker*. A seguir, detalhamos a implementação das imagens deste cenário bem como os *scripts* usados para inicialização e finalização do cenário.

4.3 Implementação da Prova de Conceito do CyRM

Nessa etapa demonstramos a criação das imagens *Docker* com todas as ferramentas necessárias para execução do cenário. A prova de conceito está disponível no ***GitHub***[3].

4.3.1 Imagem do Ataque Distributed Denial of Service (DDoS)

A criação da imagem do ataque DDoS tem por finalidade gerar *contêiners* que enviem vários pacotes simultâneos usando o protocolo *Transmission Control Protocol* (TCP), na porta específica do serviço para que este último fique inacessível. A Figura 8 mostra o *Dockerfile* desta imagem. Na imagem base, foi utilizada o *Alpine Linux*. Instalamos o *Bash*, para habilitar o funcionamento da imagem no *containernet* (pré-requisito). Além disso, instalamos o *hping3* (e suas dependências), ferramenta responsável por realizar o ataque de *DDoS* possuindo várias funcionalidades. O *script* do ataque foi copiado para dentro da imagem, de forma que a mesma ficasse habituada a realizar o ataque quando o *contêiner* fosse inicializado.

O *script* do ataque de *DDoS* foi criado em *Shell Script*. Na terceira linha do *script*, a variável *TARGET* recebeu o endereço *IP* que foi o nosso alvo, livre para escolher qualquer endereço. Na sétima linha fizemos uma verificação de comunicação. O ataque só começou

[3] https://github.com/rafaelpinheiroo1/cyrm

Figura 8 – *Dockerfile* do ataque de *DDoS*.

```
rafael@CyRM:~/cyrm/images/ddos_image$ cat -n Dockerfile
     1  FROM alpine:latest
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apk update
     4  RUN apk add bash \
     5          tcpdump \
     6          iperf \
     7          busybox-extras \
     8          iproute2 \
     9          iputils
    10  RUN apk add hping3 --update-cache --repository http://dl-cdn.alpinelinux.org/alpine/edge/testing
    11  COPY ddos.sh /
    12  CMD ["/bin/sh","-c","/ddos.sh"]
```

Fonte: Elaborado pelo autor.

quando o alvo estava disponível, a oitava linha foi o *hping3* realizando o ataque, estava atacando na porta 445 que era o *Server Message Block (SMB)* no protocolo TCP.

Figura 9 – *Script* do ataque de *DDoS*.

```
rafael@CyRM:~/cyrm/images/ddos_image$ cat -n ddos.sh
     1  #!/bin/bash
     2
     3  TARGET=$TARGET_IP
     4
     5  echo "Disparando ataques para o host:"
     6  echo ${TARGET}
     7  while true; do ping -c1 ${TARGET} > /dev/null && break; done
     8  hping3 -c 10000 -d 1024 -S -w 64 -p 445 --flood ${TARGET}
     9  /bin/bash
```

Fonte: Elaborado pelo autor.

Criamos um diretório destinado para a criação da imagem, possuindo um arquivo, cujo nome é *Dockerfile* e outro arquivo com o *script* de ataque. Possui a finalidade de instalações de programas, pacotes, formas de chamar o container e a imagem base que foi executada. Após o arquivo com todas as dependências, usamos o comando do *Docker* para criar a imagem que necessitávamos e definimos o nome que desejávamos. Abaixo apresentamos a Figura 10 que demonstra o comando para a criação da imagem de *DDoS*.

Figura 10 – Comando para a criação da imagem de *DDoS*.

```
rafael@CyRM:~/cyrm/images/ddos_image$ sudo docker build . -t ddos-attack
Sending build context to Docker daemon  3.072kB
Step 1/7 : FROM alpine:latest
 ---> e66264b98777
Step 2/7 : LABEL maintainer="RafaelPinheiro"
 ---> Using cache
 ---> 3dffdf3cce4d
Step 3/7 : RUN apk update
 ---> Using cache
 ---> bfe2fb0f2cfd
Step 4/7 : RUN apk add bash         tcpdump         iperf         busybox-extras         iproute2         iputils
 ---> Using cache
 ---> 46603420e603
Step 5/7 : RUN apk add hping3 --update-cache --repository http://dl-cdn.alpinelinux.org/alpine/edge/testing
 ---> Using cache
 ---> b1f6443ac025
Step 6/7 : COPY ddos.sh /
 ---> Using cache
 ---> 97d54fae9877
Step 7/7 : CMD ["/bin/sh","-c","/ddos.sh"]
 ---> Using cache
 ---> e31b65f2f754
Successfully built e31b65f2f754
Successfully tagged ddos-attack:latest
```

Fonte: Elaborado pelo autor.

4.3.2 Imagem do Ataque de Dicionário ou Força Bruta

A imagem do ataque de dicionário contém todas as dependências necessárias para a realização do ataque de forma automatizada. A Figura 11 ilustra o seu *Dockerfile*. Como todas as imagens, é necessário a instalação do *Bash* para que a mesma funcione com êxito na topologia. A imagem base que utilizamos foi o *Alpine Linux*, uma distribuição leve, com o objetivo de consumir menos memória e armazenamento. A ferramenta que utilizamos para realizar o ataque foi o *THC Hydra*[4]. Com o *script* realizamos o ataque e automatizamos o disparo.

Figura 11 – Dockerfile do ataque de dicionário.

```
rafael@CyRM:~/cyrm/images/dic_ssh_image$ cat -n Dockerfile
     1  FROM alpine:latest
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apk update
     4  RUN apk add bash \
     5          tcpdump \
     6          iperf \
     7          busybox-extras \
     8          iproute2 \
     9          iputils
    10  RUN apk --no-cache --update add build-base
    11  RUN apk add git
    12  RUN git clone https://github.com/vanhauser-thc/thc-hydra
    13  RUN apk add libssh-dev
    14  RUN cd /thc-hydra/ && ./configure
    15  RUN cd /thc-hydra/ && make
    16  RUN cd /thc-hydra/ && make install
    17  COPY users.txt /
    18  COPY password.txt /
    19  COPY dic_attack_ssh.sh /
    20  CMD ["/bin/sh","-c","/dic_attack_ssh.sh"]
```

Fonte: Elaborado pelo autor.

O arquivo do Dockerfile serve apenas para instalar as dependências e a instalação completa do *THC Hydra*. Também copiamos do mesmo diretório os arquivos de texto onde estão alocados os logins e senhas para a ferramenta realizar a combinação. No ato da criação da imagem, é passado o *script* do ataque responsável por atacar o servidor. A Figura 12 mostra o *script* do ataque de dicionário.

Figura 12 – Script do ataque de dicionário.

```
rafael@CyRM:~/cyrm/images/dic_ssh_image$ cat -n dic_attack_ssh.sh
     1  #!/bin/bash
     2
     3  attack_login=users.txt
     4  attack_password=password.txt
     5  TARGET=$TARGET_IP
     6  attack_protocol=ssh
     7
     8  while true
     9  do
    10  while true; do nc -zv ${TARGET} 22 > /dev/null && break; done
    11  hydra -fV -L ${attack_login} -P ${attack_password} ${TARGET} ${attack_protocol}
    12  done
```

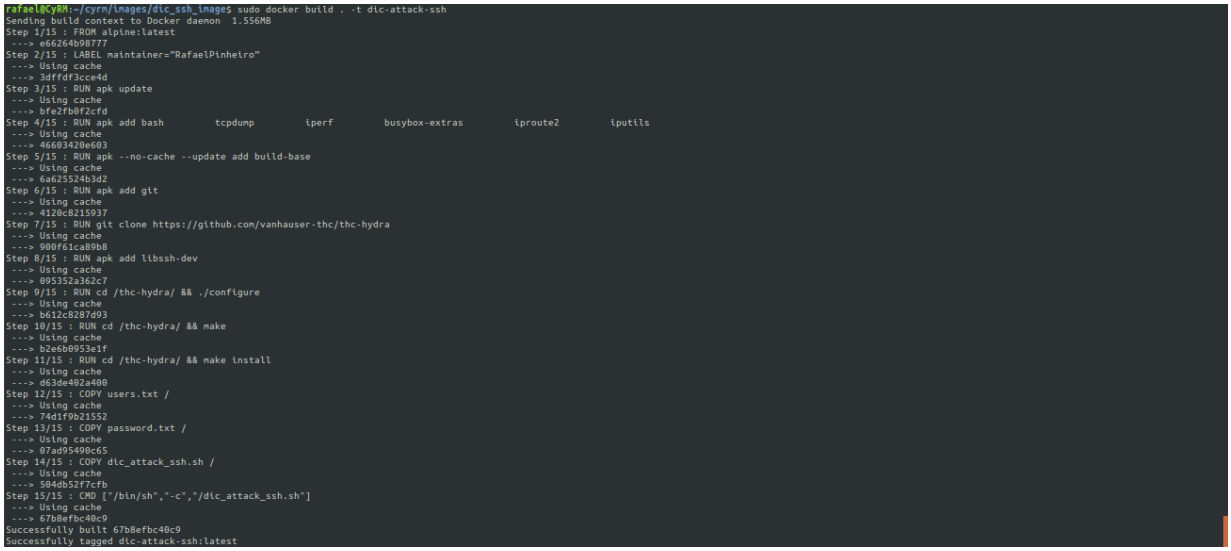
Fonte: Elaborado pelo autor.

O *script* serviu para disparar o ataque automaticamente. Na terceira linha, a variável *attack_login* recebeu o arquivo de texto contendo *wordlist* de usuários. Na quarta linha, a variável *attack_password* recebeu o arquivo de texto que era referente a senhas. Na quinta

[4] https://github.com/vanhauser-thc/thc-hydra

linha, o *TARGET* recebeu o endereço *IP*, que desejávamos atacar, pois definimos no container e possuíamos total liberdade de escolha. Na sexta linha, attack_protocol recebeu o protocolo no qual a ferramenta realizou o ataque; o primeiro *while* foi um *loop* responsável por manter o ataque e o segundo *while* foi usado quando o servidor mudou a porta 22 para que o container não caísse, testando a conexão com o *netcat*. Finalmente, a linha onze foi onde a ferramenta realizou o ataque. A Figura 13 ilustra o processo para a criação da imagem de ataque de dicionário.

