Capítulo 5

Testbeds para Pesquisa Experimental em Cibersegurança: Da Teoria à Prática

Michelle Silva Wangham[*‡], Bruno H. Meyer[†], Davi D. Gemmer[*], Khalil G. Q. de Santana[‡], Lucas Rodrigues Frank[§], Luiz Eduardo Folly de Campos[*], Emerson Ribeiro de Mello[¶] e Marcos Felipe Schwarz[*]

Abstract

The current scenario for experimental research in cybersecurity is promising and broad, encompassing various infrastructures and application domains. However, conducting experiments faces significant challenges, such as high costs, operational risks, resource and network management, heterogeneity, capacity and quantity of devices, flexible experiment orchestration, and a large volume of generated experimental data. This chapter aims to present specialized testbeds for conducting cybersecurity experiments, focusing on the MENTORED Testbed, particularly emphasizing the analysis of vulnerabilities, attacks, and defense strategies associated with Internet of Things devices. This testbed is a controlled environment for experimentation and is used as a case study for hands-on demonstrations of theoretical concepts. This chapter describes two DDoS attack experiments, and their results are presented and analyzed.

Resumo

O cenário atual para pesquisa experimental em cibersegurança é promissor e abrangente, englobando diversas infraestrutura e domínios de aplicação. Porém, a condução de experimentos enfrenta uma série de desafios significativos, tais como altos custos, riscos operacionais, gerenciamento de recursos e de redes, heterogeneidade, capacidade

[*]Rede Nacional de Ensino e Pesquisa. Email: michelle.wangham@rnp.br, davi.gemmer@rnp.br, luiz.campos@rnp.br, marcos.schwarz@rnp.br

[†]Universidade Federal do Paraná. Email: bruno.meyer@ufpr.br

[‡]Universidade do Vale do Itajaí. Email: wangham@univali.br, khalil.santana@edu.univali.br

[§]Universidade Federal de Juiz de Fora. Email: lucasrodrigues@ice.ufjf.br

[¶]Instituto Federal de Santa Catarina. Email: mello@ifsc.edu.br

e quantidade de dispositivos, orquestração flexível de experimentos e grande volume de dados experimentais gerado. Este capítulo tem como objetivo apresentar testbeds especializados para a condução de experimentos em cibersegurança, com destaque para o MENTORED Testbed, com um foco particular na análise de vulnerabilidades, ataques e estratégias de defesa associadas aos dispositivos da Internet das Coisas. Este testbed é um ambiente controlado para experimentação e é utilizado como estudo de caso para demonstrações práticas de conceitos teóricos. Dois experimentos com ataques de DDoS são descritos neste capítulo e seus resultados apresentados e analisados.

5.1. Introdução

O relatório de Riscos Globais do *World Economic Forum* 2024 (FORUM, 2024) aponta que os ataques cibernéticos continuam entre os maiores riscos advindos da rápida disseminação da Inteligência Artificial (IA) generativa e de outras novas tecnologias que podem ser facilmente utilizadas em ciberataques, representando uma séria ameaça tanto para as empresas como para a vida pública. Diante da evolução constante das ameaças e dos ataques de segurança, é crucial o desenvolvimento e o aprimoramento contínuo de soluções robustas de cibersegurança.

Desenvolver soluções robustas para prevenir, detectar e mitigar ataques de cibersegurança requer ferramentas e métodos para testá-los e validá-los. Devido aos custos e aos riscos legais, éticos e operacionais, tais como interrupção ou danos às infraestruturas, a realização de experimentações em ambientes reais ou de produção nem sempre são viáveis ou adequadas (SÁEZ-DE-CÁMARA et al., 2023).

De acordo com Goodfellow, Thurlow e Ravi (2018), o cenário atual para experimentação é diverso e abrangente, englobando a proteção de infraestruturas críticas, a segurança de sistemas ciberfísicos, redes celulares, plataformas automotivas, dispositivos IoT e os sistemas de controle industrial. Para pesquisadores e profissionais da área de cibersegurança, a seleção de plataformas adequadas para validar soluções e testar hipóteses de pesquisa é uma tarefa complexa (CHOULIARAS et al., 2021).

Essa busca envolve enfrentar desafios críticos, como a necessidade de encontrar ambientes para experimentação que atendam a requisitos fundamentais, incluindo reprodutibilidade, fidelidade, isolamento, realismo, flexibilidade, escalabilidade e custo (BENZEL, 2011; CHERNYSHEV et al., 2017; PRATES JR et al., 2021).

Na busca por validar sistemas, protocolos e implementações de segurança, os pesquisadores se deparam com a difícil escolha entre simuladores, emuladores e *testbeds*. Embora simuladores e emuladores forneçam ambientes controlados para experimentos reproduzíveis, muitas vezes carecem de realismo. Já os *testbeds* oferecem realismo e escalabilidade, mas podem ser limitados pelo custo e pela disponibilidade de equipamentos reais (GOMEZ et al., 2023). *Testbeds* também enfrentam alguns desafios e nem sempre conseguem reproduzir aspectos de eventos reais, mesmo que tenham a capacidade de reproduzir cenários relevantes. Contudo, a importância dos *testbeds* é refletida pela quantidade abundante de estudos que pesquisam o assunto, além de *datasets* populares obtidos por meio de *testbeds* como o Ton_IoT (MOUSTAFA, 2021), o que será detalhado na Seção 5.2. A impossibilidade de testar soluções em ambientes realistas em larga escala representa uma barreira significativa para empresas, instituições e fornecedores de solu-

ções, resultando em atrasos na implementação de novos recursos e limitando a inovação em cibersegurança.

Este capítulo tem como objetivo apresentar *testbeds* especializados para a condução de experimentos em cibersegurança, com destaque para o MENTORED *Testbed*, desenvolvido no âmbito do MENTORED Project[1], financiado pela FAPESP, e construído sobre o Cluster Nacional da Rede Nacional de Ensino e Pesquisa (RNP). Por fim, com o objetivo demonstrar na prática conceitos teóricos de experimentação em cibersegurança, dois experimentos de ataques de DDoS, utilizando o MENTORED *Testbed*, são descritos e analisados.

O restante deste capítulo está estruturado da seguinte forma. Na Subseção 5.1.1, os desafios para conduzir experimentos em cibersegurança e os requisitos para a concepção de *testbeds* para cibersegurança são analisados. Na Seção 5.2 são apresentados *testbeds* que se destacam na pesquisa experimental em cibersegurança e os compara, considerando suas principais características e limitações. Na Seção 5.3 é descrito o MENTORED *Testbed*, suas principais características e os recursos tecnológicos relevantes que este oferece para os experimentadores. Na Seção 5.4 são apresentados dois casos de uso de ataques DDoS, os roteiros práticos para execução dos experimentos no MENTORED *Testbed*, bem como alguns resultados obtidos. Por fim, na Seção 5.5 são feitas as considerações finais e apresentadas perspectivas futuras.

5.1.1. Desafios para conduzir experimentos em cibersegurança

A realização de experimentos em cibersegurança enfrenta uma série de desafios que dificultam o seu progresso. O gerenciamento de recursos e agendamento em diversas regiões geográficas contribuem para um ambiente complexo, devido à necessidade de manter a comunicação entre os dispositivos de forma sincronizada. Além disso, a instabilidade da rede e o tamanho do sistema podem influenciar na precisão dos resultados, bem como na capacidade de resposta imediata às ameaças (GOMEZ et al., 2023).

Adicionalmente, a capacidade dos dispositivos usados são outro ponto que influencia diretamente na execução dos experimentos que envolvem um número elevado de dispositivos, principalmente por conta dos avanços em Internet das Coisas (IoT) (SÁEZ-DE-CÁMARA et al., 2023). Com capacidades variáveis, os dispositivos com menos recursos ou mais antigos podem não ser capazes de executar tarefas com cargas de trabalho intensas, o que por consequência pode limitar a escala e a complexidade dos experimentos e, desta forma, impossibilitar a simulação de ataques em larga escala de forma adequada (MIRKOVIC; BENZEL, 2012).

Schwab e Kline (2019) destacam outros desafios significativos na área, como a necessidade de sistemas que possam suportar tanto experimentação interativa quanto em lote[2], além de uma orquestração flexível e abrangente para configurar e executar cenários automaticamente. Observa-que, à medida que o tamanho do experimento aumenta, a

[1]https://mentored.dcc.ufmg.br/

[2]Na experimentação interativa, o pesquisador interage diretamente com o sistema ou ferramenta podendo ajustar parâmetros, realizar ações e observar os resultados de imediato. Na experimentação em lote, uma série de testes é automatizada e executada em lote, sem a necessidade de intervenção humana durantes os testes.

complexidade de sua configuração também aumenta, demandando mais tempo para configurar e adquirir recursos. Isso muitas vezes resulta na execução contínua de experimentos para evitar o desperdício de tempo.

Além dos desafios já mencionados, a imensa quantidade de dados gerados durante experimentos em cibersegurança apresenta uma complexidade adicional (SÁEZ-DE-CÁMARA et al., 2023; GOMEZ et al., 2023). Lidar com esse volume de dados requer o emprego técnicas avançadas de análise de dados e inteligência artificial. Essas ferramentas são essenciais para identificar padrões e anomalias, permitindo a detecção e mitigação de ataques em tempo real, enquanto minimiza o impacto nos serviços online.

Para enfrentar esses desafios, diversos pesquisadores e profissionais de cibersegurança estão continuamente desenvolvendo e aprimorando ferramentas e técnicas (GOMEZ et al., 2023). Juntamente a esses avanços, destacam-se o uso de tecnologias de virtualização e computação em nuvem, além do desenvolvimento de algoritmos para análise de grandes conjuntos de dados. Além disso, práticas de automação têm sido implementadas para simplificar o gerenciamento de dispositivos distribuídos. Assim, a colaboração entre instituições acadêmicas e empresas torna-se fundamental para proteger sistemas e dados contra ameaças cibernéticas (MIRKOVIC; BENZEL, 2012).

5.1.2. Requisitos dos *testbeds* de cibersegurança

Segundo Siaterlis, Garcia e Genge (2013), diferentes tecnologias podem ser empregadas para realização de experimentos em cibersegurança, como emuladores, simuladores e ambientes reais. A escolha da tecnologia a ser utilizada depende dos objetivos do experimento, bem como das restrições de recursos, como orçamento, tempo e disponibilidade de equipamentos. Geralmente, os emuladores são mais adequados para experimentos de larga escala, enquanto os simuladores são mais adequados para experimentos de pequena escala. Já os *testbeds* são mais adequados para experimentos que exigem realismo e fidelidade. Em Gomez et al. (2023) é apresentada uma revisão da literatura que descreve a utilização de emuladores, simuladores e ambientes reais (*testbeds*) empregados tanto em pesquisa quanto como ferramenta de ensino na área de redes de computadores.

Os *testbeds* apresentam-se como uma alternativa para a realização de experimentos em cibersegurança, uma vez que seria inadequado realizar experimentos em ambientes de produção ou mesmo replicar cenários reais. Contudo, para que seja possível a execução de experimentos realistas e escaláveis, é necessário que os *testbeds* atendam uma série de requisitos, como indicado por Siaterlis, Genge e Hohenadel (2013), os quais serão detalhados a seguir.

- **Fidelidade** – é desejado que *testbeds* possam replicar com precisão do ambiente real, alvo do estudo, de forma a garantir que os resultados obtidos sejam relevantes e significativos. Nota-se que não é necessário replicar todos os detalhes do ambiente real, mas sim os detalhes essenciais para o estudo em questão. Essa abstração é necessária para garantir que os experimentos sejam executados de forma eficiente, sem comprometer a fidelidade dos resultados;
- **Flexibilidade** – é desejado que o *testbed* possa ser facilmente adaptado para diferentes cenários de experimentos, de forma a permitir que o pesquisador possa

realizar experimentos com diferentes configurações de rede e dispositivos. Assim, o *testbed* deve permitir que o pesquisador defina a topologia da rede, os dispositivos envolvidos, os protocolos de comunicação, entre outros aspectos relevantes para o estudo em questão;

- **Reprodutibilidade** – consiste na capacidade de repetir os experimentos e obter os mesmos resultados ou estatisticamente semelhantes. Para garantir a reprodutibilidade, é necessário que os *testbed* permita ao pesquisador definir os estados inicial e final dos experimentos, bem como os parâmetros de execução dos experimentos que indiquem como os experimentos devem ser executados, saindo de um estado inicial para um estado final;

- **Escalabilidade** – é desejado que o *testbed* possa suportar um grande número de dispositivos, com diferentes capacidades, e que possam estar geograficamente dispersos. A escalabilidade do *testbed* consiste na capacidade de adicionar dispositivos e recursos de forma eficiente, isto é, sem que seja necessário reconfigurar toda a infraestrutura do *testbed*;

- **Isolamento** – é importante que os experimentos sejam executados de forma isolada, de modo que um experimento não afete o outro. O isolamento dos experimentos é fundamental para garantir a segurança e a privacidade dos dados utilizados nos experimentos, bem como para garantir a reprodutibilidade dos resultados;

- **Execução segura** – em experimentos em cibersegurança, entende-se que serão empregadas técnicas e ferramentas de adversários para que o pesquisador possa estudar e entender como os ataques gerados a partir dessas funcionam e como podem ser mitigados. Assim, o *testbed* deve permitir o emprego de tais técnicas e ferramentas, porém sem que isso comprometa a infraestrutura do próprio *testbed* ou de terceiros, como aplicações e serviços externos disponíveis na Internet ou na própria rede local da instituição que mantém o *testbed*.

O MENTORED *Testbed*, apresentado na Seção 5.3, foi projetado para atender a esses requisitos, além de alguns outros relacionados ao acesso em tempo real, a interface com o usuário e gestão de times (PRATES JR et al., 2021). Assim, o MENTORED *Testbed* oferece acesso em tempo real aos dispositivos, para que um usuário possa redefinir, reprogramar e monitorar de forma não intrusiva o tráfego de rede e o estado de cada dispositivo durante a execução dos experimentos. A interface amigável fornecida ao usuário, permite a definição, execução e análise dos experimentos de forma simples e intuitiva. A gestão de times permite a atribuição de permissões e papéis aos usuários por projeto, de forma a garantir a distribuição eficiente das tarefas e a colaboração entre os pesquisadores. O testbed oferece ainda a liberdade para os usuários desenvolverem novas classes de ferramentas que facilitem suas pesquisas experimentais.

5.2. *Testbeds* para experimentação

Nesta seção são apresentados *testbeds* para experimentação que se destacam na literatura ou no seu amplo uso em pesquisas acadêmicas, com intuito de apresentar como estes

podem ser usados em pesquisas aplicadas de cibersegurança e as diversas abordagens que estes oferecem, como virtualização, emulação, conteinerização, dentre outras.

5.2.1. *Cluster* Nacional RNP

O *Cluster* Nacional da RNP é uma infraestrutura de computação em nuvem, implantada sobre recursos do Serviço de *Testbeds*[3] da RNP, distribuídos geograficamente pelo território brasileiro, e oferecida como uma plataforma robusta e flexível em ambiente nativo de nuvem para apoio à experimentação em temas de Tecnologias de Informação e Comunicação (TIC) dentro da pesquisa acadêmica e científica nacional.

Este *cluster* é composto por servidores de computação localizados nos Pontos de Presença (PoPs) da rede *backbone* da RNP, disponíveis em 12 capitais do Brasil. Como pode ser observado na Figura 5.1, a conectividade entre os PoPs é garantida pela infraestrutura de *backbone* da RNP, a Rede IPÊ, proporcionando uma rede estável e de alta velocidade com enlaces de até 100 Gbps para o *Cluster* Nacional com o objetivo de suportar as demandas de rede dos projetos de experimentação. A Figura 5.2 ilustra, por meio de um *heatmap*, as atuais latências (arrendondadas) em milisegundos entre os PoPs da Rede IPÊ.

A pilha de *software* desse ambiente utiliza como sistema operacional a distribuição Ubuntu, por se tratar de uma distribuição Linux bem conhecida e amplamente utilizada no meio acadêmico e científico, bem como por possuir alta estabilidade, suporte de 5 anos e uma boa frequência de atualizações, sendo bi-anual para novas versões LTS da distribuição e semestral para novas versões do *kernel* Linux. Como plataforma de orquestração de contêineres se utiliza o Kubernetes, juntamente com o ecossistema de componentes nativos de nuvem reconhecidos pela CNCF (*Cloud Native Computing Foundation*), e adicionalmente alguns serviços externos como DNS, armazenamento centralizado de dados e repositório privado de imagens de contêineres, possibilitando a orquestração, o compartilhamento e o uso eficiente e escalável dos recursos do *testbed* por diferentes projetos de pesquisa simultaneamente.

Além disso, o *Cluster* Nacional também pode integrar novos componentes, serviços e configurações personalizadas para atender à demandas dos projetos de pesquisa, garantindo um ambiente personalizável, flexível e completo. Como um exemplo dessa flexibilidade, foram adicionados novos componentes de *software* e configurações de *kernel* e sistema operacional para suportar aplicações de Core 5G, se tornando a primeira infraestrutura da RNP com esse suporte, após a implementação e validação de uma arquitetura de rede 5G distribuída geograficamente, com *Core Sites* e *Edge Sites*[4], desenvolvida em um outro cluster similar dentro do Testbeds RNP para atender a um projeto de pesquisa[5]. Um outro exemplo muito interessante é o do projeto KCDN[6], um projeto de pesquisa sobre redes de distribuição de conteúdo (CDNs - *Content Delivery Networks*) usando arquiteturas nativas de nuvem (*cloud native*). O projeto desenvolveu a própria arquitetura de CDN em microserviços considerando a abrangência nacional do *cluster*, e utilizou

[3]https://www.rnp.br/servicos/testbeds

[4]https://github.com/zanattabruno/5G-all-in-one-helm

[5]https://inatel.br/brasil6g/

[6]https://doi.org/10.48448/ejkd-xd29

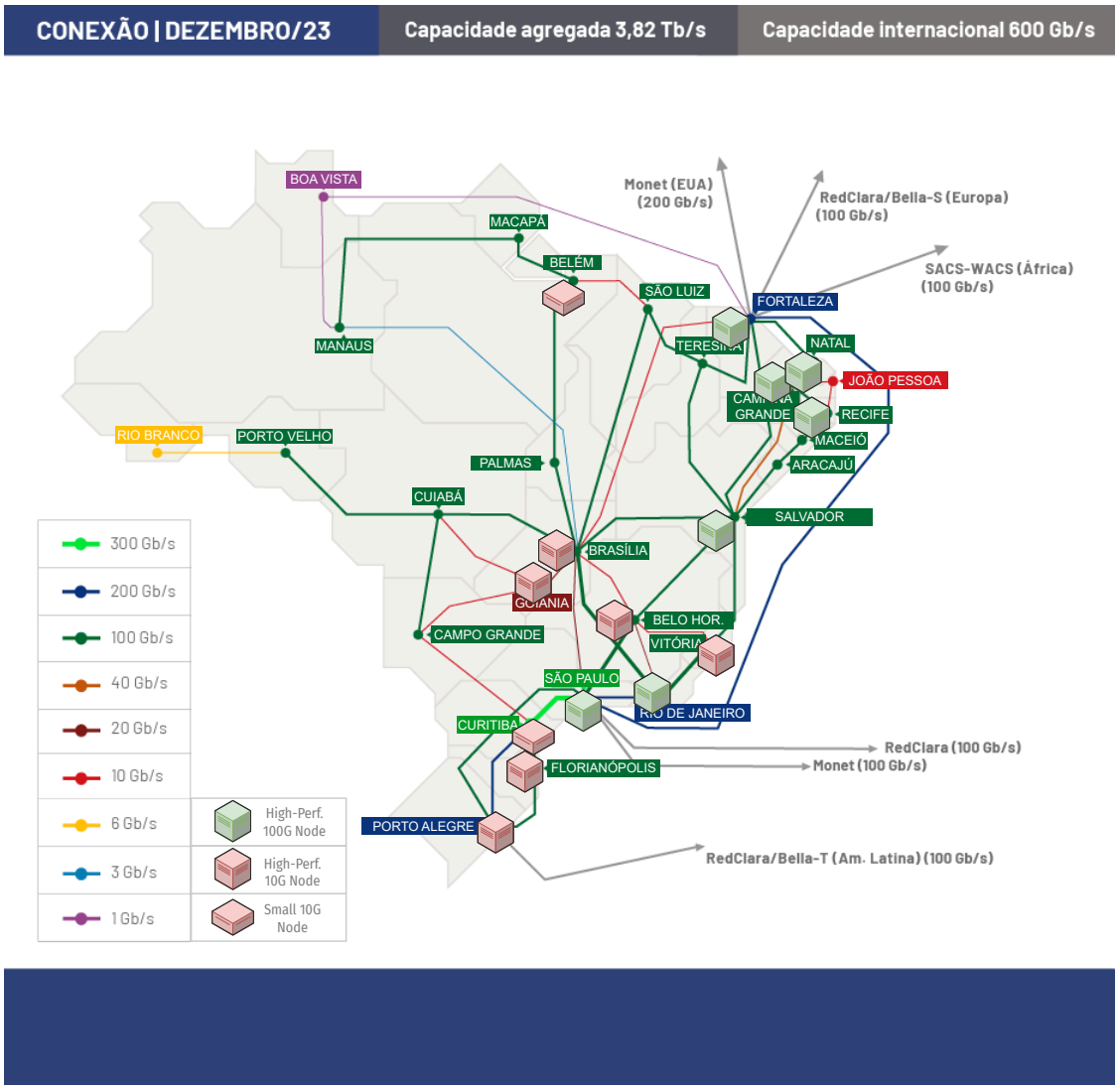


Figura 5.1: Mapa da Infraestrutura do *Cluster* Nacional interconectada pela Rede Ipê.

recursos avançados de DNS e configurações Kubernetes para implantação seletiva por localidade. Foram desenvolvidos 2 serviços DNS externos ao *cluster* para o projeto, sendo 1 deles totalmente configurado pelo KCDN. Algumas funcionalidades desenvolvidas pelo projeto foram posteriormente adicionadas à aplicação eduplay, uma plataforma de *streaming* de video usada pela RNP, e o servidor DNS principal continuou como opção de uso para os demais projetos de pesquisa.

Atualmente, o *cluster* utiliza equipamentos hospedados nos PoPs da RNP, sendo composto principalmente por servidores físicos dedicados, mas também por máquinas virtuais instanciadas no ambiente de TI da RNP em alguns casos. O modelo atual de servidor físico tem como características: 24 núcleos de CPU, 192 GigaBytes de memória RAM e 10 TeraBytes de armazenamento HDD somados a 1 TeraByte de armazenamento NVMe de alto desempenho. Outros modelos de equipamentos podem ser adicionados ao *Cluster* Nacional sob demanda, como por exemplo modelos de servidores menos robustos, com 4 núcleos de CPU, 16 GigaBytes de memória RAM e 300 GigaBytes de armazenamento

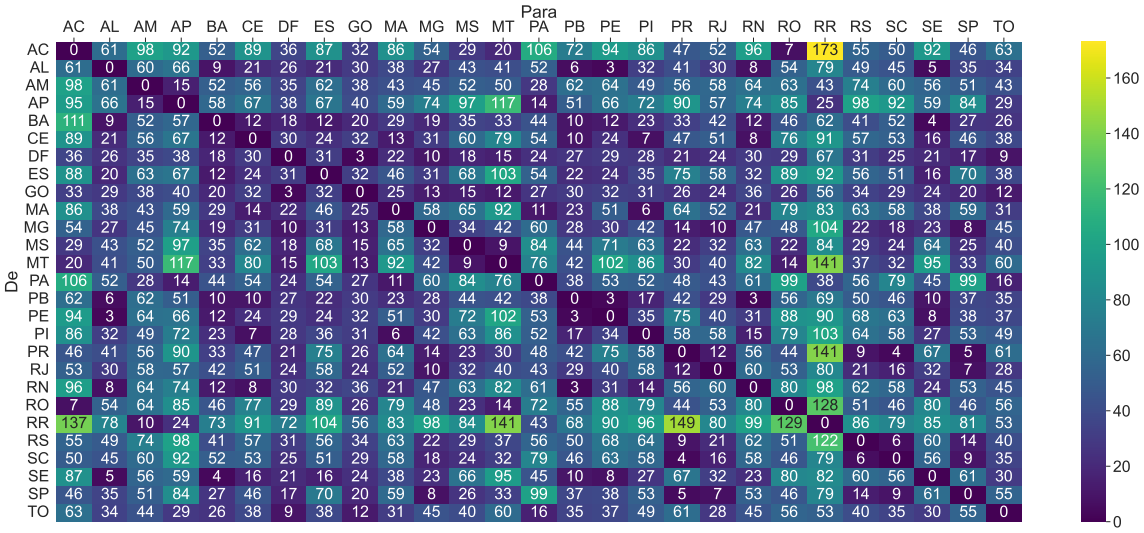


Figura 5.2: Gráfico *heatmap* referente as latências (ms) entre os PoPs da Rede IPÊ.

SSD, ou até mesmo equipamentos especializados como por exemplo dispositivos IoT (Internet das Coisas) de pequeno porte/desempenho e função específica, e *Switches* SDN P4 (*Software Defined Network*). A diversidade de equipamentos do *cluster* garante uma infraestrutura adaptável às diferentes necessidades de pesquisa.

O *Cluster* Nacional é formado por uma quantidade variável de servidores do Serviço de *Testbeds* da RNP, que são alocados como *nodes* do *cluster* conforme as características das demandas de pesquisa corrente, podendo o *cluster* estar com mais ou menos *nodes* ao longo de seu ciclo de vida. Em Julho de 2023, o somatório de recursos alocados para o *cluster* contabilizava um total de mais de 400 núcleos de CPU, mais de 2 TeraBytes de Memória RAM, e mais de 100 TeraBytes de armazenamento. Os equipamentos que são removidos do *cluster* podem ser alocados para projetos que demandem o uso dedicado do servidor, como testes de rede ou de transferências de dados de 100 Gbps, ou para constituírem *clusters* temporários de propósito específico, como por exemplo, para homologação, validação e implantação de novas versões de toda a pilha de software para a evoluçao do próprio Cluster Nacional. O ciclo de vida do *cluster* inclui o *upgrade* da própria versão da pilha Kubernetes, que é executado usando um *cluster* de homologação com a nova versão funcionando simultaneamente ao *cluster* de produção com a versão corrente. Após a validação de funcionamento de todos os projetos de pesquisa na nova versão, é feita a migração das cargas de trabalho, do acesso dos usuários e dos demais *nodes* para o novo *cluster*, fazendo com que o *cluster* de homologação se torne o novo *cluster* de produção, reiniciando todo o ciclo. Os *nodes* em migração para o novo *cluster* também podem ter seus sistemas operacionais atualizados. Com essa estratégia, mesmo as atualizações de toda a pilha do *cluster* ocorrem de forma suave, não incorrendo em qualquer impacto para os usuários.

Os *nodes* do *cluster* e seus respectivos microsserviços em contêineres podem se comunicar utilizando *plugins* de rede (CNI) oferecidos pelo ecossistema Kubernetes. Por padrão é utilizado o CNI Calico, responsável pela conectividade básica do *cluster*. CNIs adicionais como Multus e macvlan são utilizados para permitir a interconexão por meio

de circuitos virtuais criados sobre a Rede Ipê, o *backbone* da RNP, utilizando tecnologias como VXLAN e EVPN, dessa forma permitindo a criação de topologias isoladas definidas pelo usuário e com maior desempenho por utilizarem equipamentos de rede para o encapsulamento e encaminhamento dos pacotes.

A oferta de experimentação no *cluster* é feita através da disponibilização de uma fatia (uma parte) específica dos recursos dessa infraestrutura para cada projeto de pesquisa interessado. A interação com o ambiente é viabilizada através de um nó "*Bastion*", que funciona como ponto de acesso único à *API* de controle do *cluster*, ou até mesmo diretamente a partir de uma máquina remota do usuário.

Cada pesquisador, munido de suas credenciais de acesso, pode então explorar todos os recursos de sua fatia do *cluster* e demais serviços e funcionalidades do ambiente de experimentação de forma isolada dos demais usuários. Cada fatia também pode ser divida em ambientes isolados, através de *namespaces* hierárquicos, a serem utilizados por experimentos distintos, podendo ter seu compartilhamento e permissionamento configurado de forma personalizada. Para garantir uma experiência de uso consistente, são estabelecidas quantidades mínimas de recursos alocados de forma dedicada, garantindo disponibilidade, bem como limites máximos de utilização, evitando a sobrecarga do ambiente. Esta abordagem assegura a segregação dos recursos entre os usuários, ao mesmo tempo em que compartilha esses recursos disponíveis de forma eficiente. Todos esses parâmetros de garantias e limites de recursos podem ser ajustados conforme necessidade e acordo prévio com cada equipe de pesquisa usuária do *testbed*.

Os recursos alocados podem então ser utilizados pelos pesquisadores para a implantação e execução de suas aplicações, seja utilizando arquiteturas tradicionais monolíticas ou adotando abordagens mais modernas como a de microsserviços, aproveitando todas as vantagens de uma infraestrutura distribuída de nuvem. O *Cluster* Nacional também é configurado para aproveitar funcionalidades avançadas da arquitetura Kubernetes, permitindo, por exemplo, que os pesquisadores controlem o comportamento de suas aplicações dentro do *cluster* através da manipulação de alguns parâmetros, e até mesmo as tornem acessíveis externamente ao *cluster*, publicando-as como um serviço.

Para fins de coleta de dados dos experimentos e de uso do *cluster*, o ambiente também proporciona algumas ferramentas de monitoramento que podem auxiliar o experimentaddor, como a ferramenta netdata, instalada em cada *node* para prover métricas gerais de utilização em tempo real, assim como a pilha de monitoramento Kube-Prometheus-Grafana para obtenção de métricas dos objetos internos do *cluster* e suas aplicações. Adicionalmente, os pesquisadores também podem utilizar serviços de suporte externos ao *cluster*, como um armazenamento centralizado, além do armazenamento distribuído do próprio *cluster*, para centralizar coletas de dados ou outras informações em um único lugar facilitando o acesso dos dados para os projeto de pesquisa.

De maneira geral, esse ambiente proporciona o suporte a uma vasta gama de experimentos e projetos de pesquisa. Os temas abordados por grupos de pesquisa no Cluster Nacional, variam desde temas e questões já tradicionais, como Redes, Internet, Redes Programáveis (SDN), Monitoramento e Orquestração de Recursos, até a tópicos mais avançados de pesquisa, como Computação em Nuvem, Computação de Borda, Arquitetura de Microsserviços, Funções Virtualizadas de Rede (NFV), Redes de Distribuição de

Conteúdo (CDN), Segurança Cibernética, IoT (Internet das Coisas), Redes 5G e *Blockchain*. Como resultado, o *Cluster* Nacional tem desempenhado um papel importante no apoio à pesquisa em TIC, contribuindo com muitos projetos de pesquisa desde 2019.

A infraestrutura do *Cluster* Nacional oferece uma série de vantagens notáveis para os pesquisadores. Destacam-se a vasta quantidade de recursos computacionais disponíveis, distribuídos geograficamente por todo o território nacional, abrangendo todas as cinco regiões. Essa distribuição geográfica permite aos pesquisadores explorar essa característica em seus experimentos. Além disso, o ambiente de experimentação é baseado em um sistema tecnológico real, utilizando Kubernetes e o ecossistema CNCF, amplamente reconhecidos e utilizados em ambientes de produção e no mercado em geral. A escalabilidade flexível dos *nodes* e do ambiente como um todo é uma característica fundamental. O *cluster* também oferece aplicações de monitoramento integradas, bem como a modularização de componentes, que proporciona flexibilidade para personalização e integração de novos componentes pelos próprios usuários. Ademais, a interconexão é facilitada através de circuitos virtuais pelo *backbone* da RNP, garantindo taxas de vazão de até 100 Gbps em alguns casos. Essas características combinadas, proporcionam um ambiente adequado para a pesquisa e experimentação em TIC.

5.2.2. DeterLab e MergeTB

O DeterLab (WROCLAWSKI et al., 2016) é um *testbed* inicialmente criado para experimentação na área de cibersegurança, com capacidade de executar experimentos em grande escala, incluindo a execução de ataques e defesas em redes, sistemas distribuídos, sistemas cibernéticos-físicos e sistemas de controle industrial. Iniciamos esta seção descrevendo o DeterLab em seu estado antigo, e, em seguida, apresentamos a nova plataforma adotada pelo projeto, denominada MergeTB.