Figura 13 – Comando para a criação da imagem do ataque de dicionário.



Fonte: Elaborado pelo autor.

4.3.3 Imagem do Blue Team

A imagem do *blue team* foi construída contendo as ferramentas apropriadas para a execução do roteiro de estudo. *Softwares* para viabilizar a conexão com servidores, analisadores de tráfegos, identificador de endereço, etc. A Figura 14 ilustra o Dockerfile desta imagem. A imagem base é *Ubuntu 22.04*. Um dos *softwares* instalados foi o *bash*, para que a imagem funcionasse em nossa topologia utilizando o *containernet*. Para conectar ao servidor Samba foi instalado *smbclient*, com essa ferramenta o suporte pode acessar o servidor de diretórios públicos e diretórios destinados ao próprio usuário do *blue team*.

Para o acesso remoto ao servidor *Web*, instalamos o *SSH Client*. Portanto, o suporte poderia fazer qualquer operação no servidor de forma remota. Para a captura e análise de tráfego, instalamos o *tshark*. Esta ferramente é baseado no *wireshark*, porém com a diferença que não possui interface gráfica, o que é ideal para nossa ferramenta, já que toda a usabilidade é por terminal. O suporte precisava verificar a origem do ataque ocorrido, para essa situação instalamos

o *geoiplookup* e o banco de dados possuindo todas as faixas de endereço *IP* existentes. Esse *software* não precisa de conexão com a Internet.

Figura 14 – *Dockerfile* do *Blue Team*.

```
rafael@CyRM:~/cyrm/images/blue_team_image$ cat -n Dockerfile
     1  FROM ubuntu:latest
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apt update -y
     4  RUN apt install -y \
     5          bash \
     6          net-tools \
     7          iputils-ping \
     8          iproute2 \
     9          smbclient \
    10          tshark \
    11          ssh
    12  RUN apt-get install geoip-bin geoip-database -y
    13  CMD ["/bin/sh"]
```

Fonte: Elaborado pelo autor.

Criamos um diretório destinado para o desenvolvimento da imagem, possuindo um arquivo, cujo nome é *Dockerfile* que possui a finalidade de instalar programas, pacotes, formas de chamar o *container* e a imagem base que foi executada. Após o arquivo com todas as dependências, usamos o comando do *Docker* para criar a imagem que necessitávamos e definimos o nome que desejávamos. A Figura 15 demonstra o comando para a criação da imagem do *blue team*.

Figura 15 – Comando para a criação da imagem do *blue team*.

```
rafael@CyRM:~/cyrm/images/blue_team_image$ sudo docker build . -t blue-team
Sending build context to Docker daemon  2.048kB
Step 1/6 : FROM ubuntu:latest
 ---> 27941809078c
Step 2/6 : LABEL maintainer="RafaelPinheiro"
 ---> Using cache
 ---> 65c6ebc8d3b8
Step 3/6 : RUN apt update -y
 ---> Using cache
 ---> 6ac7806de917
Step 4/6 : RUN apt install -y   bash    net-tools       iputils-ping    iproute2        smbclient       tshark  ssh
 ---> Using cache
 ---> ea85e03221a7
Step 5/6 : RUN apt-get install geoip-bin geoip-database -y
 ---> Using cache
 ---> 0dddac98b4df
Step 6/6 : CMD ["/bin/sh"]
 ---> Using cache
 ---> 6d528750a9e7
Successfully built 6d528750a9e7
Successfully tagged blue-team:latest
```

Fonte: Elaborado pelo autor.

4.3.4 Imagem do Servidor Samba

Para o servidor Samba, utilizamos uma imagem pronta do *Docker Hub*[5], contendo todas as configurações que precisávamos e que possui a facilidade em determinar a criação de usuários, diretórios para compartilhamento de arquivos etc. Está disponível no repositório público do *Docker Hub*.

[5] https://hub.docker.com/r/dperson/samba

4.3.5 Imagem do Servidor Web

A imagem do servidor *Web* foi criada para simular uma aplicação acessível externamente. O ataque que esse servidor sofreu foi um ataque de dicionário no serviço Secure Shell (SSH) (YLONEN; LONVICK, 2006). Portanto, essa imagem foi construída para rodar apenas um serviço *OpenSSH*[6]. A Figura 16 apresenta o *Dockerfile* da imagem do servidor *Web*. Utilizamos o *Alpine Linux* como imagem base. Assim como nas demais imagens, instalamos o *bash*, pois é requisito para ser executado na topologia do *Containernet*. No *Dockerfile* foram instalados todas as dependências e realizadas as configurações do serviço *OpenSSH*. Dentre as configurações, habilitamos a conexão *SSH* com o uso de senhas e o acesso com usuário *root*. Criamos dois usuários, um com nome *root* e outro com nome *user*, definindo senhas para ambos. Definição da porta 22 do *container* referente ao serviço. Adicionalmente, foi copiado um arquivo de texto com senha difícil de acertar e *script* para inicialização do servidor. Por fim, foi instalado o *syslog-ng* que é um *software* responsável por guardar todos os *logs* de acesso no servidor e possui um grande detalhamento de informações.

Figura 16 – *Dockerfile* do servidor *Web*.

```
rafael@CyRM:~/cyrm/images/ssh_image$ cat -n Dockerfile
     1  FROM alpine:latest
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apk add --update --no-cache openssh \
     4          bash \
     5          tcpdump \
     6          iperf \
     7          busybox-extras \
     8          iproute2 \
     9          iputils
    10  RUN echo 'PasswordAuthentication yes' >> /etc/ssh/sshd_config
    11  RUN echo 'PermitRootLogin yes' >> /etc/ssh/sshd_config
    12  RUN adduser -h /home/user -s /bin/sh -D user
    13  RUN echo -n 'user:123456' | chpasswd
    14  RUN echo -n 'root:123456' | chpasswd
    15  RUN apk add --no-cache syslog-ng
    16  RUN /usr/bin/ssh-keygen -A
    17  RUN ssh-keygen -t rsa -b 4096 -f /etc/ssh/ssh_host_key
    18  EXPOSE 22
    19  COPY pass.txt /root/
    20  COPY start.sh /
    21  CMD ["/bin/sh","-c","/start.sh"]
```

Fonte: Elaborado pelo autor.

Ao criar um *contêiner*, a partir desta imagem, um *script* (*start.sh*) de inicialização será executado automaticamente. A Figura 17 ilustra o código do *script start.sh*. Ele é responsável por inicializar os serviços do *sshd* e *syslog-ng*, sendo escrito em *shell script*.

Figura 17 – *Script* de inicialização servidor *Web*.

```
rafael@CyRM:~/cyrm/images/ssh_image$ cat -n start.sh
     1  !/bin/sh
     2
     3  /usr/sbin/syslog-ng -F -f /etc/syslog-ng/syslog-ng.conf &
     4  sleep 5
     5  /usr/sbin/sshd -D 2>&1 &
     6  /bin/bash
```

Fonte: Elaborado pelo autor.

[6] https://www.openssh.com/

4.3.6 Imagem dos Usuários não Atacantes

A imagem dos usuários não atacantes é simples, possuindo pouca ferramenta instalada. Como imagem base, foi utilizado o sistema operacional *Alpine Linux* com o propósito de pouco armazenamento, segurança e simplicidade em utilizar. Para que a imagem fosse bem sucedida na execução da topologia, o *Containernet* precisava que as imagens possuíssem o *bash* instalado. A Figura 18 ilustra o *Dockerfile* da imagem dos usuários.

Figura 18 – *Dockerfile* dos usuários não atacantes.

```
rafael@CyRM:~/cyrm/images/user_image$ cat -n Dockerfile
     1  FROM alpine:latest
     2  LABEL maintainer="RafaelPinheiro"
     3  RUN apk update
     4  RUN apk add bash \
     5          tcpdump \
     6          iperf \
     7          busybox-extras \
     8          iproute2 \
     9          iputils
    10
    11  CMD ["/bin/sh"]
```

Fonte: Elaborado pelo autor.

4.3.7 Inicialização da Topologia

Após ter criado todos os *scripts* de configuração e ataque das respectivas imagens, juntamente com o *Dockerfile* em seus diretórios de cada imagem, criamos um diretório chamado de *topology* e dentro dessa pasta criamos um *script* em *Python*, chamado *scenario-1.py*. Esse arquivo é responsável pela a execução de todo o cenário (descrito na Seção 4.2) junto ao ***containernet_cyrm***. Os *Contêineres Docker* são *estações* de trabalho, servidores e atacantes. Adicionalmente, *switches* e roteadores também foram instanciados para interconectar tais contêineres.

Para a preparação do ambiente e execução do cenário, desenvolvemos um *script* escrito em *shell script* (*environment_preparation.sh*) com o propósito de automatizar todo o processo de instalação de dependências e construção das imagens *dockers*. A Figura 19 ilustra o *script* de preparação do cenário. Como toda topologia foi executada em um *contêiner* baseado no *Containernet*, precisamos apenas instalar, caso fosse necessário, o *Docker* e o *OpenVSwitch* como dependências. Por outro lado, as imagens que foram baixadas ou construídas são as imagens descritas nas subseções anteriores, bem como a imagem do *containernet_cyrm*.