Concebido em 2004 com base no código de gerenciamento do Emulab[7] da Universidade de Utah, o DeterLab é um dos *testbeds* mais renomados da literatura, financiado pelo Departamento de Segurança Interna (*US Department of Homeland Security*), Fundação Nacional de Ciência (NSF, *National Science Foundation*) e a Agência de Pesquisa de Projetos Avançados de Defesa (DARPA, *Defense Advanced Research Projects Agency*).

A arquitetura do DeterLab é baseada em um modelo de experimentação federada, de forma que é possível interligar outros *testbeds*. Além disso um grupo de políticas permite que cada provedor de infraestrutura controle sobre como seus recursos podem ser utilizados. Assim é possível a execução de experimentos federados, utilizando recursos de múltiplas fontes para possibilitar um ambiente de experimentação coesivo, os quais são descritos em um formato baseado no *Network Simulator* (NS) utilizado pelo Emulab.

A infraestrutura do DeterLab é composta por um conjunto de servidores x86-64, interconectados por múltiplas interfaces Ethernet em *switches* com suporte a VLANs. Cada nó é capaz de executar múltiplos experimentos simultaneamente, sendo que cada experimento é isolado dos demais e da Internet. Além disso, cada nó permite ao experimentador escolher um dos cinco níveis de fidelidade para a execução do experimento, variando desde a alocação de um nó físico por completo para um experimento, passando

[7]https://www.emulab.net

por nós representados por máquinas virtuais QEMU, contêineres OpenVZ, processos virtualizados (ViewOS), e até a execução de *threads* (*co-routines*) em um nó físico.

A flexibilidade para escolha do nível de fidelidade permite ao experimentador escolher o nível mais adequado para o experimento em questão, sendo que os níveis mais altos de fidelidade impactam diretamente na escalabilidade do experimento. Contudo, com um nível de fidelidade mais baixo, é possível estudar fenômenos complexos envolvendo 100.000 ou mais nós, considerando que o experimentador atenha-se aos aspectos mais interessantes ao experimento em questão.

O *testbed* foi amplamente utilizado por pesquisadores, com mais de 2.600 usuários de pesquisa e educação ao redor do mundo, além de 47 instituições em 6 países (MIRKOVIC; BENZEL, 2012). Como ferramenta de ensino, o ambiente de experimentação provido pelo *testbed* pode ser utilizado para a análise de ameaças de segurança e suas defesas, com total liberdade de modificar o sistema operacional utilizado e as ferramentas aplicadas. Existe também a possibilidade de execução de uma série de exercícios pré-definidos pelo docente ou pelo próprio DeterLab, como por exemplo, uma introdução ao Linux e ao *testbed*, ataques de injeção SQL, permissionamento e *firewalls*, computação forense, detecção de intrusão em redes, ataques *Man-in-The-Middle* (MiTM), worms, botnets, e ataques DDoS diversos. Por fim, resaltamos que o *testbed* continua em funcionamento para finalidades educacionais.

Atualmente o DeterLab está em um processo de modernização, começando pela adoção de uma nova plataforma chamada MergeTB[8] e migrando para a infraestrutura do projeto SPHERE[9], que também é financiado pela NSF. Apesar desta migração, o conteúdo didático para execução de experimentos educacionais ainda está acessível por meio do antigo portal do DeterLab[10], e este continua sendo o portal para tais tipos de usuários. No entanto, o site do projeto recomenda que usuários utilizem o novo portal[11] para fins de pesquisa experimental.

A arquitetura do MergeTB é descrita em Goodfellow, Thurlow e Ravi (2018), a qual descreve o *testbed* como sendo composto por diversos serviços *stateless*. Tais serviços ao receber requisições ou dados por meio de uma interface *protobuf*, realizam operações como a realização (alocação) e materialização (instanciação) dos recursos solicitados pelo pesquisador. Durante o processo de materialização, diversos serviços denominados *drivers* são invocados para configurar recursos específicos, reservando eles temporariamente para a execução das tarefas necessárias, e, desta maneira, evitando problemas relacionados ao acesso simultâneo por múltiplos *drivers*. Além deste artigo, a arquitetura do MergeTB também é descrita em sua documentação disponível *online*[12] e em seu *blog*[13], a qual detalha aspectos adicionais como *softwares* específicos utilizados na criação do *testbed*.

[8]https://mergetb.org
[9]https://sphere-project.net
[10]https://www.isi.deterlab.net/newproject.php
[11]https://launch.mod.deterlab.net/
[12]https://mergetb.org/docs/
[13]https://mergetb.org/blog/2021/05/01/merge-1.0/

5.2.3. FIT IoT LAB

O FIT IoT-LAB (ADJIH et al., 2015) é um *testbed* de acesso aberto projetado para experimentação em larga escala, focado em dispositivos IoT sem fio. Com mais de 2.700 dispositivos IoT e 117 robôs móveis, distribuídos em seis locais na França, o objetivo do *testbed* é servir como ferramenta científica para o estudo e avanço de tecnologias sem fio, bem como de tópicos relacionados, como mobilidade, por exemplo.

O *testbed* conta com diferentes tecnologias de comunicação, incluindo 802.15.4, BLE, LoRa, rádios operando em frequências Sub-GHz e *Ultra Wide Band* (UWB). Os dispositivos contam com diferentes sistemas operacionais e são reconfiguráveis e reprogramáveis, permitindo que os usuários substituam o *firmware* conforme necessário. No momento, o *testbed* abrange 18 plataformas de IoT.[14].

O processo para a execução de experimentos envolve inicialmente a reserva de um conjunto de nós, seguido pela configuração desses nós, que pode incluir a carga de um *firmware* específico. Uma vez configurados, o experimento é então executado. Os experimentadores têm a opção de utilizar uma *API* para controlar os dispositivos ou acessar uma aplicação *web* para visualização dos experimentos em andamento.

O *hardware* dos dispositivos IoT consiste em uma placa de experimentação conhecida como *Open Node* (ON), que é reprogramável pelo usuário, e está conectada a um mini computador denominado *Gateway* (GW). O *Gateway* atua como intermediário na comunicação entre o *Open Node* e um *hardware* de controle autônomo chamado de *Control Node* (CN). O *Control Node* é responsável por coordenar a reprogramação, inicialização, *resets*, análise de consumo de energia, manipulação dos sensores, inspeção do tráfego e injeção de pacotes na rede do *Open Node*. A utilização de um microcontrolador dedicado no CN proporciona a vantagem de garantir a execução em tempo real das atividades de medição nos experimentos (ADJIH et al., 2015).

Para plataformas IoT baseadas em microcontrolador, o *Control Node* (CN) oferece ao experimentador acesso em tempo real aos dados dos sensores, bem como a capacidade de realizar ações como *sniffing* e injeção de pacotes na rede do dispositivo. Em plataformas que operam com o sistema operacional Linux, além das facilidades oferecidas para dispositivos baseados em microcontrolador, os experimentadores têm acesso a um sistema de arquivos montado no espaço de usuário e disponível via NFS (*Network File System*), permitindo a persistência de dados entre execuções de experimentos.

O FIT IoT-Lab emergiu como um *testbed* amplamente reconhecido e utilizado por pesquisadores, tendo sido citado em mais de 500 artigos científicos desde sua publicação em 2015. Embora seu foco principal não seja cibersegurança, o *testbed* tem sido empregado em diversos casos de uso. Há uma percepção de que o FIT IoT-Lab oferece potencial para experimentos relacionados à cibersegurança em redes IoT, incluindo ataques e defesas.

5.2.4. Gotham Testbed

Construído sobre o emulador de redes GNS3, o *testbed* Gotham (SÁEZ-DE-CÁMARA et al., 2023) é um *middleware* e conjunto de *scripts* projetados para gerar cenários e realizar

[14]https://www.iot-lab.info/docs/boards/overview/

ataques usando o *testbed*. Este *testbed* tem como escopo a experimentação em dispositivos IoT, bem como a geração de *datasets* contendo capturas de pacote e registros (*logs*) de aplicações. Uma contribuição notável deste trabalho é sua ênfase na reprodutibilidade. Ao contrário de muitos *testbeds* fechados e difícil acesso, o Gotham disponibiliza seu código fonte[15], possibilitando sua reprodução. Embora diversos conjuntos de dados provenientes de *testbeds* estejam amplamente disponíveis na literatura, os próprios *testbeds* raramente são publicados. Essa transparência e disponibilidade do código-fonte são essenciais para promover a reprodutibilidade e a colaboração na pesquisa em IoT e segurança cibernética.

A arquitetura deste *testbed* pode ser descrita como operando em três camadas. A camada mais inferior trata dos executores, isto é, soluções como Docker, QEMU, Dynamips, responsáveis por executar aplicações e imagens de dispositivos, tais como VPCS (*Virtual PC Simulators*), roteadores, *switches*, dentre outros. Essa camada é controlada pela segunda camada, que utiliza o GNS3, um software de código aberto capaz de interconectar dispositivos em uma topologia virtual. Essa topologia pode ser visualizada e controlada por meio de sua interface gráfica ou da *API* REST do próprio GNS3. A terceira camada do *testbed* é composta por diversos subcomponentes.

O cenário incorporado no *testbed* conta com 140 dispositivos, distribuídos em três camadas: *edge*, *network*, e *cloud*. Na primeira camada, encontram-se os dispositivos IoT e IIoT[16], distribuídos em duas zonas: zona de ameaças e zona da cidade. Dentro da zona de ameaças, o primeiro grupo é subdividido em três agentes de ameaças (*threat actors)*: *Maroni*, *Falcone* e *Calabrese*, sendo cada um representado por um modelo de ameaça distinto. Na zona da cidade, também há uma subdivisão em quatro subgrupos, cada um representando locais distintos: o Museu de História Natural (contendo sensores de monitoramento e câmeras IP); o bairro *Bristol* (contendo casas com mecanismos automação residencial e câmeras IP); a fábrica de aço *Rennington* (a qual envia telemetria de equipamentos industriais para a zona *cloud*); e, por último, a central de energia de Gotham (uma rede industrial que também envia telemetria para monitoramento de sistemas hidráulicos e de geração de energia).

A segunda camada do cenário é a zona de redes (*network*), que interconecta os diversos dispositivos mencionados anteriormente por meio de 30 switches e 10 roteadores, formando um *backbone* que orquestra rotas através do protocolo OSPF. Por fim, a zona de nuvem (*cloud*) é responsável por fornecer serviços para as camadas anteriores, como DNS, NTP, dentre outros.

No que diz respeito à experimentação de segurança, os agentes de ameaça realizam uma variedade de ataques na infraestrutura modelada. Diversos tipos de ataques e comportamentos são modelados, incluindo o uso da *botnet* Mirai[17] e a central de comando e controle (C&C) Merlin[18]. Por meio destas ferramentas, são executados ações como *scans* de rede, ataques de força bruta (usando uma lista de credenciais pré-definida), transferência arquivos maliciosos, coleta de dados, ataques DDoS (10 tipos, incluindo ataques de *flood* UDP, DNS, TCP SYN, TCP ACK, TCP *stomp*, GRE e HTTP), execução de

[15]Código fonte Gotham *testbed*: https://github.com/xsaga/gotham-iot-testbed
[16]IIoT: Industrial Internet of Things
[17]Mirai *botnet*: https://www.cloudflare.com/pt-br/learning/ddos/glossary/mirai-botnet/
[18]Merlin C&C: https://github.com/Ne0nd0g/merlin

código remoto (RCE), assim como ataques a fraquezas em protocolos específicos, como MQTT e CoAP, usando ferramentas como MQTTSA e SlowTT-Attack. Algumas alterações no código fonte Mirai foram necessárias para sua execução no ambiente isolado do *testbed*, como troca do endereço DNS *hardcoded* (8.8.8.8) para o endereço DNS do servidor do *testbed*, dentre outros.

Sáez-de-Cámara et al. (2023) faz uma avaliação dos requisitos do *testbed*. Em relação a reprodutibilidade, destaca-se que o uso de Dockerfiles[19] contribui para atingir este requisito, uma vez que estes arquivos descrevem todas as dependências e configurações dos dispositivos. Além disto, também é examinada a emulação de enlaces, uma vez que dispositivos IoT se comunicam por meio de diversos tipos de enlaces. Nesse sentido, o *testbed* deve ser capaz de representar enlaces com qualidade variável, incluindo latência e vazão. Esse requisito é atendido por meio da utilização da ferramenta `tc` do Linux.

Com relação a emulação de recursos de hardware, demonstrou-se que limites de recursos impostos via Docker resultam em métricas consistentes na ferramenta *stress-ng*. Sobre o requisito de escalabilidade, os autores indicam que o consumo de memória RAM é linear de acordo com o número de dispositivos emulados via Docker e QEMU. Por fim, são descritos os critérios de fidelidade e heterogeneidade em relação à execução de cenários com e sem atacantes. Isso inclui uma topologia complexa, diversidade de protocolos e configurações de aplicações, além da capacidade de executar ataques e monitorar o tráfego de rede e registros (*logs*) de aplicações.

5.2.5. GENI e FABRIC

GENI *testbed* (BERMAN et al., 2014) tem como principal característica sua arquitetura distribuída e federada em uma escala nacional nos Estados Unidos. Operando por mais de uma década, o projeto foi descontinuado em 2023, sendo sucedido pelo FABRIC[20]. Apesar do encerramento, é importante entender a arquitetura do GENI, dada sua influência significativa na literatura. Posteriormente, o seu sucessor, o FABRIC, é apresentado.

A arquitetura do GENI baseava-se fortemente na virtualização, o que permitia o compartilhamento eficiente de recursos, tanto computacionais quanto de armazenamento. Sua rede também dispunha de recursos virtualizados e programáveis por meio de uma rede definida por software (SDN, *Software Defined Network*) ou VLANs, além de oferecer conectividade WiMAX em determinados pontos de presença.

A virtualização no *testbed* GENI era implementada por meio de diversas tecnologias, como Linux Vserver[21], KVM (Kernel-based Virtual Machine) ou OpenVZ. O *testbed* também contava com dispositivos físicos, sem virtualização. Além disto, a SDN do *testbed* era composta por componentes utilizando o padrão OpenFlow. A federação GENI ocorria por meio de diversas camadas de abstração, autenticação e autorização. Uma das principais camadas de abstração são as 'agregações' do *testbed*, que representavam os gestores de um grupo de recursos dentro do *testbed*.

[19]Dockerfiles: são arquivos de instruções para a construção de uma imagem de contêiner por meio da ferramenta Docker

[20]https://learn.fabric-testbed.net/knowledge-base/transitioning-from-geni-to-fabric/

[21]http://linux-vserver.org/Welcome_to_Linux-VServer.org

O *testbed* também disponibilizava uma *API* aberta para permitir o desenvolvimento de múltiplas soluções de interação com o ambiente. Tais ferramentas ofereciam funcionalidades, como descoberta de recursos do ambiente, reserva de tais recursos, dentre outros. Um exemplo disso era o ambiente web chamado *Flack*, que permitia a descoberta e configuração de recursos, bem como a criação de topologias por meio da funcionalidade de "clicar e arrastar". Outra ferramenta, denominada *Omni*, servia como uma interface de linha de comando (CLI) de baixo nível para a execução automatizada de tarefas. Já *Myslice* era uma ferramenta Web que auxiliava na identificação dos recursos computacionais mais adequados para um determinado experimento. O *Gush* era utilizado como ambiente de gestão de execução de experimentos, enquanto o *Stork* permitia a implantação e configuração de *softwares* dos experimentos em larga escala. Por fim, o *framework OMF* oferecia o controle, gestão e medição dos experimentos.

FABRIC (BALDIN et al., 2019) é um *testbed* de escala nacional nos Estados Unidos e sucessor do GENI. O principal objetivo deste *testbed* é fornecer uma infraestrutura de experimentação programável, não somente nos dispositivos *edge*, mas também na própria rede que interconecta esses dispositivos. Essa programabilidade é alcançada, em parte, pela utilização de ativos de rede com suporte à tecnologia P4 [22], que é uma linguagem de domínio específico (DSL) para a definição de como um caminho de dados deve processar determinado pacote.

Outro aspecto importante da infraestrutura FABRIC é sua conectividade. Em 2023, o *testbed* inaugurou o TeraCore[23], uma rede com uma topologia de anel capaz de atingir velocidades de 1.2 Tbps por meio de conexões de fibra óptica. Além disto, o *testbed* conta com outros recursos, como FPGAs, GPUs, centenas de núcleos e terabytes de memória RAM, possibilitando experimentação em larga escala.