Para inicializar o cenário, temos o *script shell start_cyrm.sh*. Este *script* inicializa um contêiner do *containernet_cyrm*, criando um volume que aponta para a pasta do cenário (*topology*) e alterando o *endpoint* do contêiner para executar o código do cenário: *python3*

Figura 19 – *Script* de preparação do cenário.

```
rafael@CyRM:~/cyrm$ cat -n environment_preparation.sh
     1  #!/bin/bash
     2
     3  apt update
     4  apt install curl openvswitch-switch -y
     5
     6  docker_check=$(dpkg -l | grep -i docker)
     7  if [ -n "$docker_check" ]; then
     8      echo "DOCKER INSTALADO!"
     9  else
    10      curl -fsSL https://get.docker.com -o get-docker.sh
    11      sh get-docker.sh
    12      echo "DOCKER INSTALADO COM SUCESSO!!!"
    13  fi
    14
    15  docker build $(pwd)/images/blue_team_image/ -t blue-team
    16  docker build $(pwd)/images/ssh_image/ -t web-server
    17  docker build $(pwd)/images/user_image/ -t alpine-user
    18  docker build $(pwd)/images/ddos_image/ -t ddos-attack
    19  docker build $(pwd)/images/dic_ssh_image/ -t dic-attack-ssh
    20  docker build $(pwd)/images/containernet_cyrm_image/ -t containernet-cyrm
    21
    22  docker pull dperson/samba
```

Fonte: Elaborado pelo autor.

/cyrm/scenario-1.py.

Para encerrar a topologia, basta inserir o comando *exit* no terminal do *containernet*, caso haja travamento pressione as teclas *ctrl+c* para retornar ao terminal *linux*. Em seguida, o usuário deve limpar as configurações feitas pelo cenário. Para tanto, basta executar o *script shell clear_cyrm.sh*. Vale ressatar que, tanto o *start_cyrm.sh* como o *clear_cyrm.sh* devem ser executados com o usuário *root* ou com o *sudo*.

A ferramenta está disponível no ***GitHub***[7]. Adicionalmente, utilizaremos esta plataforma para documentar todo o roteiro necessário para orientar a condução do treinamento, para o *blue team*. O objetivo é que a nossa solução possa ser utilizada por outras instituições.

4.4 Roteiro de Estudo

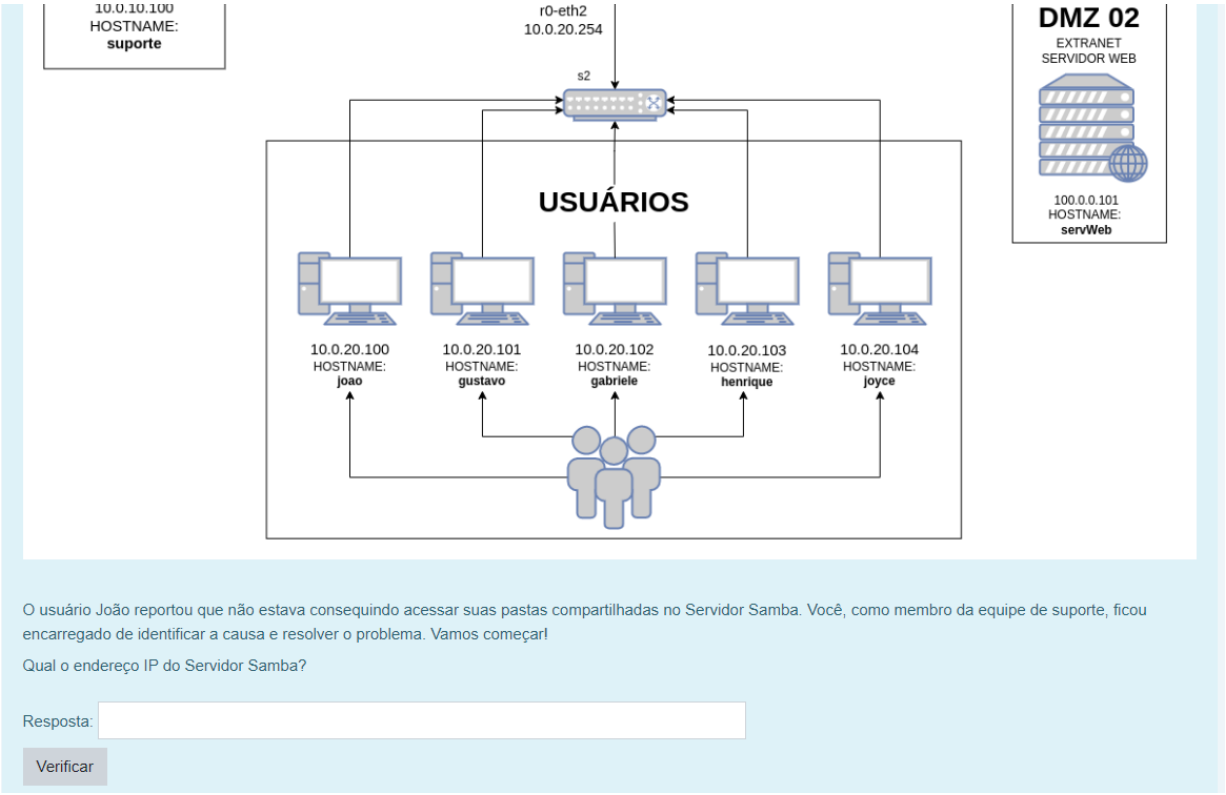
Definimos o roteiro do treinamento com um Questionário (Apêndice A) no *Moodle* do Campus da UFC em Quixadá. A Figura 20 ilustra uma das questões. Este questionário considerou os aspectos de detectar, diagnosticar e mitigar os ataques que ocorrem no cenário. Para tanto, os alvos foram os dois servidores: *Samba* e *Web*. O questionário foi configurado da seguinte forma:

- Mostrar uma questão por vez;
- Navegação sequencial, ou seja, uma vez que se passa para próxima pergunta não é possível retornar;
- Cada questão mostra a topologia do cenário e permite múltiplas tentativas de resposta, onde cada tentativa apresenta uma dica que aproxima o usuário da resposta correta. O objetivo é treinar, não avaliar.

[7] https://github.com/rafaelpinheiroo1/cyrm

A seguir, descreveremos um breve relato do roteiro.

Figura 20 – Questão do roteiro.



Fonte: Elaborado pelo autor.

No servidor Samba ocorreu o ataque de *DDoS*, o servidor ficou inacessível. O usuário João relatou que não estava conseguindo acessar o servidor de compartilhamento e solicitou ao suporte para verificar a causa. O aluno, responsável pelo suporte, tinha a tarefa de detectar o que estava acontecendo. Testou a conectividade utilizando o comando *ping* e o servidor estava acessível. Porém, o suporte não conseguiu acessar o serviço Samba. Em seguida, acessando diretamente o servidor, ele utilizou o *netstat* para verificar quais as conexões *TCP* o servidor estava recebendo. Neste ponto, foram detectadas conexões com o PC *henrique*.

Diante dessa obeservação, o Aluno deveria acessar o roteador r0 e executar o *Tshark* para capturar o tráfego na interface do roteador conectada a VLAN USUARIOS. Posteriormente, o aluno analisou o tráfego e percebeu que havia muitas requisições do mesmo endereço que tinha sido identificado no *netstat*. Neste processo o aluno diagnosticou que era um ataque, e começou o processo de mitigação do ataque bloqueando o tráfego de entrada no r0 com o *firewall iptables* e desconectando o host final do usuário para que fique impossibilitado de enviar pacotes.

Aparentemente, a mitigação estava completa. Contudo, o aluno detectaria que serviço do samba estava inacessível. Ele então realizaria o mesmo procedimento anterior e

para descobrir que o PC *joyce* também estava realizando o mesmo ataque, realizando assim a mitigação do ataque. Ao verificar novamente a disponibilidade do serviço, o aluno observou que o mesmo estava acessível, ou seja, o problema foi resolvido.

No Servidor Web ocorreu o ataque de dicionário no serviço *SSH*. Portanto, o suporte realizou algumas tarefas para detectar, diagnosticar e mitigar o ataque. Para detectar, utilizou o *netstat* novamente e percebeu um acesso de endereço *IP* estranho que estava conectado ao servidor. Prosseguindo com a detecção, o Aluno visualizou os arquivos de *logs* do servidor e diagnosticou que o mesmo endereço *IP* estava disparando várias tentativas de acesso remoto, utilizando usuários e senhas diferentes. Com isso, o aluno diagnosticou que era um ataque de dicionário ou força bruta. O processo de mitigação foi trocar a porta padrão do *SSH* e mudar a senha fácil do usuário. Além disso, o aluno bloqueou o tráfego proveniente do *IP* em questão no r0, usando *iptables*. Verificou novamente e os ataques não ocorreram mais.

5 EXPERIMENTOS

Neste capítulo, descrevemos o processo de realização dos experimentos e analisaremos os resultados obtidos.

5.1 Descrição do Experimento

Nesta etapa, realizamos um teste de usabilidade da prova de conceito com os alunos da disciplina de Tópicos Avançados em Redes de Computadores, do curso de Redes de Computadores do Campus da UFC em Quixadá. Ao todo, 14 alunos participaram. O experimento foi realizado de forma remota pelo o *Google meet*.

Para realização deste experimento, cada aluno criou uma instância na *AWS (Amazon Web Services)*, com a seguinte configuração:

- Sistema Operacional: Ubuntu Server 20.04;
- Tipo de instância: t2.large (2 vCPUs e 8GB de RAM);
- Armazenamento: 30 GB.

Em seguida, eles clonaram o repositório *GitHub* da prova de conceito e rodaram o *script* para preparar o ambiente e, em seguida, o *script* para inicializar o ambiente. No início, fizemos uma breve descrição da atividade e do funcionamento da ferramente, e tiramos as dúvidas durante a realização da atividade. Foi criada uma disciplina no *Moodle* do Campus, exclusivamente para rodar o questionário da ferramenta, os alunos foram matriculados e respondiam às questões em paralelo com a interação com o cenário.

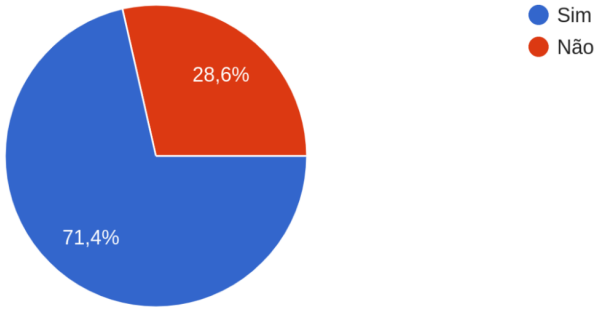
Ao final da atividade, os alunos participantes responderam um questionário de avaliação no *Google Forms*, como forma de prover um *feedback* sobre a experiência de uso, bem como possíveis bugs encontrados. O quadro 2 mostra as perguntas do *Google Forms* que os alunos responderam. Ao todo, os alunos tiveram que responder 16 perguntas de avaliação.

5.2 Avaliação dos Resultados

Nesta seção, analisamos os resultados obtidos a partir do preenchimento do questionário de avaliação. Vale ressaltar que, todos os alunos participantes (14 no total) responderam ao questionário.

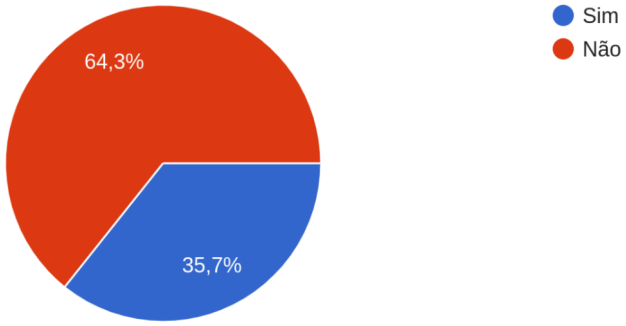
As **Questões 5, 15 e 16** tinham como objetivo avaliar as condições de execução do CyRM durante a atividade. Os resultados podem ser visualizados nas Figuras 21, 22 e 23,

Figura 21 – **Questão 5**: Houve lentidão/travamento na máquina durante o experimento?



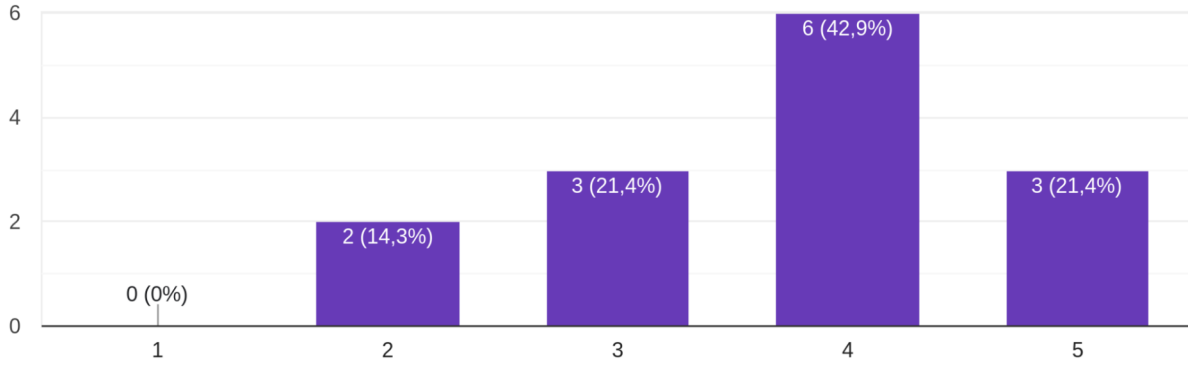
Fonte: Elaborado pelo autor.

Figura 22 – **Questão 15:** Encontrou algum bug?



Fonte: Elaborado pelo autor.

Figura 23 – **Questão 16:** Como você avalia o tempo de resposta das ações do CyRM?

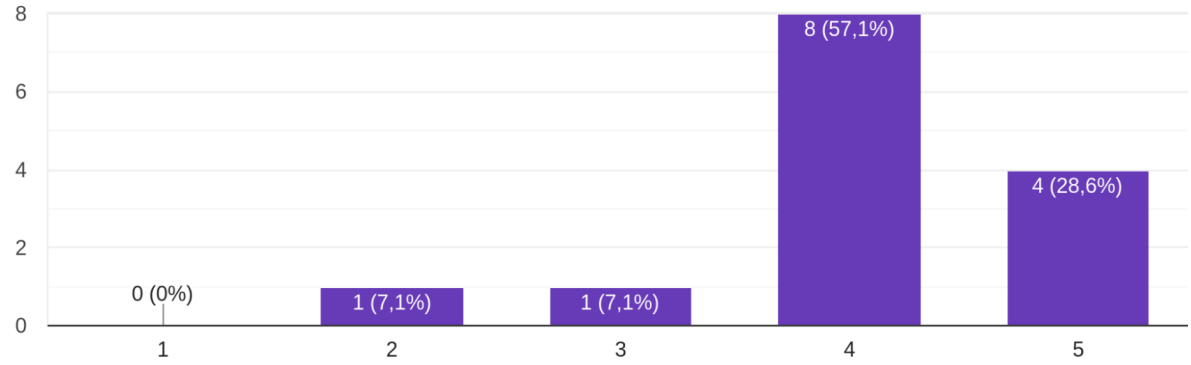


Fonte: Elaborado pelo autor.

respectivamente. O gráfico representado na Figura 21 evidencia que a maioria dos alunos (71,4%) relataram problemas de travamento e/ou lentidão durante a execução da atividade (**Questão 5**). Por outro lado, o gráfico representado na Figura 22 demonstra que a minoria dos alunos (35,7%) relataram que encontraram bugs no CyRM (**Questão 15**). Os resultados da Figura 23 avaliaram a lentidão da ferramenta (**Questão 16**), em uma escala de 1 (péssimo) a 5 (excelente). Nesse

caso, a maioria dos alunos (64,3%) reportou que CyRM teve um desempenho bom (nota 3) e excelente (nota 4). Finalmente, vale ressaltar que, mesmo com alguns pontos negativos, estas métrica não parecem influenciar no nível de satisfação da ferramenta (Figura 33).

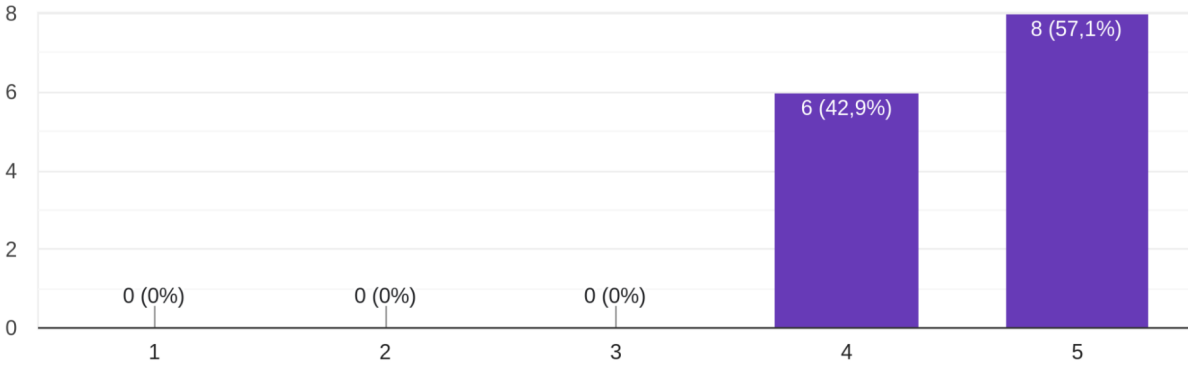
Figura 24 – **Questão 6**: As questões estavam de fácil compreensão?



Fonte: Elaborado pelo autor.

A Figura 24 ilustra os resultados da **Questão 6** onde, na escala, nota 5 significa excelente e nota 1 significa péssimo. Em relação ao nível de compreensão das questões, tivemos um resultado satisfatório com a ferramenta, com a maioria dos alunos dando nota 4 e 5 (85,7% no total). Contudo, vale mencionar que os dois alunos que deram notas mais baixas (2 e 3) já cursaram ou estão cursando a disciplina de Segurança da Informação ou Segurança em Redes. Neste caso, delimitamos que um possível ponto de melhoria para a ferramenta seja uma revisão no texto das questões do questionário, de forma a deixá-las mais esclarecedoras.

Figura 25 – **Questão 7**: As questões conseguiram lhe guiar em direção a resposta correta?

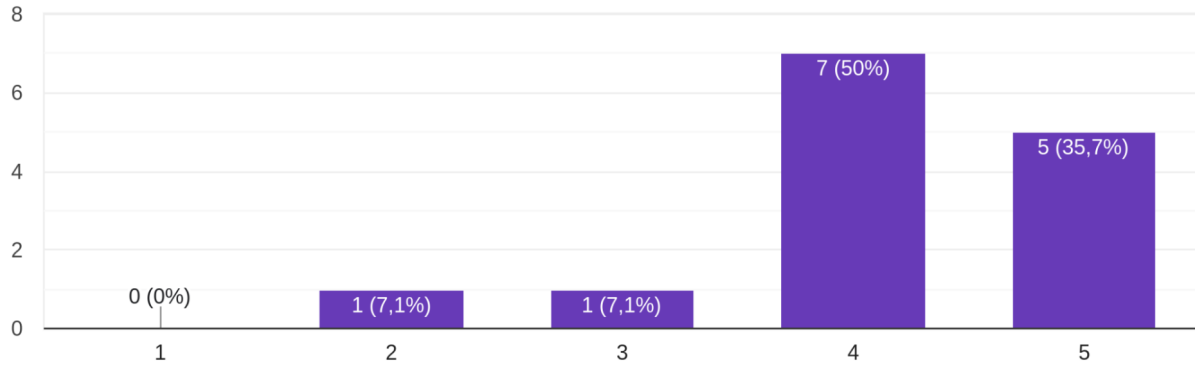


Fonte: Elaborado pelo autor.