Cunha Pontes et al. (2023) realizam uma análise deste *testbed* sob a perspectiva do experimentador. Eles descrevem o fluxo de experimentação como: (*i*) criação de um *slice*, (*ii*) automação da configuração dos nós do experimento como roteadores e nós *edge*, e (*iii*) execução do experimento. No FABRIC, um *slice* representa a alocação de um determinado tipo de recurso do *testbed*, como uma máquina virtual, com recursos específicos e vinculada a um projeto no *testbed*. Para a configuração dos dispositivos, os autores recomendam a utilização de ferramentas de automação, como Ansible, que permitem criar um ambiente de experimentação reproduzível. Além disso, os autores destacam que há total liberdade para a modificação do sistema operacional em execução, inclusive permitindo a execução de contêineres dentro dos nós, se desejado.

5.2.6. Fed4Fire+

O projeto europeu Fed4Fire+, concluído em 2021, representou uma significativa iniciativa de pesquisa e desenvolvimento. Seu principal objetivo foi aprimorar a federação de *testbeds* já estabelecida do Fed4FIRE[24], fornecendo uma infraestrutura facilitadora para ex-

[22]https://p4.org/

[23]https://learn.fabric-testbed.net/knowledge-base/nsf-fabric-project-announces-groundbreaking-high-speed-network-infrastructure-expansion/

[24]O projeto inicial Federation for Future Internet Research and Experimentation (Fed4FIRE), financiado pela União Europeia e executado de 2012 a 2016, foi um projeto integrador que abordou o tema de pesquisa e experimentação na Internet do Futuro.

perimentação em áreas-chave, como SDN, IoT e computação em nuvem[25](DEMEESTER et al., 2022). O principal objetivo da federação é proporcionar aos pesquisadores, desenvolvedores e empresas a oportunidade de testar e validar novas tecnologias e serviços em um ambiente realista e controlado.

O portal Fed4FIRE permanece acessível[26] e é mantido pela IMEC, uma instituição de pesquisa belga. Ele oferece acesso a uma variedade de recursos na federação de *testbeds*, incluindo computação, armazenamento e redes, juntamente com ferramentas de gestão e monitoramento. Isso permite que os usuários conduzam testes em larga escala, reproduzindo cenários complexos e avaliando o desempenho, segurança e confiabilidade de suas soluções.

O portal Fed4FIRE+ apresenta e oferece uma gama de tecnologias (cabeadas, sem fio, redes móveis, IoT, SDN, Cloud, Big Data, IA, entre outras) e os dezoito *testbed* federados, incluindo descrições, localizações, configurações e documentações[27]. Além disso, uma funcionalidade útil é o monitoramento, que permite visualizar quais *testbeds* estão conectados no momento[28]. A documentação necessária para criação de conta de acesso, configuração, execução e análise dos resultados dos experimentos permanece sendo atualizada pela comunidade da federação[29]. Por fim, o portal fornece um guia sobre como integrar novos *testbeds* à federação Fed4FIRE+.

5.2.7. *Testbeds* para geração de *datasets*

Esta seção tem como propósito apresentar alguns *testbeds* dedicados à geração de *datasets* para pesquisa. Esses ambientes de experimentação são projetados para coletar dados, como tráfego de rede, telemetria de sensores e registros de aplicação (*logs*), por meio de cenários controlados. Os *datasets* gerados por esses *testbeds* têm uma ampla gama de usos, sendo frequentemente utilizados na concepção e avaliação de ferramentas de detecção e prevenção de intrusões (IDS/IPS), entre outros propósitos científicos.

TON_IoT

TON_IoT (MOUSTAFA, 2021) é um *testbed* dedicado e focado na produção de um *dataset* para pesquisa. O *testbed* é organizado em três camadas: camada *edge* – onde são dispostos dispositivos físicos como sensores, *smartphones*, *smart TVs*, além de outros dispositivos como *workstations* e *laptops*; camada *fog* – utiliza um *hypervisor* para a gestão de recursos virtualizados, por meio da plataforma NSX-VMware, que possui recursos de rede definida por software (SDN), virtualização de funções de rede (NFV), e orquestração de serviços; camada *cloud* – contêm um *dashboard* MQTT para coleta de dados de telemetria IoT, um *website* PHP vulnerável para servir como alvo de ataques de injeção e outros serviços nas *clouds* Microsoft Azure e *Amazon Web Services* (AWS).

O *dataset* TON_IoT compreende dados de telemetria de sensores, assim como

[25]https://www.tu.berlin/av/forschung/projekte/fed4fireplus
[26]https://portal.fed4fire.eu/
[27]https://portal.fed4fire.eu/explore/discover
[28]https://fedmon.fed4fire.eu/overview/
[29]https://doc.fed4fire.eu/#

capturas de pacote dos sistemas implantados. Entre os eventos contidos no *dataset*, encontram-se ataques como *scanners* (Nmap, Nessus), negação de serviço (DoS e DDoS, usando scripts em *Python*), *ransomware*, *backdoor*, injeções SQL (Sqlmap), *cross-site scripting*, *cracking* de senhas e ataques Man-in-The-Middle (Ettercap). Este *dataset* é então processado por meio de ferramentas como Zeek[30], extraindo 44 atributos dos fluxos de dados do *dataset*. A partir destes dados, quatro modelos de inteligência artificial são confeccionados: *Gradient Boosting Machine* (GBM), *Random Forest*, *Naive Bayes*, e *Deep Neural Network*, atingindo um *F1-score* de 0.99, no caso do *Random Forest*.

Em Alsaedi et al. (2020), são detalhadas outras características do *testbed*, como por exemplo, a descrição de *scripts* Javascript que simulam o comportamento de dispositivos como uma geladeira, termostato, e GPS, interligados por meio da ferramenta Node-RED. As informações geradas pelos sensores são enviadas por meio do protocolo MQTT para um *broker* Hive-MQTT na camada de nuvem. Além disto, o *pipeline* de processamento dos dados é apresentado. Os dados gerados pelo *tested* são catalogados (*labelling*), é feita a divisão 80/20% entre treinamento e avaliação, ocorre a estratificação dos dados usando a técnica *k-fold* e o processamento por meio de outros algorítimos de inteligência artificial, tais como *Logistical Regression*, *Linear Discriminant Analysis*, *k-Nearest Neighbour* (kNN), *Classification and Regression Trees* (CART), *Random Forest*, *Naïve Bayes*, *Support Vector Machine*, *Long Short-Term Memory* (LSTM). Os resultados obtidos são variados, conforme o tipo de sensor incluso no *dataset* em questão.

Bot_IoT

Em Koroniotis, Moustafa, Sitnikova et al. (2019), é apresentado o Bot-IoT, um *dataset* construído a partir de um *testbed* dedicado, que utiliza máquinas virtuais utilizando o VMWare EXSi. O cenário utilizado conta com 4 máquinas virtuais utilizando do sistema operacional Kali (executando *scans* de rede, ataques DDoS e outros ataques típicos de *botnets*). Uma VM que utiliza o Windows 7, além de uma instância do Metasploitable[31]. O cenário também conta com uma VM com servidor Ubuntu, configurada com diversos serviços rede, como DNS, e-mail, FTP, HTTP, e SSH, além de dispositivos IoT simulados por meio da utilização de *scripts* e da ferramenta Node-RED. Além do servidor, há um dispositivo Ubuntu *mobile*, um servidor para o monitoramento do tráfego de rede, além de outros ativos de rede como um roteador pfSense e um firewall.

O *dataset* contém diversos tipos de fluxos de dados, abrangendo uma variedade de ataques e sensores que utilizam protocolos como o MQTT. Esses dados são coletados no formato *pcap*, e os fluxos de dados contidos nessa captura são analisados usando a ferramenta Argus. Em seguida, esses dados são persistidos em um banco de dados MySQL para etapas adicionais de processamento. É importante destacar que a etiquetagem (*labelling*) dos fluxos como maliciosos ou não é realizada com base nos endereços IP, ou seja, os fluxos provenientes dos IPs atribuídos às máquinas atacantes são marcados como tráfego malicioso.

[30]Zeek: https://zeek.org/

[31]Metasploitable é uma máquina virtual que intencionalmente contem um grande número de vulnerabilidades, servindo então como um alvo para ataques diversos.

Em (KORONIOTIS; MOUSTAFA; SITNIKOVA et al., 2019) são apresentadas as etapas para a criação de modelos de aprendizado de máquina para a classificação dos fluxos. Diante da grande quantidade de dados (69GB), foram selecionados os 10 melhores atributos do *dataset* por meio de métodos estatísticos, como análise de entropia e correlação de atributos. Estes atributos foram utilizados em modelos como SVM (*Support Vector Machine*), RNN (*Recurring Neural Network*) e LSTM (*Long Short-Term Memory*). Constata-se que há uma boa acurácia dos modelos empregados para a detecção da existência de ataques (classificação binária) e classificação do tipo de ataque. Contudo, o ataque de exfiltração de dados possuiu o pior índice de classificação (multi-classe).

5.2.8. Considerações sobre os *testbeds*

Tendo em vista os *testbeds* apresentados anteriormente, esta seção visa realizar uma comparação entre eles, destacando seus pontos fortes e, ao mesmo tempo, suas limitações de escopo ou arquitetura.

Primeiramente, em relação aos *testbeds* apresentados, observa-se uma ampla variedade de escalas. Projetos como Gotham possuem uma arquitetura geograficamente centralizada, onde os diversos nós modelados nos experimentos estão nos mesmos dispositivos físicos ou localizados no mesmo ambiente. Por outro lado, outros *testbeds* na literatura adotam uma arquitetura distribuída (DeterLab, GENI e Fed4FIRE+), e alguns incluem recursos como federação, permitindo a interconexão de recursos de diferentes entidades (Fed4FIRE+ e FABRIC).

A fidelidade é outro aspecto crucial para avaliar um *testbed*. Neste contexto, destaca-se a abordagem do projeto DeterLab, que oferece múltiplos níveis de representação de nós. Isso permite aos pesquisadores fazer compensações entre a fidelidade de cada componente no experimento e a escalabilidade. Por outro lado, outros *testbeds*, como o apresentado no FIT IoT, optam por utilizar apenas dispositivos físicos para representar os nós do experimento. Embora esta abordagem proporcione maior fidelidade, também apresenta desafios, como a manutenção, escalabilidade limitada e a necessidade de substituir dispositivos para acompanhar novos paradigmas e tecnologias em estudo.

Já em relação a flexibilidade, os *testbeds* apresentados permitem a configuração de diversos tipos de experimentos. Tal flexibilidade é alcançada por meio da utilização de redes e dispositivos programáveis, como por exemplo no caso do GENI, DeterLab, FABRIC, Fed4FIRE. Frisamos outro aspecto importante para a flexibilidade, a programabilidade da rede interconectado os nós de um experimento. Tal programabilidade pode ser alcançada de diversas maneiras, contudo destacamos técnicas de redes definidas por software (SDN) empregadas por protocolos como OpenFlow. Este protocolo permite ter uma malha de rede configurável e inteligente, permitindo a criação de cenários com topologias dinâmicas.

Em relação ao isolamento, observa-se uma variedade de abordagens, com a virtualização e a conteinerização emergindo como duas das principais para modelagem dos nós. Além dos benefícios em termos de segurança, essas abordagens permitem um compartilhamento mais eficiente de recursos, eliminando a necessidade de reservar um nó físico para cada experimento. Outro aspecto do isolamento abordado pelos *testbeds* diz respeito à segmentação da rede, onde soluções baseadas em segmentação dividem as interfaces do

caminho de dados de um experimento das interfaces do plano de controle (FIT IoT, GENI, dentre outros). A execução segura é outro aspecto relacionado ao isolamento, empregando as mesmas ferramentas mencionadas anteriormente, mas com uma ênfase diferente. As principais características dos *testbeds* apresentados são resumidas na Tabela 5.1.

Tabela 5.1: Comparação dos *testbeds* apresentados

Testbed	**Recursos**	**Flexibilidade**	**Escala**	**Execução Segura**	**Disponibilidade**
DeterLab (Antigo)	CT, VM, *Phy**	Alta	Alta	Sim	Educação apenas
DeterLab (MergeTB)	VMs, *Phy*	Alta	ND	Sim	Sim
FIT-IoT LAB	*Phy*	Alta	Alta	ND	Sim
Gotham	CT e VMs	Alta	Média	Sim	Sim (código fonte)
GENI	VM, CT, *Phy*	Alta	Alta	Sim	Descontinuado
FABRIC	VMs	Alta	Alta	Sim	Sim
Fed4Fire+	Misto‡	Misto‡	Misto‡	Misto‡	Sim
TON_IoT	*Phy* e VMs	Baixa	Baixa	ND	Apenas *dataset*
Bot-IoT	VMs	Baixa	Baixa	ND	Apenas *dataset*

Phy Dispositivos físicos; *VM* Máquinas Virtuais; *CT* Containers; ‡ Dependente do *testbed* federado selecionado; *ND* Não detalhado; $*$ Dentre outros

Além dos *testbeds* anteriormente discutidos, há uma variedade de outros *testbeds* na literatura, cada um com focos de atuação distintos e arquiteturas diferentes. Por exemplo, em Koroniotis, Moustafa, Schiliro et al. (2021), é apresentado um *testbed* voltado para o estudo de dispositivos IoT industriais (IIoT). Outros *testbeds*, como o abordado em Siboni et al. (2018), realizam análises das vulnerabilidades de dispositivos IoT. Em Thom et al. (2021), é apresentado um *testbed* que modela uma cidade, incluindo dispositivos IIoT e IoT, juntamente com tecnologias como redes definidas por software (SDN) para experimentação. Além disso, vale destacar, a influência de *testbeds* como o Planetlab (CHUN et al., 2003) e Emulab, que desempenharam um papel significativo na formação do panorama atual de *testbeds* existentes.

Por fim, é importante ressaltar que os *testbeds* destinados à pesquisa experimental em cibersegurança são ambientes complexos e dispendiosos, não apenas em termos financeiros, mas também em relação ao tempo necessário para sua criação, manutenção, atualização e validação. Por estes e outros motivos, muitos *testbeds* são descontinuados ou substituídos ao longo do tempo, ou até mesmo nunca são disponibilizados para uso por pesquisadores externos.

5.3. MENTORED *Testbed*

O MENTORED *Testbed* é uma infraestrutura avançada projetada para conduzir experimentos em cibersegurança, com um enfoque específico na análise de vulnerabilidades, ataques e estratégias de defesa associadas aos dispositivos da Internet das Coisas (IoT, *Internet of Things*). Esse *testbed* se destaca pela sua capacidade de criar ambientes de rede realistas, permitindo que pesquisadores avaliem a eficácia de diferentes técnicas de

segurança em um ambiente controlado, mas ao mesmo tempo complexo e diversificado.

A arquitetura do MENTORED *Testbed*, conforme delineado por Gemmer et al. (2023), destaca-se por sua flexibilidade e escalabilidade, o que facilita a integração com uma gama de dispositivos IoT. Um dos seus principais diferenciais é sua independência em relação aos recursos de processamento, o que possibilita a construção e adaptação do *testbed* de acordo com as necessidades específicas de cada experimento. Além disso, a arquitetura oferece suporte a uma diversidade de funcionalidades que simplificam as diferentes etapas do ciclo de vida dos experimentos, desde a sua definição e execução até a análise e interpretação dos resultados (MEYER; GEMMER; SCHWARZ et al., 2022).

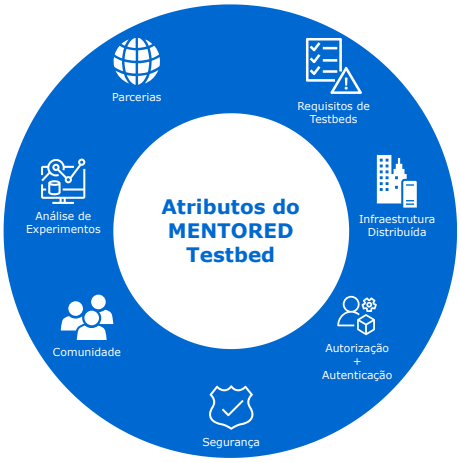


Figura 5.3: Atributos do MENTORED *Testbed*. Adaptado de (GEMMER et al., 2023).

A Figura 5.3 ilustra os principais atributos do MENTORED *Testbed*, os quais são detalhados a seguir:

- Parcerias: o MENTORED *Testbed* promove colaborações com diferentes projetos e equipes, visando a melhoria contínua da infraestrutura. Por exemplo, uma equipe da RNP está atualmente envolvida no desenvolvimento do *testbed*, especialmente para aprimorar a infraestrutura baseada em Kubernetes;
- Requisitos de *testbeds*: o *testbed* foi construído e é atualizado de forma a atender os requisitos essenciais para *testbeds* de cibersegurança. Estes requisitos incluem fidelidade, flexibilidade, escalabilidade, isolamento, execução segura, reprodutibilidade, transparência, perspectiva centrada no usuário e acesso em tempo real;
- Infraestrutura distribuída: ao contrário das simulações executadas em uma única máquina, a infraestrutura do MENTORED *Testbed* se destaca por sua distribuição entre diversos nós. Essa abordagem possibilita uma representação mais realista da comunicação entre os diferentes componentes da topologia, refletindo com maior precisão as dinâmicas das redes complexas;
- Autenticação e Autorização: o *Testbed* implementa autenticação federada e mecanismos robustos de autorização para garantir que apenas usuários autorizados possam acessar seus recursos. Essa abordagem visa reduzir o atrito no acesso aos recursos do *testbed*, tornando o processo mais fluido para o usuário;

- Segurança: devido à criticidade de experimentos científicos de cibersegurança, é fundamental que o MENTORED *Testbed* consiga evitar que os experimentos conduzidos sejam direcionados para a Internet e que não ocorra vazamento de dados entre diferentes experimentos, independentemente de serem executados por um mesmo usuário ou por usuários diferentes;

- Comunidade: todos os usuários do MENTORED *Tesbed* podem contribuir por meio de *feedbacks* e compartilhando suas definições de experimentos, possibilitando que outros pesquisadores repliquem e validem os resultados obtidos;

- Análise de experimentos: o MENTORED *Testbed* oferece aos usuários acesso contínuo aos dados e resultados dos seus experimentos, tanto em tempo real durante sua execução quanto após sua conclusão. Isso é viabilizado por meio de um mecanismo de armazenamento que otimiza a seleção de arquivos para preservação futura e consultas. Adicionalmente, para cada execução de experimento são gerados arquivos de *logs* detalhados, fundamentais tanto para a análise minuciosa dos experimentos quanto para a identificação e resolução de eventuais falhas.

O MENTORED *Testbed* incorpora tecnologias avançadas de virtualização e conteinerização, como Docker e Kubernetes. Isso permite a criação de ambientes de teste isolados e reproduzíveis, uma característica crucial para a análise de *malwares* e ataques de rede, onde a contenção e a repetibilidade dos experimentos são essenciais para garantir a validade dos resultados. Além disso, o *testbed* provê suporte a uma ampla gama de ferramentas de análise de segurança e monitoramento de rede, proporcionando aos usuários a capacidade de detectar, analisar e responder a incidentes de segurança em tempo real.

Em resumo, o MENTORED *Testbed* oferece uma plataforma avançada e flexível para a investigação de ameaças de segurança. Com sua arquitetura escalável, interface de usuário amigável e integração de tecnologias avançadas, o *testbed* facilita a execução de experimentos complexos, o que pode contribuir significativamente para o avanço da pesquisa e desenvolvimento em cibersegurança.

5.3.1. MENTORED *framework*

Devido à complexidade envolvida no desenvolvimento de *testbeds* que atendam aos requisitos apresentados na Seção 5.1.2, Gemmer et al. (2023) propuseram um *framework* genérico que serve de referência para construção de *testebeds* para experimentação em cibersegurança, incluindo ataques DDoS provenientes de dispositivos de IoT. Qualquer desenvolvedor pode instanciar esse *framework* para definir seu próprio *testbed*, usando as tecnologias que desejar. Esta seção detalha como utilizar este *framework* para a experimentação em cibersegurança, descrevendo as entidades e os módulos envolvidos e como eles se relacionam para viabilizar a definição, execução e análise de experimentos.

De acordo com Gemmer et al. (2023), o *framework* proposto, usado construção do MENTORED *Testbed*, considera um controlador geral, que é responsável por interligar as cinco principais entidades: 1) Provedor de recursos de IoT; 2) Provedor de recursos de processamento; 3) Canal de comunicação de recursos; 4) Provedor federado de identidade; 5) Orquestrador e gerenciador de recursos para experimentos (*Master*).

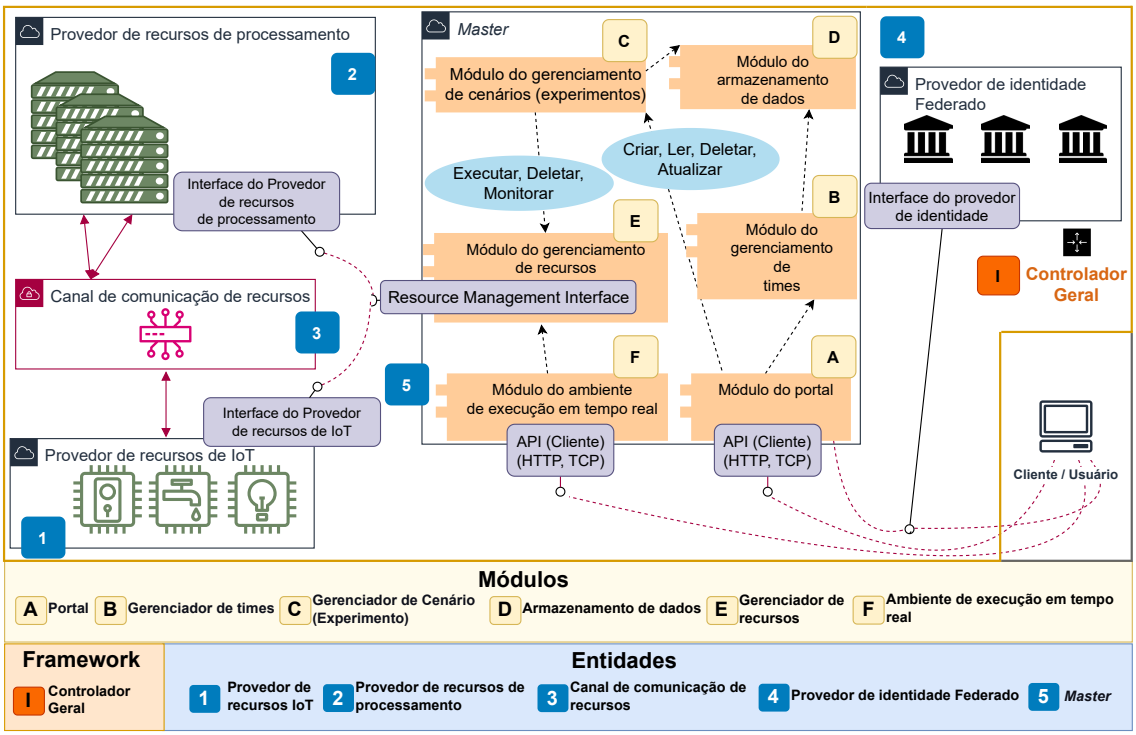


Figura 5.4: *Framework* utilizado pelo MENTORED *Testbed*. Adaptado de (GEMMER et al., 2023).

A Figura 5.4 apresenta os principais componentes do *framework* usado pelo MENTORED *Testbed* e suas relações. O *framework* considera como premissa a distribuição geográfica de provedores de recursos de processamento que são utilizados para executar experimentos. O *framework* considera ainda a existência de provedores específicos para recursos IoT, os quais podem se comunicar com os demais provedores, por meio de um canal de comunicação implementado exclusivamente para o *testbed*.

O *Master* é a entidade do *framework* que gerencia a intermediação entre experimentadores e os provedores de recursos disponíveis no *testbed*. Após se autenticar por meio de um provedor de identidade federado e obter a devida autorização, o usuário seguindo as etapas descritas na Figura 5.5 poderá ter acesso a diversas funcionalidades do *Master* como a definição, execução, armazenamento, coleta de dados e acesso em tempo real a qualquer recurso que faça parte do experimento. Um usuário autenticado poderá interagir com o *Master* por meio de duas interfaces: um portal *web*, que consiste em uma interface gráfica amigável; e uma *API* REST, que permite a interação programável com o *testbed* por meio do protocolo HTTP.

5.3.2. Requisitos e estratégias do MENTORED *Testbed*

Assim como outros *testbeds* apresentados na Seção 5.2, o MENTORED *testbed* também foi desenvolvido para atender diferentes requisitos para que o experimentador possa definir, executar e analisar os seus experimentos da melhor forma possível. A Figura 5.6 ilustra a relação entre os requisitos considerados pelo MENTORED *testbed* e as estratégias utilizadas para suprir esses requisitos.

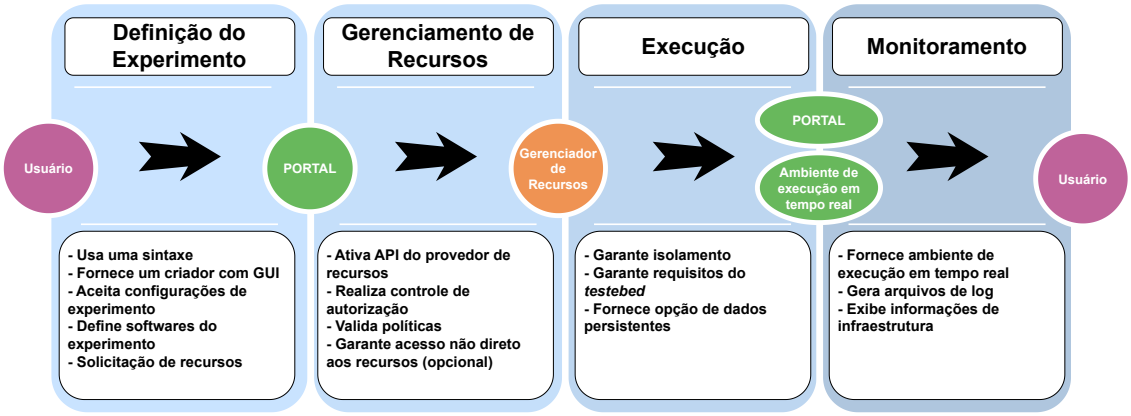


Figura 5.5: Fluxo de execução de experimento no MENTORED *Framework*.

5.3.2.1. Reprodutibilidade

Um dos requisitos considerados no MENTORED *testbed* é a reprodutibilidade de experimentos. Esse requisito é atendido por meio da padronização de definições de experimentos, os quais são feitos por meio de um arquivo do tipo YAML que estende a definição dos recursos *pods* na tecnologia Kubernetes[32].

No Kubernetes, um *pod* pode ser definido especificando um ou mais contêineres Docker e os parâmetros para inicialização (privilégios do contêiner, variáveis de ambiente, comandos para inicialização do contêiner, etc.). Os contêineres Docker por sua vez são definidos utilizando *Dockerfiles*[33], especificando o equivalente a sistema operacional como base e comandos que podem ser usados para instalar e executar softwares.

Os experimentos definidos no MENTORED *testbed* devem conter os seguintes elementos: nome do experimento, tempo limite para a execução do experimento, lista de *pods* (seguindo sintaxe Kubernetes) e tipo de topologia utilizada no experimento. Um recurso importante do *testbed* é a possibilidade de criar réplicas de *pods*, de forma que o experimentador possa escalar o experimento conforme desejar. Em uma atualização futura, o MENTORED *testbed* terá também um repositório de experimentos e definições de *pods*, potencializando a cooperação de pesquisadores, por meio do compartilhamento e reprodutibilidade de definições experimentos.

Um recurso importante em *testbeds* de cibersegurança é a possibilidade de definir topologias de redes personalizadas para diferentes tipos de experimentos. Devido à modularidade dos recursos de processamento no MENTORED *testbed*, o experimentador sempre poderá ter a opção de definir em qual recurso físico cada *pod* de seu experimento será executado. Esse tipo de recurso é facilitado pelo Kubernetes e a infraestrutura definida por software do *Cluster* Nacional da RNP.

[32]https://kubernetes.io/docs/concepts/workloads/pods/
[33]https://docs.docker.com/get-started/02_our_app/

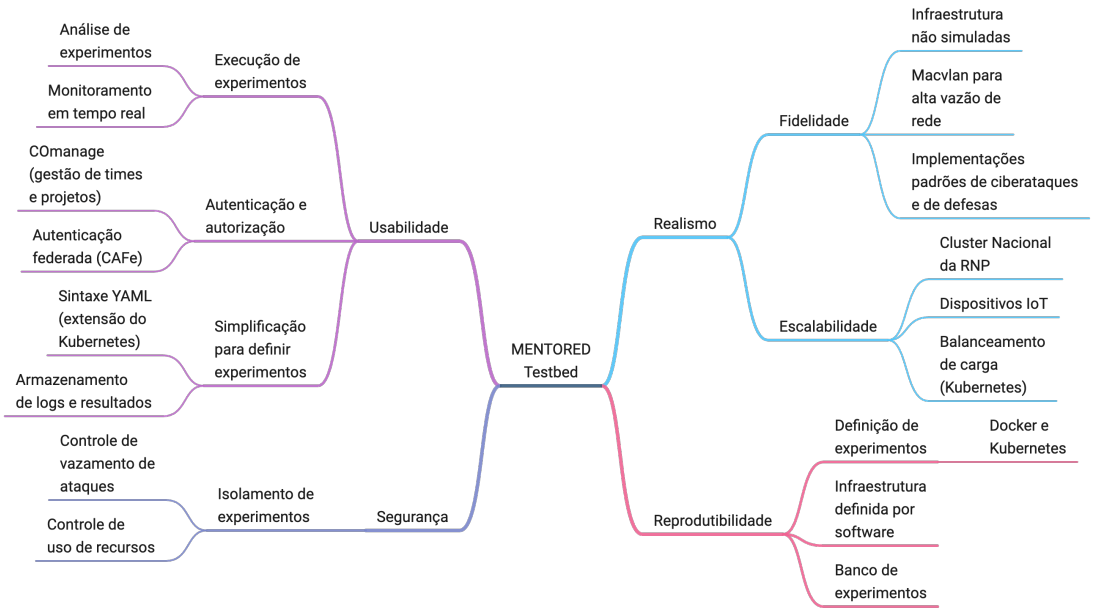


Figura 5.6: Requisitos e estratégias consideradas no MENTORED *testbed*.

5.3.2.2. Realismo

Embora seja impossível alcançar total realismo em muitos cenários de ambientes controlados para experimentos, muitos *testbeds*, incluindo o MENTORED, tem como objetivo serem o mais realista possível. O requisito de realismo pode ser dividido em fidelidade e escalabilidade. No MENTORED *testbed*, a fidelidade é alcançada usando múltiplos recursos físicos de processamento para executar experimentos, em contraste com alternativas como o *Network Simulator* (GOMEZ et al., 2023). Outro recurso importante é o uso da tecnologia de rede Macvlan. O Macvlan é uma tecnologia que reduz os componentes virtuais necessários para comunicar *pods* em um *cluster* Kubernetes, consequentemente maximizando assim o desempenho e aproximando-se da capacidade real dos recursos de comunicação e processamento (CLAASSEN; KONING; GROSSO, 2016).

Dentro do repositório de definições de experimentos do MENTORED *testbed*, estarão incluídos diversos contêineres Docker com implementações de elementos comumente necessários para experimentos de cibersegurança, como softwares para simular ataques DDoS e servidores *Web*. O experimentador poderá utilizar esses softwares, e outros caso deseja implementá-los, e replica-los em diferentes nós do *Cluster* Nacional da RNP. Além disso, em breve será possível utilizar também dispositivos IoT que se comunicarão com os nós de processamento do *Cluster* Nacional. A orquestração e balanceamento de carga da execução e comunicação dos contêineres são realizados em alto nível pelo MENTORED *Master* e em baixo nível pelo Kubernetes.

5.3.2.3. Segurança

Para que o ambiente seja seguro para o experimentador, o MENTORED *testbed* implementa estratégias que asseguram o isolamento dos experimentos. Isso significa que todas as atividades realizadas pelos softwares dos experimentos são controladas para não interferir no funcionamento adequado do *cluster*, nem na execução de outros experimentos. Além disso, o *Kubernetes* pode ser usado para evitar que softwares potencialmente maliciosos tenham acesso à internet e infectem ou ataquem serviços não relacionados aos experimentos (BUDIGIRI et al., 2021; NGUYEN et al., 2020).