A Figura 25 ilustra os resultados da **Questão 7** onde, na escala, nota 5 significa excelente e nota 1 significa péssimo. Tivemos um resultado satisfatório com a ferramenta, com

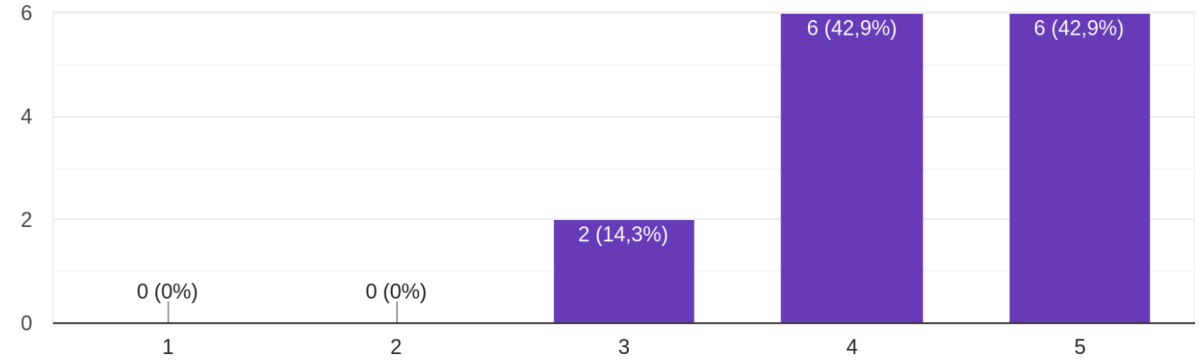
100% dos alunos dando nota 4 e 5. Tais ferramentas evidenciam que o questionário alcança o seu objetivo, que consiste em prover treinamento para os alunos, não avaliá-los.

Figura 26 – **Questão 8**: Em que escala está a sua satisfação em realizar um experimento teórico e pratico em paralelo?



Fonte: Elaborado pelo autor.

Figura 27 – **Questão 9**: Em que escala o CyRM conseguiu instruir, testar ou aperfeiçoar os conhecimentos a respeito da detecção, diagnóstico e mitigação de ataques?

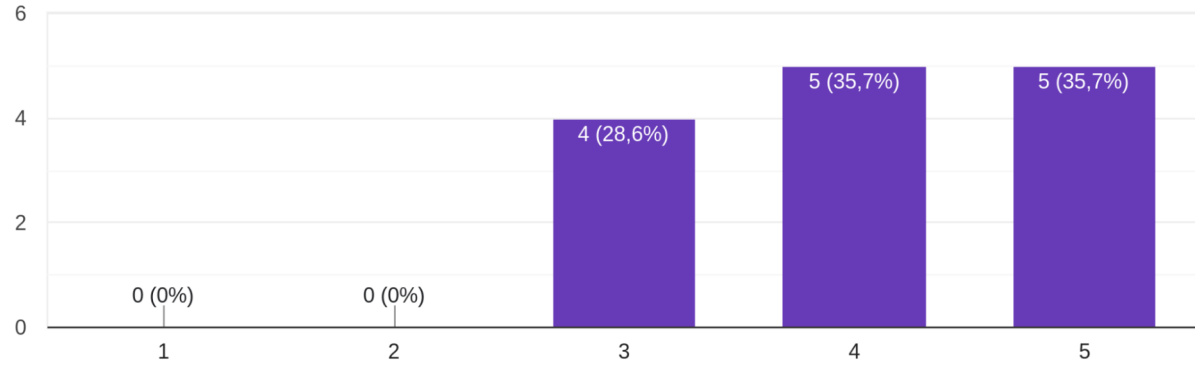


Fonte: Elaborado pelo autor.

As **Questões 8, 9, 10 e 14** tinham como objetivo avaliar o impacto da experiência de uso da ferramenta no conhecimento do aluno, em uma escala de 1 (péssimo) a 5 (excelente). Os resultados podem ser visualizados nas Figuras 26, 27, 28 e 29, respectivamente. De modo geral, o resultado foi satisfatório, dado que a maioria dos alunos atribuiu notas 4 e 5 nos quatro gráficos. Dentre os resultados negativos, podemos destacar as quatro notas 3 na Questão 10. Isso se deve principalmente ao fato das questões guiarem o aluno em relação a resposta correta, evitando provocar momentos de reflexão.

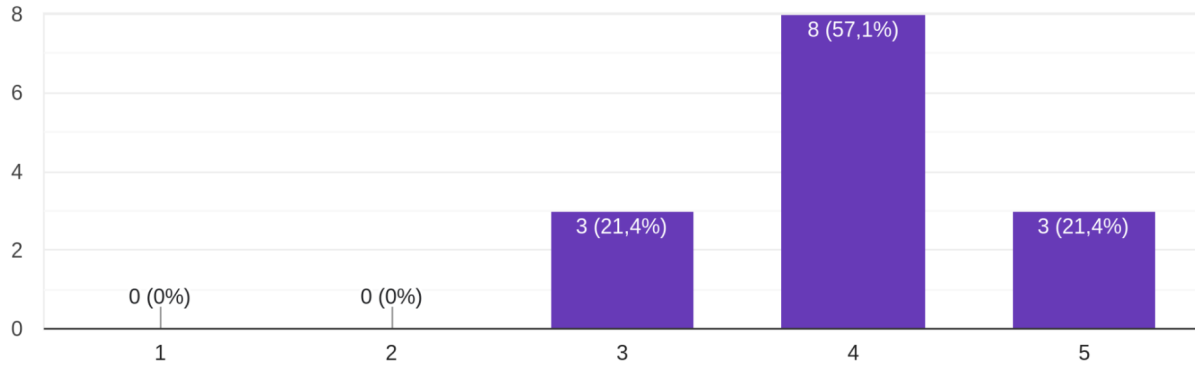
O gráfico representado pela Figura 30 mostra que 92,9% dos alunos responderam positivamente sobre o CyRM conseguir simular atividades realísticas em um ambiente corpora-

Figura 28 – **Questão 10**: Em que escala o CyRM estimula a criatividade do aluno?



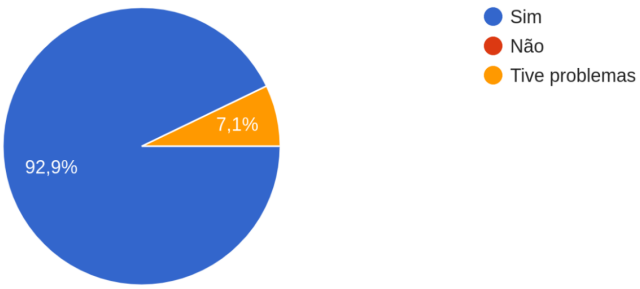
Fonte: Elaborado pelo autor.

Figura 29 – **Questão 14:** Qual o nível de aprendizagem?



Fonte: Elaborado pelo autor.

Figura 30 – **Questão 11:** O CyRM conseguiu simular atividades realísticas em um ambiente corporativo?

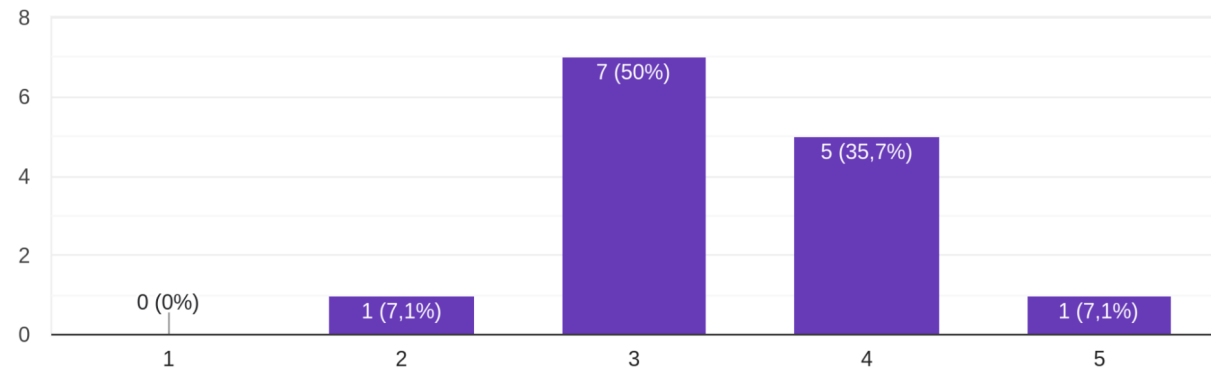


Fonte: Elaborado pelo autor.

tivo (**Questão 11**). Este resultado demonstra que conseguimos alcançar o objetivo de simular uma experiência real em ambiente coorporativo na prova de conceito.

Avaliamos o grau de dificuldade que os alunos tiveram ao usar CyRM, em uma escala de 1 (muito difícil) a 5 (facílimo). O gráfico da Figura 31 ilustra os resultados. De maneira geral, a maioria dos alunos (92,9%) não tiveram dificuldades em utilizar o CyRM, considerando

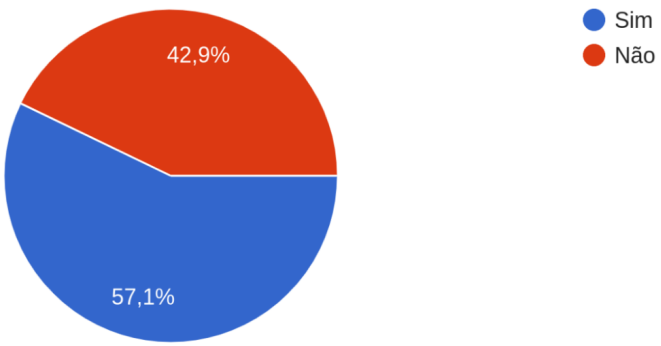
Figura 31 – **Questão 13:** Qual o nível de dificuldade em utilizar o CyRM?