5.3.2.4. Usabilidade

Conforme discutido anteriormente, uma das estratégias principais do MENTORED *testbed* é simplificar o processo de definição de experimentos. Os usuários podem criar experimentos complexos utilizando apenas um arquivo no formato YAML, estendendo a sintaxe do Kubernetes, conhecido por sua ampla popularidade e documentação (SAYFAN, 2017). Na definição dos experimentos, os experimentadores podem especificar quais arquivos e *logs* devem ser salvos após a conclusão do experimento, facilitando análises posteriores.

Para definir e executar experimentos no MENTORED *testbed*, o usuário deve autenticar-se e obter a autorização adequada. Essas etapas são implementadas usando as tecnologias de autenticação federada CAFe e de gerenciamento de times COmanage, o que será explicado com mais detalhes na Seção 5.3.3.3.

Após receber a devida autorização, um experimentador poderá executar a definição de um experimento dentro do contexto de um projeto. Se o projeto tiver acesso aos recursos especificados na definição do experimento, um novo ciclo de vida de experimento é iniciado. O ciclo de vida do experimento será explicado com mais detalhes na Seção 5.3.3.4. Durante esse processo, o experimentador pode acessar em tempo real qualquer *pod* em execução por meio da tecnologia *Webkubectl*[34]. Isso permite ao experimentador acompanhar os experimentos e realizar as análises necessárias dos dados gerados no ambiente do *testbed*.

5.3.3. Recursos tecnológicos do MENTORED *Testbed*

O desenvolvimento do MENTORED *testbed* é uma combinação de diversas tecnologias, sendo algumas delas criadas exclusivamente para o *testbed*, como ilustrado na Figura 5.7. O MENTORED *testbed* é uma instância do *framework* apresentado na Figura 5.4 que utiliza o *cluster* nacional da RNP como principal infraestrutura para os provedores de recursos de processamento. A manutenção e acesso desses dispositivos são realizados majoritariamente usando a tecnologia Kubernetes. Atualmente, a versão v1.27.7 do Kubernetes é utilizada tanto pelo MENTORED quanto pelo *cluster* nacional da RNP.

[34]https://github.com/1Panel-dev/webkubectl

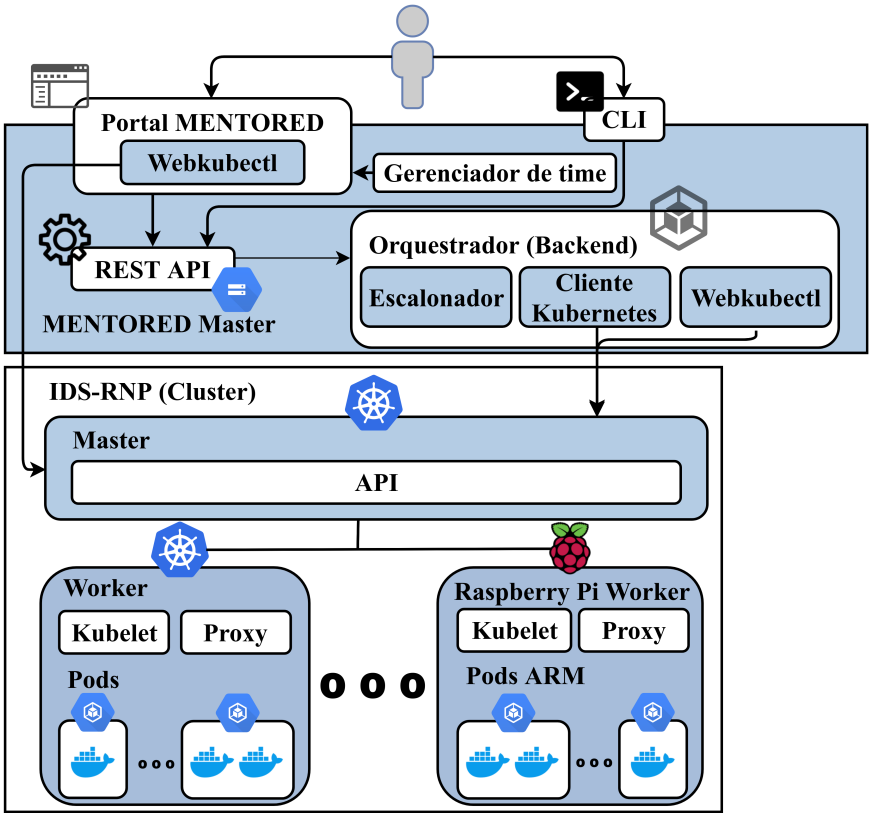


Figura 5.7: Implementação do MENTORED *testbed framework*. Adaptado de (GEMMER et al., 2023).

5.3.3.1. MENTORED Master

O MENTORED *Master* é a principal entidade no MENTORED *Testbed*, responsável por intermediar o experimentador e os recursos computacionais. Sua implementação é feita utilizando a ferramenta Docker-Compose, que gerencia diversas tecnologias que compõe o MENTORED *Master*. A entidade MENTORED *Master* pode ser dividida nos seguintes módulos:

- **Portal web**: interface GUI usada pelo experimentador, implementada usando o *framework* React.
- **Módulo de gestão de times**: implementado usando a tecnologia COmanage, este controla a interação entre diferentes experimentadores e suas autorizações para acessar determinados recursos do *testbed*.
- **MENTORED *core***: responsável por implementar o gerenciamento de recursos e comunicação com os provedores de recursos. Sua implementação é feita em Python, e utiliza a biblioteca oficial do Kubernetes para gerenciar *pods* e outros recursos dos provedores de recurso de processamento. Além dos recursos padrões do Kubernetes, o MENTORED *core* implementa diversas funcionalidades importantes para experimentos, tais como: mecanismos de *warmup* (sincronização da ini-

cialização), gerenciamento de *logs*, armazenamento de arquivos gerados por *pods*, definição de IPs e mecanismos de comunicação entre *pods*.

- **MENTORED *api***: serviço *web* responsável por validar e armazenar dados relacionados a experimentos. Implementa uma *api* REST usando o *framework* Django e biblioteca Django-Rest, que permite a adoção da especificação OpenAPI. A *api* implementada é utilizada pelo Portal *web*, de forma que os recursos implementados pelo MENTORED *core* possam ser acionados usando o protocolo HTTP. Atualmente, não existe nenhuma biblioteca oficial em Python ou qualquer outra linguagem para acessar o MENTORED *api*, mas o experimentador pode utilizar a documentação da *api* para utilizar o MENTORED *testbed* de forma automatizada e programável.
- **Módulo do ambiente de execução em tempo-real**: implementado usando a tecnologia Webkubectl, que permite utilizar comandos do Kubernetes por meio de *web browsers*. Seu uso é controlado por mecanismos de acesso via *tokens* gerados pelo Kubernetes, que garantem isolamento de experimentos por meio do recurso chamado *namespace*. Um usuário pode acessar qualquer *pod* em execução dentro dos *namespaces* relacionados aos experimentos que este tem acesso. Uma vez que um usuário tem acesso a um *pod*, qualquer comando pode ser executado dentro do contêiner em execução, semelhante a um acesso SSH.

5.3.3.2. Portal do MENTORED *testbed*

O portal que implementa a principal interface de acesso de experimentadores ao MENTORED *testbed* consiste em uma aplicação *web* que acessa a API *REST* disponibilizada pelo MENTORED *master*. Por meio do acesso a essa API, o usuário pode ter acesso a diversas funcionalidades e recursos do *testbed*, como no *Dashboard* ilustrado na Figura 5.8a. Além disso, o usuário pode gerenciar os projetos ao qual pertence como mostrado na Figura 5.8b[35]. Por fim, definições e execuções experimentos de experimentos também podem ser criados e monitorados no portal, assim como demonstra as Figuras 5.11a e 5.11b.

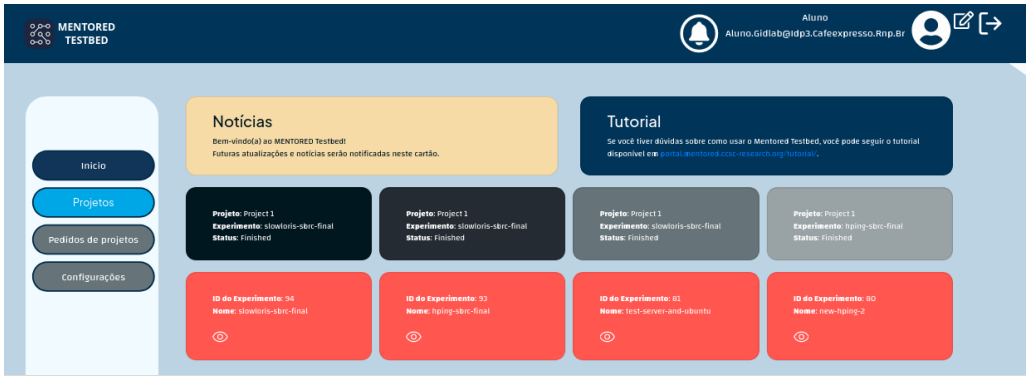
5.3.3.3. Gestão de projetos e times

Um projeto de pesquisa em cibersegurança geralmente envolve a colaboração de uma equipe multidisciplinar, composta por um coordenador principal do projeto, pesquisadores e desenvolvedores. Para garantir que todos os membros da equipe possam contribuir efetivamente, é essencial que haja uma estrutura clara de responsabilidades e permissões, que permita a distribuição das tarefas e a coordenação das atividades.

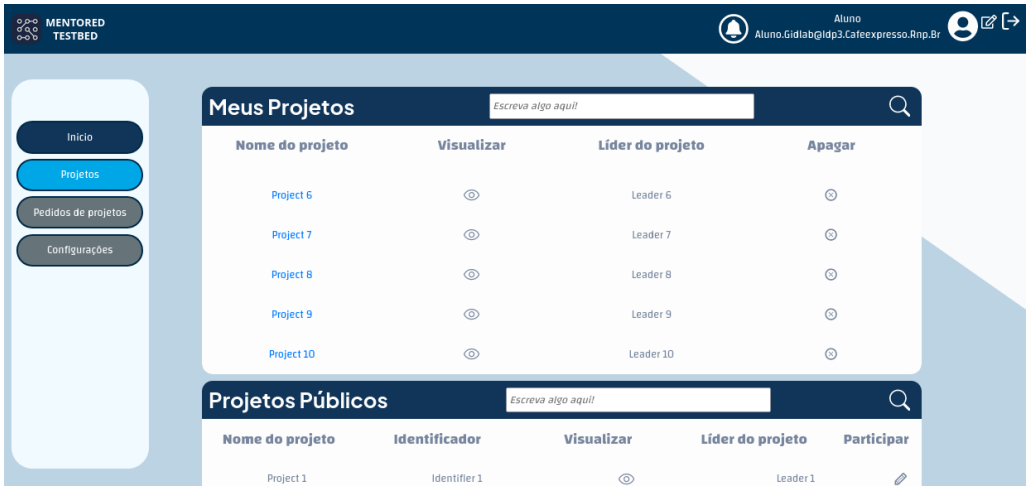
O COmanage[36] é uma ferramenta versátil que pode ser incorporada em soluções de gerenciamento de identidades e colaboração (FLANAGAN et al., 2012). O MENTORED *Master* faz uso do COmanage para gerenciar os usuários e suas permissões, facili-

[35]O recurso de gerenciamento de times ainda está em desenvolvimento no MENTORED *testbed*

[36]https://incommon.org/software/comanage



(a) *Dashboard* do portal no MENTORED *testbed*.



(b) Interface para gestão de times (em desenvolvimento).

Figura 5.8: Portal do MENTORED *testbed*.

tando assim a colaboração nos experimentos de cibersegurança usando o MENTORED *Testbed*. Para a integração do COmanage com o MENTORED *Testbed* foi modelado um conjunto de papéis que podem ser atribuídos aos usuários, determinando os recursos e funcionalidades disponíveis para cada usuário de acordo com o seu papel.

Na Figura 5.9 é apresentado um diagrama de casos de uso do MENTORED *Testbed*. O atores são os usuários do sistema, de acordo com os papéis que desempenham, enquanto os casos de uso representam as funcionalidades disponíveis no sistema, para cada ator.

O ator "Não membro" representa usuários que não estão cadastrados no sistema. Assim, a única interação possível é a realização do cadastro no sistema. Todo pedido de cadastro é avaliado por um Administrador do *testbed*, que pode aceitar ou rejeitar a solicitação.

O ator "Membro ativo" representa usuários que estão cadastrados e podem requisitar a criação de novos projetos. O pedido de criação de um novo projeto é avaliado por um Administrador do *testbed*, que pode aceitar ou rejeitar a solicitação. Uma vez que o projeto é aceito, o usuário que fez a requisição se torna o Líder de Projeto. Trata-se do

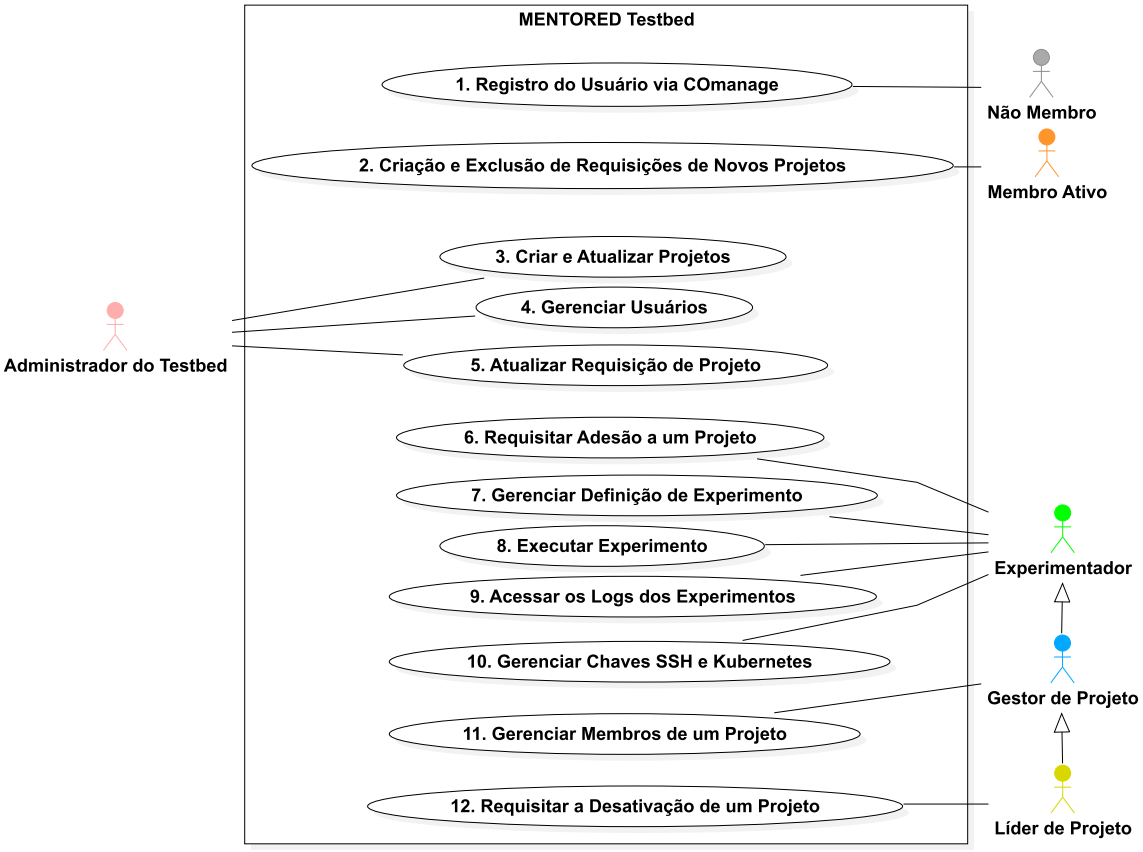


Figura 5.9: Diagrama de Casos de Uso do MENTORED *Testbed*.

papel com maior privilégio dentro de um projeto, possuindo os mesmos direitos concedidos aos papéis Gestor de Projeto e Experimentador. Compete ao Líder de Projeto delegar os papéis de Gestor de Projeto e Experimentador aos usuários do projeto

O "Gestor de Projeto" é um papel que pode ser designado a mais de um usuário que ficará responsável pela administração de usuários e experimentos. Ou seja, o Gestor de Projeto pode gerenciar membros da equipe e configurar experimentos. Por fim, o "Experimentador" é o usuário que pode realizar experimentos dentro do projeto ao qual está associado.

Cabe ressaltar que uma vez que o pesquisador possua um usuário no sistema, esse poderá participar de diversos projetos, podendo assumir diferentes papéis em cada um deles. Dessa forma, o MENTORED *Testbed* oferece uma estrutura flexível e adaptável, que permite a colaboração eficiente entre os membros da equipe.

O gerenciamento de equipes[37] traz diversas funcionalidades, como o registro de usuários pelo COmanage que simplifica o processo de entrada. Após logado no sistema, o usuário pode criar e remover requisições de novos projetos diretamente na plataforma. Além disso, a administração de membros e permissões em projetos gerencia os papéis atribuídos a cada membro. Essas funcionalidades visam fomentar uma colaboração mais

[37]A implementação do gerenciamento de equipes e suas funcionalidades encontra-se em desenvolvimento no momento.

efetiva e segura entre os membros da equipe. Adicionalmente, os usuários possuem a oportunidade de conduzir e supervisionar experimentos, acessar registros e gerenciar chaves de autenticação SSH e configurações do Kubernetes, com o objetivo de assegurar a segurança e a eficácia do sistema.

Essas funcionalidades proporcionam maior controle e flexibilidade aos usuários durante sua interação com o sistema, ao mesmo tempo em que simplificam a administração de projetos e experimentos. Com isso, o objetivo da gestão é proporcionar uma experiência mais satisfatória para todos os envolvidos no MENTORED *Testbed*.

5.3.3.4. Ciclo de vida do experimento

Conforme ilustrado na Figura 5.10, o ciclo de vida de um experimento pode ser dividido em três etapas: 1) Definição do experimento, onde o experimentador define topologias de rede, softwares e outros recursos que são especificados em uma sintaxe definida pelo MENTORED *Master*; 2) A execução do experimento, onde um experimentador seleciona uma definição de experimento e o momento em que esse experimento será iniciado; 3) As análises dos experimentos, que incluem armazenamento e processamento de *logs*, alertas e, se necessário, o monitoramento dos experimentos em tempo real.

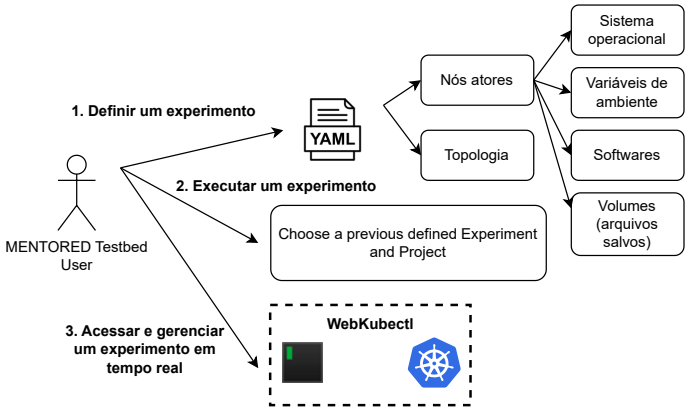
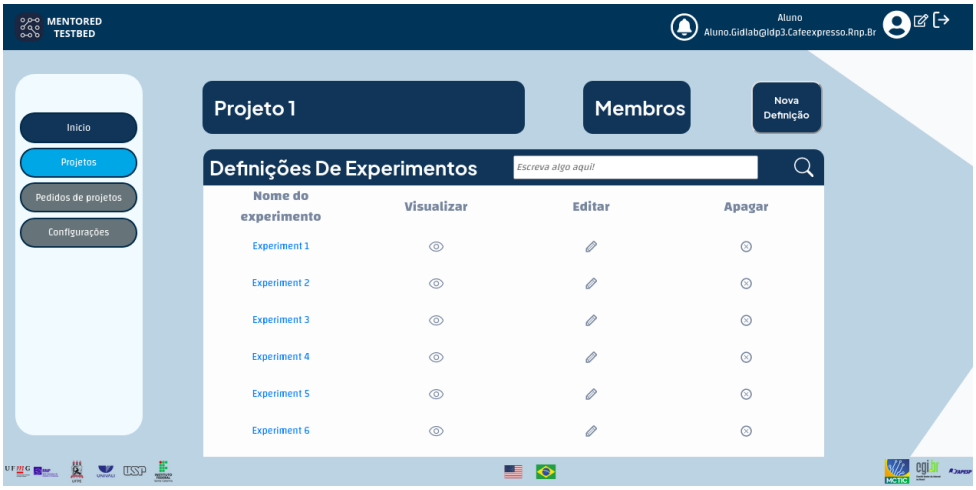


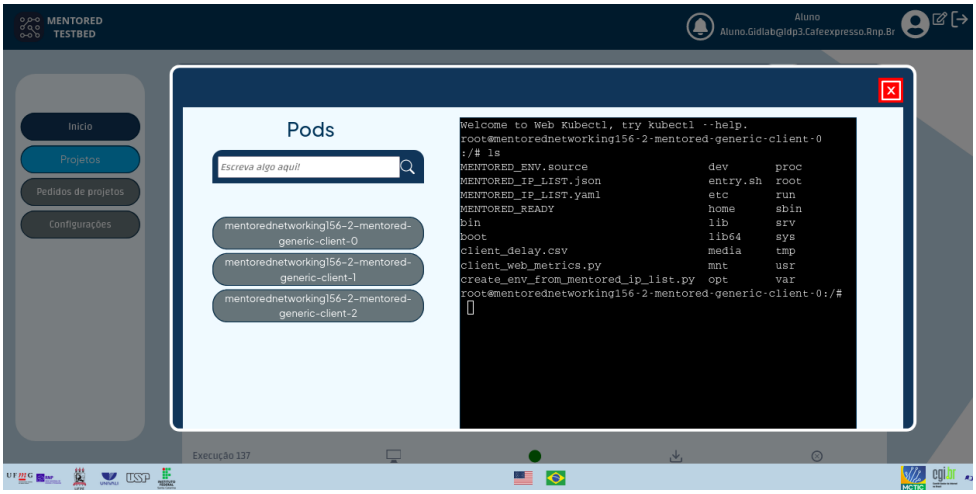
Figura 5.10: Perspectiva do usuário para definição de um experimento no MENTORED *testbed*.

A Seção 5.4, discutirá exemplos de uso práticos do ciclo de vida de experimentos, usando o MENTORED *Master* e o seu portal para executar as etapas do ciclo, considerando a perspectiva do experimentador. A Figura 5.11a ilustra um exemplo da página de definição e controle de experimentos e a Figura 5.11bilustra o monitoramento em tempo real de experimentos no Portal do MENTORED *Testbed*.

De forma complementar, em Meyer, Gemmer, Santana et al. (2024), os autores descrevem um fluxo para criação e análise de *datasets* representativos de ataque de negação de serviço, usando o MENTORED Testbed, conforme resumido na Figura 5.12.



(a) Página para controle de definições de experimentos.



(b) Página para monitoramento em tempo real de experimentos.

Figura 5.11: Páginas sobre experimentos do Portal do MENTORED *testbed*.

5.3.3.5. Aprimoramentos do MENTORED *Testbed*

A estrutura e funcionalidade do MENTORED *Testbed* têm sido submetidas a análises e melhorias contínuas, conforme documentado em diversos estudos anteriores (PRATES JR et al., 2021; MEYER; GEMMER; SCHWARZ et al., 2022; GEMMER et al., 2023). A infraestrutura do *testbed* está em constante evolução, focando não apenas na correção de falhas, mas também na introdução de novas funcionalidades que aprimoram sua eficácia e usabilidade.

Recentemente, uma significativa atualização do MENTORED *Testbed* facilitou o desenvolvimento de um procedimento eficiente para a criação de *datasets* dedicados a pesquisas de cibersegurança (MEYER; GEMMER; SANTANA et al., 2024). As inovações que estão em fase de desenvolvimento e prometem impulsionar a capacidade do *testbed* incluem:

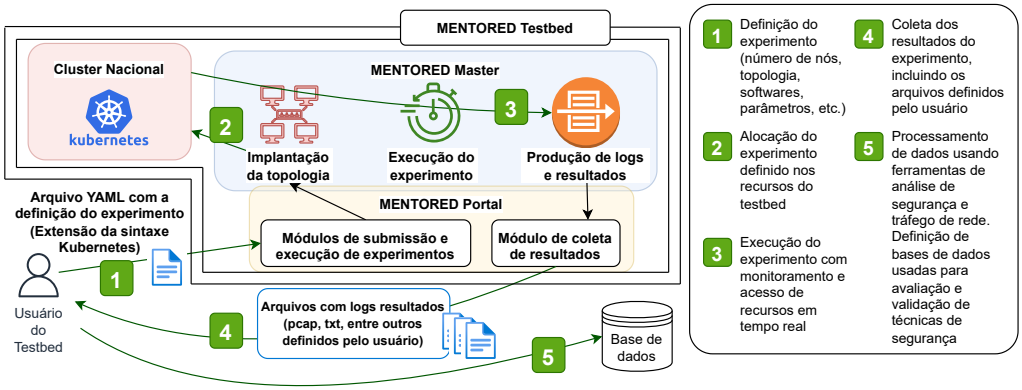


Figura 5.12: Fluxo que usa o MENTORED *testbed* para criação e análise de *datasets*. Fonte: (MEYER; GEMMER; SANTANA et al., 2024).

- **Aprimoramento na coleta de *logs* e resultados**: Os experimentos de cibersegurança no *testbed* podem envolver múltiplos nós operando softwares de maneira isolada, o que gera uma grande quantidade de dados e *logs*. Por exemplo, a simulação de ataques DDoS pode resultar na produção de *logs* de monitoramento de rede que alcançam a ordem de *gigabytes* em poucos minutos. Estão sendo desenvolvidos mecanismos avançados de monitoramento que permitirão aos usuários acompanhar os experimentos e acessar os resultados de maneira simplificada e ágil. Os principais desafios desta frente incluem a compressão eficaz de dados e a coleta de *logs* de diferentes componentes do MENTORED (como Kubernetes, softwares emulados e o MENTORED *Master*).

- **Desenvolvimento de um repositório de definições de experimentos e imagens Docker**: A efetivação de testes e a validação das capacidades do *testbed* são conduzidas através de experimentos detalhadamente definidos. Essas definições muitas vezes requerem a implementação de configurações essenciais para a execução de testes de cibersegurança, tais como a sincronização de nós e a configuração de serviços e ataques. O acervo de imagens Docker está em constante expansão e futuramente permitirá que o *testbed* ofereça funcionalidades para a publicação e compartilhamento de configurações experimentais com a comunidade de usuários.

- **Gestão de recursos e autorizações**: Conforme mencionado anteriormente, o MENTORED *Testbed* incorpora diversas tecnologias para autenticação e autorização, além do gerenciamento de equipes. Um aspecto crucial atualmente em desenvolvimento é a implementação de um sistema robusto para o controle de recursos utilizados por cada experimentador, possibilitando a definição precisa da quota de recursos disponíveis para cada equipe ou usuário individual nos seus experimentos.

- **Dispositivos IoT**: Atualmente a equipe do MENTORED *Testbed* está realizando análises para a inclusão de dispositivos IoT para experimentação. Em específico, dispositivos Raspberry Pi 4 (4G) estão sendo utilizados em um *cluster* IoT dedicado, realizando provas de conceito e validações do funcionamento da pilha de *software* utilizada pelo MENTORED *Testbed*. Ademais, foi obtido sucesso em testes preliminares de conectividade WiFi, isto é, servindo como a interface de rede

para nós modelados em um experimento. Como próximos passos, serão realizados estudos da integração desdes dispositivos com o Cluster Nacional RNP, possibilitando então a criação de experimentos abrangendo dispositivos x86 tradicionais e dispositivos IoT, simultaneamente.

5.4. Estudos de casos

Com o intuito de demonstrar a capacidade do MENTORED *Testbed*, esta seção apresenta dois estudos de caso que abordam a execução de experimentos práticos com foco em ataques DDoS[38]. As atividades práticas para este minicurso foram divididas em duas partes. A primeira atividade consiste em um ataque volumétrico utilizando a ferramenta hping3 (veja Seção 5.4.1), enquanto a segunda atividade apresenta um cenário em que são explorados problemas que servidores *Web* podem enfrentar ao lidar com milhares de conexões simultâneas, por meio de um ataque com o *Slowloris* (veja 5.4.2).

Ambas as atividades práticas são compostas pelo mesmo número de clientes, atacantes, vítima, dinâmica de ataque e tempo de execução, variando apenas o tipo de ataque e a distribuição geográfica. Como ilustrado na Figura 5.13, o ataque tem uma duração total de 300 segundos, dividido em três partes: 1) O pré-ataque ocorre nos primeiros 60 segundos, composto apenas pelo tráfego dos clientes legítimos; 2) O ataque é iniciado aos 60 segundos e tem uma duração de 180 segundos, agregando o tráfego legítimo dos clientes ao tráfego malicioso dos atacantes com o objetivo de interromper o serviço, e; 3) O período pós-ataque é composto pelos últimos 60 segundos, onde é avaliado se os clientes legítimos ainda conseguem estabelecer a conexão com o servidor *Web*.

Figura 5.13: Cronograma do Ataque.

Como descrito na Seção 5.3.2.1, a definição de um experimento no MENTORED *testbed* é realizada por meio de um código `YAML` que estende objetos do Kubernetes, acrescentando assim características exclusivas do MENTORED *testbed*. Para ambos os estudos de casos que serão apresentados nas atividades práticas, a definição do experimento foi dividida em três partes: 1) Definição do *pod* que atuará como vítima; 2) Definição dos *pods* responsáveis pelo ataque, e; 3) definição dos *pods* que atuarão como clientes legítimos e que farão requisições ao serviço disponibilizado pelo *pod* vítima.

Na Listagem 5.1 é apresentada a definição dos *pods* que atuarão como clientes legítimos que farão requisições HTTP GET ao servidor *web* hospedado no *pod* vítima. É

[38]Uma documentação e código-fonte dos softwares utilizados nos experimentos está disponível em: https://github.com/mentoredtestbed/minicurso-sbrc-2024-testbeds

possível observar que serão instanciadas 70 réplicas do cliente, cada um com uma imagem personalizada responsável por efetuar as requisições *Web* em um intervalo de 1 segundo durante 300 segundos. O tempo de resposta das requisições é salvo em um arquivo do tipo *csv*. Por fim, esse conjunto de clientes será executado na região de Goiás do *Cluster* Nacional da RNP.

```yaml
- name: 'generic-client-go'
  persitent_volume_path: "/client_delay.csv"
  replicas: 70
  containers:
    - name: 'client-go'
      image: ghcr.io/mentoredtestbed/generic-client:latest
      command: ["/entry.sh"]
      args: ['python3', 'client_web_metrics.py', "1", "1"]
      env:
        - name: TIMEOUT_CMD
          value: "300"
        - name: ADD_SERVER_IP_TO_COMMAND
          value: "true"
      resources:
        requests:
          memory: "64Mi"
          cpu: "100m"
        limits:
          memory: "128M"
          cpu: "200m"
  region: 'ids-go'
```

Listagem 5.1: YAML descrevendo o cliente *Web*.

Na Listagem 5.2 é apresentada a definição dos *pods* que atuarão como atacantes, responsáveis por realizar o ataque DDoS. É possível perceber que a definição do atacante é bastante semelhante à definição do cliente, com a diferença que é feito uso de uma imagem de contêiner configurada com os ataques utilizados. Sendo assim, é possível determinar que serão utilizados 10 atacantes com a ferramenta *slowloris* na porta 80. O ataque será iniciado aos 60 segundos e interrompido aos 180 segundos. Por fim, esse conjunto de atacantes será posicionado na região da Paraíba do *Cluster* Nacional da RNP.