Fonte: Elaborado pelo autor.

a nota 3 como normal.

Figura 32 – **Questão 1:** Já fez ou está fazendo a disciplina de Segurança em Redes ou Segurança da Informação?

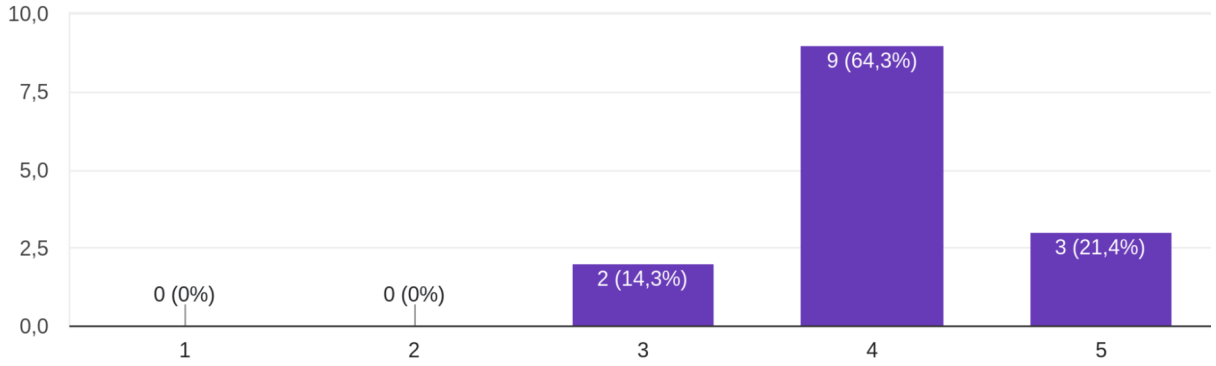


Fonte: Elaborado pelo autor.

No gráfico representado na Figura 32, a maioria dos alunos (57,1%) concluíram ou estão cursando a disciplina de Segurança da Informação ou Segurança em Redes. Esta métrica é importante para avaliar o grau de satisfação (Questão 2) e complexidade (Questão 13) de uso da ferramenta e como as dicas auxiliaram na resolução as questões (Questão 7).

A Figura 33 ilustra os resultados da **Questão 2** onde, na escala, nota 5 significa excelente e nota 1 significa péssimo. Em relação aos resultados, tivemos um resultado satisfatório com a ferramenta, com a maioria dos alunos dando nota 4 e 5 (85,7% no total). Vale ressaltar que, considerando a **Questão 3**, 100% dos alunos conseguiram executar a ferramenta com sucesso, não apresentando quaisquer erros na inicialização. Além disso, considerando a **Questão 4**, 100% dos alunos nunca haviam realizado alguma prática de segurança usando Cyber Range. Finalmente, 85,7% dos alunos conseguiram concluir o experimento com êxito (**Questão 12**), ou

Figura 33 – **Questão 2:** Qual a satisfação com a ferramenta?



Fonte: Elaborado pelo autor.

seja, 2 alunos não conseguiram finalizar a atividade, o que pode ter gerado certa insatisfação.

Quadro 2 – Perguntas aplicadas no questionário de avaliação.

Número	**Questões**
1	Já fez ou está fazendo a disciplina de Segurança em Redes ou Segurança da Informação?
2	Qual a satisfação com a ferramenta?
3	Conseguiu executar o CyRM?
4	Já tinha realizado alguma prática de segurança da informação com Cyber Range?
5	Houve lentidão/travamento na maquina durante o experimento?
6	As questões estavam de fácil compreensão?
7	As questões conseguiram lhe guiar em direção a resposta correta?
8	Em que escala está a sua satisfação em realizar um experimento teórico e pratico em paralelo?
9	Em que escala o CyRM conseguiu instruir, testar ou aperfeiçoar os conhecimentos a respeito da detecção, diagnóstico e mitigação de ataques?
10	Em que escala o CyRM estimula a criatividade do aluno?
11	O CyRM conseguiu simular atividades realísticas em um ambiente corporativo?
12	Concluiu o experimento com êxito?
13	Qual o nível de dificuldade em utilizar o CyRM?
14	Qual o nível de aprendizagem?
15	Encontrou algum bug?
16	Como você avalia o tempo de resposta das ações no CyRM?

Fonte: elaborado pelo autor (2022).

5.3 Discussão Final

Avaliamos os resultados dos gráficos que os alunos responderam. Observamos que 57,1% dos alunos cursaram ou estão cursando a disciplina de Segurança da Informação e, para a realização do treinamento é interessante que mais de 50% tenha um breve conhecimento na área, este ponto foi satisfatório. Outro ponto importante é que mesmo que nenhum dos envolvidos tivessem realizado alguma prática de Segurança da informação com *Cyber Range*, a grande maioria conseguiu executar o CyRM com êxito.

Ainda de acordo com as análises, muitos alunos relataram que houve lentidão e/ou travamento durante o experimento, mas esses obstáculos não impossibilitaram o processo. As questões foram considerados de fácil compreensão e excelentes guias para a resolução correta da

atividade. Os participantes também se mostraram satisfeitos em realizar o experimento teórico e prático simultaneamente.

Como o CyRM provê treinamento para os alunos, ele não estimula muito a criatividade do aluno, pois consegue orientar melhores estratégias para resolver as questões, sem provocar um momento de reflexão por parte do aluno. Além disso, o CyRM simula as operações reais de um ambiente corporativo juntamente com a topologia desenvolvida.

Os estudantes encontraram erros durante a execução do experimento, tais como a topologia do *containernet* que se desconectou algumas vezes. Para resolver esse problema foi necessário reinicializar a topologia, sem causar prejuízos maiores a atividade. Além disso, *backspace*, o autocompletar e a funcionalidade das setas não funcionavam enquanto a interação com os nós acontecia no CLI do *Containernet*. Esses *bugs* são esperados por se tratar de uma fase inicial de teste, e serão incluídos como trabalhos futuros para que não ocorram em experimentos posteriores. Quanto às melhorias na ferramenta, 35,7% dos alunos desejam que os comandos ao serem inseridos no terminal da topologia, se autocompletem, que regularize a funcionalidade do *backspace* e que a utilização da seta funcione.

A partir da análise dos dados coletados, podemos constatar que o CyRM obteve um bom índice de satisfação, tendo em vista que é um experimento precursor em *Cyber Range (CR)* no Campus da UFC em Quixadá, com questões guiadas em ambiente emulado com todos os serviços, componentes finais simulados e cenário realistíco de um ambiente corporativo.

6 CONCLUSÕES E TRABALHOS FUTUROS

Esse trabalho desenvolveu uma ferramenta de *Cyber Range* para auxiliar no ensino de defesa para alunos da disciplina de Segurança da Informação, da UFQUIXADA, pois identificamos a necessidade de uma plataforma que atendesse a realidade da instituição, considerando a falta de treinamento adequado nessa especialidade e a escassez de recursos de processamento e memória dos equipamentos da faculdade. A partir das pesquisas realizadas, utilizamos duas ferramentas como base para o desenvolvimento do trabalho: *Docker* e *Containernet*, visto que o *Containernet* pode ser usado para criar uma solução leve de *Cyber Range* por habilitar a programação de uma topologia de rede programável e leve e se beneficiar da versatilidade do *Docker*, possibilitando a criação de diversos cenários de defesa.

Para atingir uma compreensão do objetivo geral que é propor e avaliar uma solução de *Cyber Range* para auxiliar no ensino de defesa para alunos da disciplina de Segurança da Informação do Campus da UFC em Quixadá, foram definidos três objetivos específicos. O primeiro consiste na definição da arquitetura da plataforma de *Cyber Range* a partir de soluções já existentes e identificação de ferramentas aplicáveis. Verificou-se a denominação atualizada de um ataque cibernético, bem como a definição da ferramenta *Cyber Range*. Além disso, identificamos que as ferramentas adequadas ao nosso propósito são o *Docker* e o *Containernet*, por necessitarem de poucos recursos computacionais como memória e processamento, para funcionar.

Depois, implementar a solução de *Cyber Range* guiado por um caso de uso. Foi desenvolvida a plataforma CyRM para o treinamento de segurança, com ênfase na área defensiva e realizado um experimento com os alunos da disciplina de Segurança da Informação da UFQUIXADA. Constatou-se que todos os componentes implementados funcionaram, dessa forma, a plataforma pode ser usada para fins educacionais. E por fim, avaliar a solução proposta na turma da disciplina de Segurança da Informação do curso de Redes de Computadores da UFQUIXADA. A análise permitiu concluir que apesar de necessitar de algumas melhorias, a plataforma cumpriu sua função e foi considerada satisfatória pelos participantes do experimento.

Com isso, a hipótese do trabalho de que é possível desenvolver uma plataforma de *Cyber Range* utilizando *Docker containers* e o emulador *Containernet* se confirmou, porque criamos a topologia personalizada de acordo com os requisitos necessários para realizar o ambiente e o desenvolvimento das imagens para nosso propósito. Sendo assim, a plataforma é gratuita e está disponível no *GitHub* para alunos e professores utilizarem. Também consome

poucos recursos computacionais.

Em pesquisas futuras, pode-se investir em melhorias para solucionar os problemas que ocorreram durante o experimento, utilizando o *Vagrant* para permitir o uso do *xterm* nos *containers*. Também é possível criar novos ataques, servidores, serviços e ampliar a topologia, destinando vulnerabilidades novas no ambiente corporativo. Além disso, pode-se desenvolver outras topologias sendo *Wide Area Network (WAN)* e ambiente de *Internet of Things (IoT)*. Outra sugestão é desenvolver o experimento com foco em *Red Team* com a responsabilidade de procurar vulnerabilidades e atacar.

REFERÊNCIAS

ANTONIOLI, D.; GHAEINI, H. R.; ADEPU, S.; OCHOA, M.; TIPPENHAUER, N. O. Gamifying ics security training and research: Design, implementation, and results of s3. In: **Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy**. New York, NY, USA: Association for Computing Machinery, 2017. (CPS '17), p. 93–102. ISBN 9781450353946. Disponível em: https://doi.org/10.1145/3140241.3140253. Acesso em: 08 maio. 2022.

HANDIGOL, N.; HELLER, B.; JEYAKUMAR, V.; LANTZ, B.; MCKEOWN, N. Reproducible network experiments using container-based emulation. In: **Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies**. New York, NY, USA: Association for Computing Machinery, 2012. (CoNEXT '12), p. 253–264. ISBN 9781450317757. Disponível em: https://doi.org/10.1145/2413176.2413206. Acesso em: 04 jun. 2022.

Karjalainen, M.; Kokkonen, T. Comprehensive cyber arena; the next generation cyber range. In: **2020 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)**. [*S. l.*: *s. n.*], 2020. p. 11–16.

LEITNER, M.; FRANK, M.; HOTWAGNER, W.; LANGNER, G.; MAURHART, O.; PAHI, T.; REUTER, L.; SKOPIK, F.; SMITH, P.; WARUM, M. Ait cyber range: Flexible cyber security environment for exercises, training and research. In: **Proceedings of the European Interdisciplinary Cybersecurity Conference**. New York, NY, USA: Association for Computing Machinery, 2020. (EICC 2020). ISBN 9781450375993. Disponível em: https://doi.org/10.1145/3424954.3424959. Acesso em: 15 fev. 2022.

PEUSTER, M.; KARL, H.; ROSSEM, S. van. Medicine: Rapid prototyping of production-ready network services in multi-pop environments. In: **2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)**. [*S. l.*: *s. n.*], 2016. p. 148–153.

PHAM, C.; TANG, D.; CHINEN, K.-i.; BEURAN, R. Cyris: a cyber range instantiation system for facilitating security training. In: . [*S. l.*: *s. n.*], 2016. p. 251–258.

STALLINGS, W. **Criptografia e segurança de redes**: princípios e práticas. PEARSON BRASIL, 2014. ISBN 9788543005898. Disponível em: https://books.google.com.br/books?id=KO8gvgAACAAJ. Acesso em: 12 out. 2021.

VITALINO, J.; CASTRO, M. **Descomplicando o Docker**. 2018. 2. ed. [S.l.]: BRASPORT. ISBN 9788574529011. Disponível em: https://books.google.com.br/books?id=LjJtDwAAQBAJ. Acesso em: 22 nov. 2021.

YAMIN, M. M.; KATT, B.; GKIOULOS, V. Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. **Computers & Security**, v. 88, p. 101636, 2020. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167404819301804. Acesso em: 05 abr. 2022.

YLONEN, T.; LONVICK, C. **The Secure Shell (SSH) Connection Protocol**. [S.l.]: IETF, 2006. RFC 4254 (Proposed Standard). (Request for Comments, 4254). Disponível em: http://www.ietf.org/rfc/rfc4254.txt. Acesso em: 02 maio. 2022.

APÊNDICE A – QUESTIONÁRIO APLICADO NO TREINAMENTO PARA OS ALUNOS

Q1-SAMBA. O usuário João reportou que não estava conseguindo acessar suas pastas compartilhadas no Servidor Samba. Você, como membro da equipe de suporte, ficou encarregado de identificar a causa e resolver o problema. Vamos começar! Qual o endereço IP do Servidor Samba?

Q2-SAMBA. Suponha que você tenha um usuário no servidor Samba e um diretório remoto, seguem os dados: Usuário: suporte Senha: badpass Diretório remoto: suporte Neste caso, podemos verificar se o serviço do Samba está acessível remotamente. Para tanto, da máquina suporte, basta executar o comando smbclient para tentar acessar o seu diretório remoto: Comando: $ smbclient //10.0.30.100/suporte -U "suporte"Foi possível se conectar ao servidor? Responda sim ou não.

Q3-SAMBA. Utilizando a máquina suporte, com o protocolo Internet Control Management Protocol (ICMP) verifique se a interface do Servidor Samba está alcançável na rede. O servidor está alcançável? Responda "sim"ou "não".

Q4-SAMBA. Agora, a partir do servidor servSamba, liste os processos em execução no sistema. O serviço do Samba está em execução? Responda "sim"ou "não".

Q5-SAMBA. Ainda no servidor servSamba, utilize o comando netstat para verificar se existem conexões TCP ativas com o serviço Samba. Existem conexões com o serviço Samba? Responda "sim"ou "não".

Q6-SAMBA. Ainda no servidor servSamba, através do comando ifconfig, verifique se está chegando muitos ou poucos dados na interface ethernet. Como está o tráfego no servidor? Responda muito tráfego ou pouco tráfego.

Q7-SAMBA. De acordo com a questão anterior, o alto tráfego passante na interface ethernet do servSamba está indisponibilizando o serviço Samba para novos acessos. Isso pode ou não ser classificado como um ataque. Neste caso, precisamos fazer uma análise mais aprofundada. Um próximo passo poderia ser analisar o tráfego no roteador r0. Vamos fazer isso! A ferramenta tshark consiste em uma versão orientada a terminal do Wireshark. Ela tem por objetivo capturar e analisar tráfego da rede. Sabe-se que o roteador r0 possui o tshark instalado. Nesse caso, a partir do r0, utilize o tshark para capturar e salvar (.pcap) o tráfego passante na interface correspondente a VLAN DMZ 01. Execute a captura por 5 segundos e filtre na porta e serviço específico. Para tanto, utilize como base o seguinte comando: $ tshark -i (interface) -w nome_desejado.pcap -a

duration:5 -d tcp.port==445,nbss Além disso, responda abaixo.Qual a interface do r0 onde o tráfego foi capturado?

Q8-SAMBA. Com o arquivo .pcap, gerado na questão anterior, agora precisamos analisar o tráfego para buscar alguma possível anormalidade. Dado que o serviço Samba está inacessível, identifique quais endereços IP estão enviando tráfego para este serviço. Dentre eles, existem endereços IP que estão enviando muito tráfego? Se sim, responda quais são os endereços IP, separados por vírgula. Se não, responda simplesmente "não".

Q9-SAMBA. Com a análise de tráfego da questão anterior, você conseguiu identificar um alto tráfego gerado pelo IP 10.0.30.103, do PC henrique, pertencente a VLAN USUÁRIOS. Além disso, conversando com o usuário deste PC, você descobriu que ele não está acessando o serviço Samba neste momento. Ou seja, provavelmente é um ataque sendo disparado a partir deste PC. Portanto, é necessário realizar a mitigação deste ataque. Inicialmente você poderá realizar o bloqueio deste tráfego no roteador r0. Para tanto, você pode usar o iptables que consiste em um firewall com a funcionalidade de bloquear e liberar o tráfego. Vamos então fazer o bloqueio do tráfego utilizando o seguinte comando: Comando: $ iptables -I FORWARD -s ip_atacante -j DROP Verifique se o fluxo diminuiu na interface correspondente a VLAN DMZ 01! Em caso positivo, você teve sucesso no bloqueio! Consegui bloquear o ataque? Responda sim ou não.

Q10-SAMBA. Você já bloqueou o tráfego com o iptables, impossibilitando o atacante de prosseguir com o ataque. Contudo, as conexões TCP previamente estabelecidas ainda se encontram ativas. Nesse caso, é necessário que desconecte o PC henrique do switch que faz parte da VLAN USUÁRIOS. Para realizar esse procedimento puxe o cabo do host que está afetando o serviço. Comando: $ link henrique <switch_name> down É necessário substituir o <switch_name> pelo nome do switch ao qual PC henrique está conectado. Após ter realizado o processo, verifique se ainda está ocorrendo o ataque! Responda se o servidor Samba ainda está sofrendo o ataque do PC henrique e qual o switch ao qual ele estava conectado? Resolva nesse formato (sim, switch_name ou não, switch_name).

Q11-SAMBA. Pronto, aparentemente mitigamos o ataque que está sendo realizado pelo PC henrique. Nesse caso, vamos testar o acesso ao serviços Samba novamente. A partir da máquina suporte, verifique se o diretório remoto do usuário suporte no servidor Samba está acessível via smbclient. Seguem os dados novamente: Usuário: suporte Senha: badpass Diretório remoto: suporte Comando: $smbclient //10.0.30.100/suporte -U "suporte"Consegue

obter acesso? Responda com "sim"ou "não".

Q12-SAMBA. Na questão anterior ainda não foi possível o PC suporte obter acesso ao serviço Samba. Nesse caso, provavelmente qualquer outro usuário também não está conseguindo! Como isso é possível? Você como suporte realizou várias atividades até encontrar o atacante e fez a mitigação do ataque. Que problemão! Após um período de reflexão, você delimitou que uma das possibilidades seria que servSamba ainda está sendo atacado, agora por outro dispositivo. Para verificar, realize o mesmo procedimento que fez com o PC henrique. A sequência de procedimentos está descrita logo abaixo! Capture e analise o tráfego novamente. Caso seja preciso, filtre para verificar se há um novo atacante. Em caso positivo, faço o bloqueio do tráfego de origem e desconecte o atacante da rede. Finalmente, tente conectar ao servSamba novamente para verificar se possui acesso!

1º) Use o seguinte comando para capturar o tráfego $ tshark -i (interface) -w nome_desejado.pcap -a duration:5 -d tcp.port==445,nbss 2º) Use o seguinte comando ler o tráfego capturado $ tshark -r nome_desejado.pcap 3º) Filtre o ip do atacante $ tshark -r nome_desejado.pcap | cut -d -f 4 | sort | uniq -c | sort -unr 4º) Use o seguinte comando para bloquear o tráfego $ iptables -I FORWARD -s ip_atacante -j DROP 5º) Use o seguinte comando para desconectar do switch: $ link hostname switch_name down 6º) Conectar ao servidor samba via smbclient com a máquina do suporte $ smbclient //10.0.30.100/suporte -U "suporte"Usuário: suporte Senha: badpass Diretório remoto: suporte

Conseguiu identificar algum novo atacante? Se sim, qual o hostname do atacante? Após a execução dos procedimento de mitigação, foi possível acessar smbclient via PC suporte? Responda da seguinte forma: Se tiver atacante: "sim, <hostname>, simsim, <hostname>, não"Se não tiver atacante "não"Onde <hostname> deve ser substituído pelo hostname do atacante.

Q13-SAMBA. Após várias análises, você mitigou os ataques que estavam indisponibilizando o serviço Samba. Agora este serviço está normalizado! Quais eram os endereços dos atacantes? Responda exatamente nesse formato “<IP do primeiro atacante> e <IP do segundo atacante>”.

Q14-SAMBA. Os ataques foram originados na rede externa ou na rede interna da empresa?

Q15-SAMBA. Quais eram os nomes dos hostnames dos atacantes? Responda nesse formato «hostname do atacante 1> e <hostname do atacante 2>".

Q16-SAMBA. No mundo da tecnologia há vários tipos de ataques, cada um com seu

devido propósito. No ataque mitigado, observamos que a indisponibilidade serviço Samba foi provocada por uma inundação de pacotes disparados por dois atacantes. Como é o nome desse ataque? Responda em letra minúscula!

Q17-SAMBA. De modo que seja possível realizar a proteção dos dados contra ameaças internas e externas, é importante que haja a garantia de alguns princípios básicos da segurança da informação. A Confidencialidade que garante que os dados sejam acessíveis e somente pessoas autorizadas possam usufruir desses dados. Integridade é o pilar que garante que a informação trafegada não foi deletada ou corrompida. A Disponibilidade garante que a informação esteja sempre disponível e acessível para os usuários. Há outros que também são muito importantes, como a Autenticidade e a Irretratabilidade (Não-Repúdio). Qual princípio da segurança da informação que o ataque DDoS violou? Digite a resposta em minúsculo.

Q18-WEB. Você, como suporte, analisou o servidor Samba, pois o mesmo estava inacessível, e com essas análises chegou a conclusão que o serviço estava sofrendo um ataque DDoS (Distributed Denial of Service). Em seguida, você realizou a mitigação, bloqueando e desconectando o acesso dos atacantes, solucionando assim o problema! O CEO (Chief Executive Officer) da corporação passou a ficar em alerta com o que aconteceu. Ele então solicitou que você analisasse o Servidor Web, pois a empresa possui muitos dados sigilosos e os cibercriminosos não podem obter essas informações e também o serviço não pode parar, e/ou ser modificado. O objetivo é verificar se o serviço está normal e, caso necessário, melhorar a segurança do ambiente para dificultar a ação de atacantes. Além do serviço Web, sabe-se que este servidor também permite o acesso remoto via Secure Shell (SSH) na porta 22, mas apenas o suporte tem acesso com as seguintes credenciais: Usuário: root Senha: 123456 Então! Vamos iniciar esta nova atividade! Utilizando a máquina suporte, com o protocolo Internet Control Management Protocol (ICMP) verifique se a interface do Servidor Web está alcançável na rede. O servidor está alcançável? Responda "sim"ou "não".

Q19-WEB. Agora, vamos testar se a partir da PC suporte é possível acessar remotamente o servidor servWeb via SSH na porta 22. Você já sabe que as credenciais são as seguintes: Usuário: root Senha: 123456 Qual comando você deve usar? Conseguiu acessar? Responda da seguinte forma: "comando,sim"ou "comando,não".

Q20-WEB. Agora que você verificou que o consegue acessar remotamente o servWeb. Lembre-se: só o suporte está autorizado a acessá-lo. Neste caso, a partir do PC suporte, acesse o servWeb via SSH. Em seguida, utilize o comando netstat para verificar as

conexões TCP ativas. Agora responda os seguintes itens: Existe alguma anormalidade? "sim"ou "não". Em caso positivo, em qual protocolo? Qual o estado das conexões com anormalidade? Qual o IP de origem? è um IP da rede interna ou externa? OBSERVAÇÃO: Responda somente com letras minúsculas. Separar as respostas por vírgula e sem espaço.

Q21-WEB. De acordo com a questão anterior, descobrimos que o IP externo 1.178.218.56 está tentando acessar o serviço SSH no servWeb. Contudo, somente o suporte tem autorização para isso. Portanto, vamos investigar mais a fundo. Vale ressaltar que o serviço SSH está acessível, logo não é ataque DDoS. Contudo, pode ainda ser um ataque. Nos sistemas operacionais Linux existem vários arquivos de logs para determinada funcionalidade, todos armazenados na pasta /var/log. Portanto, o próximo passo seria verificar o arquivo de log do serviço SSH que trata das tentativas de login, pois precisamos verificar se o suposto atacante está testando pares de login/senha aleatórios para acertar as credenciais verdadeiras e usufruir do servidor web. Ainda usando a sessão SSH com o servWeb, leia o arquivo de log que possui essas tentativas de logar no servidor. Qual o nome desse arquivo? Realmente possui múltiplas tentativas de conexão partindo do IP suspeito? Responda neste formato “<nome do arquivo>, sim” ou “<nome do arquivo>, não”.

Q22-WEB. Com a checagem do arquivo de auth.log, você conseguiu diagnosticar o problema: o endereço IP 1.178.218.56 está tentando um acesso não autorizado ao servWeb, através de múltiplas tentativas de conexão. Isso é um ataque! Portanto, é necessário realizar a mitigação deste ataque. Inicialmente, você poderá realizar o bloqueio deste tráfego no roteador r0. Para tanto, você pode usar o iptables, que consiste em um firewall com a funcionalidade de bloquear e liberar o tráfego. Vamos então fazer o bloqueio do tráfego utilizando o seguinte comando: $ iptables -I FORWARD -s ip_atacante -j DROP Verifique o arquivo de log novamente. As tentativas de conexão cessaram? Responda sim ou não.

Q23-WEB. Com o passo realizado na questão anterior, foi possível bloquear o tráfego originado pelo IP do atacante. Contudo, outros ataques podem surgir provenientes de outros IPs. Nesse caso, a nossa atividade de mitigação ainda está incompleta. Um ponto que facilita a execução de ataques em serviços na Internet é o uso da porta padrão. No caso do nosso serviço SSH, estamos usando a porta padrão 22. Portanto, vamos dificultar a vida de futuros atacantes alterando a porta padrão para outra completamente diferente. Seguem os passos necessários que devem ser executados diretamente no servWeb: 1º Liste os processos no servWeb e identifique o PID do serviço SSH $ ps aux Exemplo do pid que é para matar: 19 root 0:00 sshd:

/usr/sbin/sshd -D [listener] 0 of 10-100 startups 2º Matar o processo SSH no servWeb $ kill pid (id do processo) 3º Inicialize o serviço SSH na porta 40157 $ /usr/sbin/sshd -D -p 40157 2>1 & Para testar se realmente trocou a porta corretamente, a partir do PC suporte acesse remotamente o servWeb, utilizando a nova porta. Qual comando você usou para acessar? Conseguiu acessar ("sim"ou "não")? OBSERVAÇÃO: respostas separadas por vírgulas e sem espaço.

Q24-WEB. A cada questão estamos melhorando a segurança do servidor, mas podemos fazer mais. Percebeu que a senha de acesso ao ssh é 123456? Não acha muito fácil? Utilizar senhas mais complexas irá dificultar a atividade de novos atacantes. A Cartilha de Segurança da Internet, no Fascículo Senhas, possui boas práticas para a escolha, uso e armazenamento de senhas de forma segura. Portanto, você deve fazer a troca da senha de acesso. Para tanto, irá atualizar a senha para a que se encontra no arquivo /root/pass.txt do servWeb. Para a realização da troca de senha do usuário root utilize os seguintes comando: $ passwd "usuário"Vai pedir a nova senha (introduza) Vai pedir para repetir Para verificar se o procedimento tes sucesso, teste fazer um acesso remoto do PC suporte ao servWeb, usando a nova senha. Qual a nova senha de acesso do usuário root ao serviço SSH?

Q25-WEB. Com a mitigação completa e as configurações do servidor mais seguras, o CEO da empresa pode ficar mais tranquilo! Agora, sabendo qual é o IP do atacante, vamos verificar de qual país que o ataque se originou. Para tanto, a ferramenta geoiplookup possui a finalidade identificar qual o país de origem. Comando: $ geoiplookup “ip_encontrado” Por tanto utilize essa ferramenta no PC suporte e responda qual o país! Como é nome do lugar, responda a primeira letra em caixa alta. Neste formato "China".

Q26-WEB. Na atualidade da tecnologia que vivemos há vários tipos de ataques, cada um com seu devido propósito. Podendo observar que o atacante utilizou dicionários contendo logins e senhas com o propósito de acertar as credenciais verdadeiras para obter acesso ao serviço e roubar informação, alterar etc. Como é o nome desse ataque? Responda em letra minúscula!

Q27-WEB. De modo que seja possível realizar a proteção dos dados contra ameaças internas e externas, é importante que haja a garantia de alguns princípios básicos da segurança da informação. A Confidencialidade que garante que os dados sejam acessíveis e somente pessoas autorizadas possam usufruir desses dados. Integridade é o pilar que garante que a informação trafegada não foi deletada ou corrompida. A Disponibilidade garante que a informação esteja sempre disponível e acessível para os usuários. Há outros que também são muito importantes, como a Autenticidade e a Irretratabilidade (Não-Repúdio). Qual princípio da segurança da

informação que o ataque de força bruto poderia ter violado, caso tivesse sucesso? Digite a resposta em minúsculo.