5.4.1. Atividade prática I: ataque volumétrico com hping3

Nessa atividade será conduzido um ataque DDoS volumétrico por meio da ferramenta hping3[39]. O hping3 é uma ferramenta de código aberto amplamente utilizada para testes de penetração e avaliação de redes, porém também pode ser empregada de forma maliciosa em ataques DDoS. Uma das características do hping3 é a sua capacidade de enviar pacotes ICMP, UDP e TCP com diferentes *flags* e opções, o que possibilita a simulação de diversos tipos de tráfego malicioso. Além disso, o hping3 permite a falsificação de

[39]http://wiki.hping.org/home

```yaml
- name: 'generic-botnet-pb'
  persitent_volume_path: "/MENTORED_IP_LIST.yaml"
  replicas: 10
  containers:
    - name: 'botnet-pb'
      image: ghcr.io/mentoredtestbed/generic-botnet:latest
      command: ["/entry.sh"]
      args: ["slowloris", "-p", "80"]
      env:
        - name: PROTOCOL
          value: "ICMP"
        - name: TIMEOUT_CMD
          value: "180"
        - name: TIME_WAIT_START
          value: "60"
        - name: ADD_SERVER_IP_TO_COMMAND
      resources:
        requests:
          memory: "64Mi"
          cpu: "100m"
        limits:
          memory: "128M"
          cpu: "200m"
  region: 'ids-pb'
```

Listagem 5.2: YAML descrevendo o atacante *Slowloris*.

endereços IP de origem, dificultando a identificação e mitigação dos ataques. Sua flexibilidade e potência tornam-no uma escolha popular entre os atacantes que buscam realizar ataques DDoS de forma eficaz e difícil de rastrear.

Como ilustrado na Figura 5.14, o ataque será distribuído geograficamente em um ambiente real com três regiões presentes no *Cluster* Nacional da RNP. O nó vítima implementa um servidor *Web* Apache HTTPd desenvolvida com o *micro framework Web flask* padrão localizado em Recife - PE. Os nós atacantes foram divididos em duas regiões, sendo 10 atacantes em João Pessoa - PB e 10 atacantes em Goiânia - GO. Os atacantes foram configurados para utilizar o tipo de pacote TCP SYN com a opção *flood*, o *payload* dos pacotes SYN foram definidos em 1024 *bytes*. Isso significa que uma grande quantidade de pacotes SYN serão enviados para o servidor *Web* vítima, cada um com *payload* de 1024 *bytes* com o objetivo de tornar o serviço *Web* indisponível para os clientes legítimos. Os nós clientes foram posicionados nas mesmas regiões que os atacantes, com 70 clientes em João Pessoa - PB e 70 clientes em Goiânia - GO, realizando requisições GET ao servidor *Web* a cada segundo durante todo o experimento.

A Figura 5.15 representa a vazão de rede e o número de pacotes, obtidos por meio da captura de tráfego com a ferramenta *tshark* na interface de rede do servidor *Web* durante a execução do experimento. Como pode ser observado, a partir do inicio do ataque, há um aumento significativo na vazão de rede em virtude do tráfego malicioso gerado pelo hping3. Durante todo o ataque, o tráfego de rede mantém-se elevado, em linha com o número de atacantes e a capacidade do enlace disponibilizado pelo servidor *Web*. Também é possível perceber na Tabela 5.2 um aumento no tempo de resposta do servidor para os clientes legítimos logo após o início do ataque, afetando a disponibilidade do serviço. O tempo de resposta inicial é de aproximadamente 40 milissegundos, mas durante o ataque, esse tempo sobe para quase 2 segundos. Mesmo após o término do

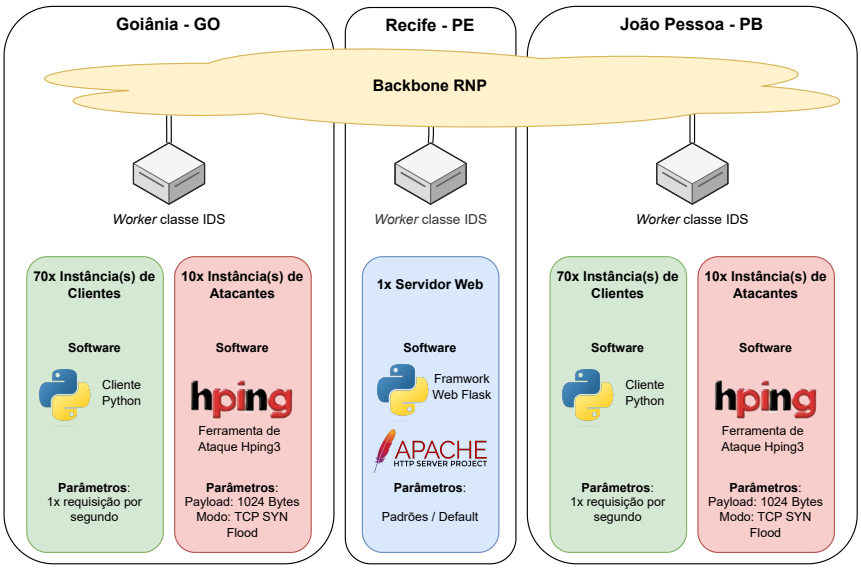


Figura 5.14: Diagrama da disposição dos clientes, atacantes e servidor - Hping.

ataque, o servidor parece enfrentar dificuldades em restabelecer sua disponibilidade pré-ataque. Isso pode ser atribuído ao uso contínuo de recursos alocados durante os ataques, que não são liberados até 60 segundos após o término do ataque.

Tabela 5.2: Resumo das médias do tempo (em segundos) de resposta de clientes na atividade I. As médias foram obtidas dentre todos os clientes para três faixas de tempo: pré ataque (0 à 60 segundos), ataque (60 à 240 segundos) e pós ataque (240 à 300 segundos).

Atividade	Tempo de resposta média dentre clientes (segundos)		
	Pré ataque	**Ataque**	**Pós ataque**
I (hping3)	0.040	1.993	0.463

Em resumo, os ataques de *Flood* DDoS representam uma ameaça significativa à disponibilidade de serviços online e requerem medidas robustas de segurança cibernética. Para mitigar um ataque de *Flood* DDoS, as organizações podem empregar várias estratégias, incluindo o uso de *firewalls*, sistemas de detecção e prevenção de intrusões (IDPS), serviços de mitigação de DDoS baseados em nuvem, balanceamento de carga, e outras técnicas de defesa cibernética. A resposta a um ataque DDoS também pode envolver a cooperação com provedores de serviços de internet (ISPs) e autoridades legais para identificar e desativar os sistemas envolvidos no ataque.

5.4.2. Atividade prática II: negação de serviço com *Slowloris*

O *Slowloris* (YALTIRAKLI, 2015) é uma ferramenta específica para ataques de negação de serviço distribuídos (DDoS) que se destaca pela sua eficácia e simplicidade. O *Slowloris* explora uma vulnerabilidade no protocolo HTTP, aproveitando-se da característica de algumas implementações de servidores *Web* que mantém conexões abertas por longos períodos de tempo.

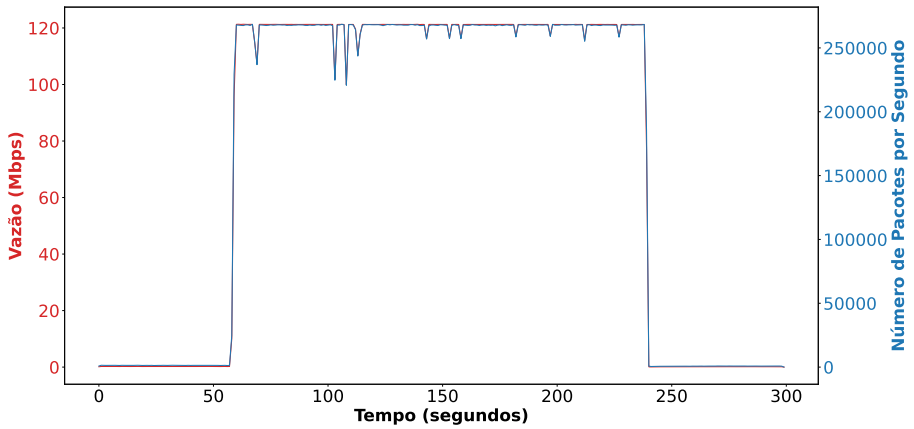


Figura 5.15: Vazão de rede e número de pacotes com o ataque hping.

O modo de operação do *Slowloris* consiste em no envio de solicitações HTTP incompletas, mantendo as conexões abertas com o servidor, mas não enviando a solicitação completa. Isso faz com que o servidor aloque recursos para a conexão aberta, aguardando a conclusão da requisição, enquanto simultaneamente limita o número de conexões que pode atender, eventualmente, levando à sobrecarga e à interrupção do serviço para usuários legítimos. O *Slowloris* é notável por sua capacidade de realizar ataques eficazes com poucos recursos, tornando-o uma escolha popular entre os cibercriminosos. No entanto, trata-se de um ataque amplamente conhecido e sua detecção e mitigação tornaram-se mais comuns à medida que os administradores de sistemas adotam medidas para proteger seus servidores contra esse tipo de ataque. O uso do *Slowloris* em um ambiente controlado é uma forma eficaz de demonstrar os efeitos de um ataque DDoS na camada de aplicação e as estratégias de defesa que podem ser empregadas para mitigá-lo.

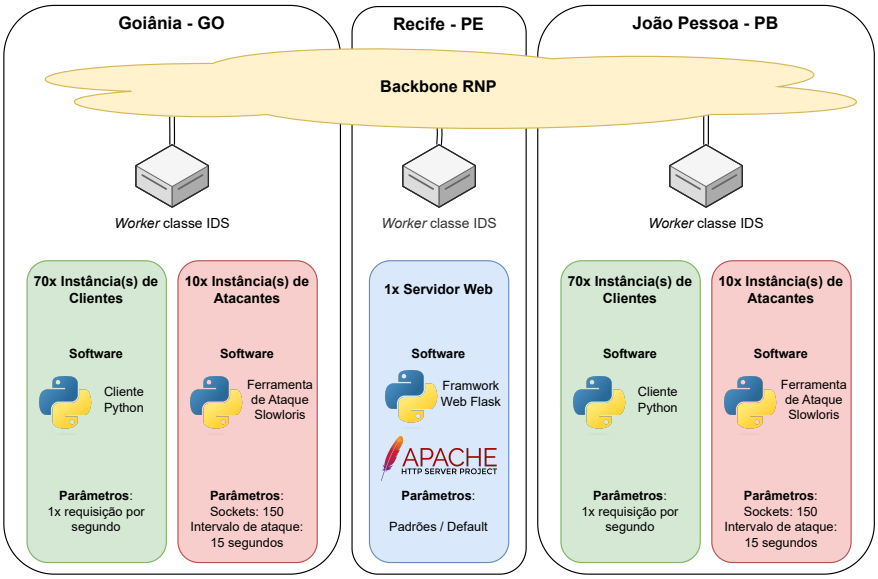


Figura 5.16: Diagrama da disposição dos clientes, atacantes e servidor - *Slowloris*.

Como ilustrado na Figura 5.16 o ataque será distribuído geograficamente em um

ambiente real com três regiões presentes no *Cluster* Nacional - RNP. O nó vítima implementa um servidor *Web* Apache HTTPd desenvolvida com o *micro framework Web flask* padrão localizado em Recife - PE. Os nós atacantes foram divididos em duas regiões, sendo 10 atacantes em João Pessoa - PB e 10 atacantes em Goiânia - GO. Os atacantes executam a implementação `slowloris.py` (YALTIRAKLI, 2015) com paramentos padrão da ferramenta, sendo 150 o número de conexões paralelas abertas simultaneamente em um intervalo de 15 segundos entre o envio de cada cabeçalho na porta 80. Os clientes efetuam requisições GET no intervalo de *1* segundo ao servidor *Web* durante todo o experimento.

A Figura 5.17 representa a vazão de rede e o número de pacotes, esses resultados foram obtidos por meio da captura de tráfego com a ferramente *tshark* na interface de rede do servidor *Web* durante a execução do experimento. Como pode ser observado, existe um pico na vazão de rede, a partir do inicio do ataque, logo resultando na indisponibilidade do serviço. Também é possível observar que diferente do experimento com o hping3, em que durante o período de ataque foi atingido um determinado limite na vazão de rede que se manteve regular e interrupto, com o *Slowloris* ocorreram picos na vazão de rede em intervalos de 15 segundos, mesmo intervalo de tempo que a ferramenta utiliza como padrão para enviar novas requisições. Mesmo esse ataque não ter sobrecarregado a rede, é possível perceber na Tabela 5.3 que ocorreu a indisponibilidade do serviço para os clientes legítimos em virtude do *Slowloris* sobrecarregar o servidor *Web*. Em específico, o *Slowloris* foi capaz de aumentar em média o tempo de resposta do servidor de 40 milissegundos para 8.6 segundos, um ataque muito mais eficiente do que o hping3. Por fim, é possível identificar que mesmo após a conclusão do ataque a maior parte dos clientes legítimos não obtiveram mais resposta em virtude do servidor *Web* ainda estar sobrecarregado com as conexões que foram abertas durante o ataque.

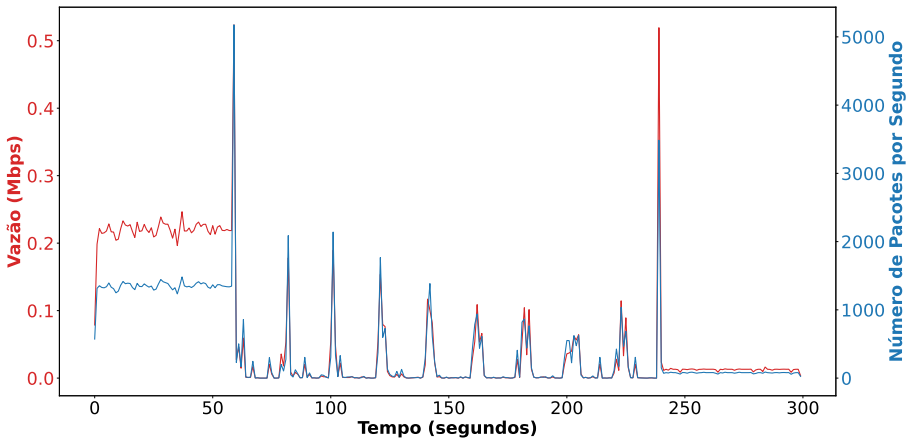


Figura 5.17: Vazão de rede e número de pacotes com o ataque *slowloris*.

Como pôde ser observado, o *Slowloris* é eficaz contra servidores HTTP que mantêm conexões abertas por longos períodos de tempo e não possuem mecanismos eficazes de limitação de conexões por IP ou de detecção de atividades suspeitas. No entanto, essa estratégia tem como base a camada 7 do modelo OSI (aplicação) e muitos servidores e sistemas de proteção contra DoS implementam contramedidas para mitigar esse tipo de ataque, como por exemplo, limitação no número de conexões por IP, fechamento de co-

nexões inativas, e o uso de *firewalls* de aplicativos *Web* (WAFs) que podem detectar e bloquear padrões de tráfego associados ao *Slowloris*. Essa característica não acontece em ataques DoS ou DDoS como os que usam o hping3 e que utilizam a camada 4 (transporte) como base do ataque, tornando complexa a tarefa de diferenciar tráfego de rede legítimo de tráfego de rede malicioso.

Tabela 5.3: Resumo das médias do tempo (em segundos) de resposta de clientes na atividade II. As médias foram obtidas dentre todos os clientes para três faixas de tempo: pré ataque (0 à 60 segundos), ataque (60 à 240 segundos) e pós ataque (240 à 300 segundos).

Atividade	Tempo de resposta média dentre clientes (segundos)		
	Pré ataque	**Ataque**	**Pós ataque**
II (*slowloris*)	0.040	8.618	0.070

5.5. Considerações finais

A crescente complexidade e evolução das ameaças cibernéticas exigem uma abordagem robusta e dinâmica para a pesquisa e desenvolvimento de soluções de segurança. Os *testbeds* desempenham um papel crucial neste contexto, fornecendo ambientes controlados e reprodutíveis que facilitam a experimentação em cibersegurança. Este capítulo destacou diversos *testbeds* especializados, ilustrando suas capacidades únicas, variando desde a simulação de dispositivos IoT até a execução de sofisticados ataques DDoS e técnicas de mitigação.

Por meio dos exemplos detalhados, como o MENTORED *testbed*, DeterLab e FIT IoT-LAB, observa-se a importância dessas infraestruturas no avanço do conhecimento científico e técnico na área. Cada *testbed* apresenta características distintas que atendem a diferentes requisitos de pesquisa, desde a escalabilidade até a fidelidade dos experimentos, refletindo a diversidade de necessidades no campo da cibersegurança.

A implementação de *testbeds* como o MENTORED oferece uma visão sobre como a combinação de tecnologias avançadas e um design estratégico pode resultar em uma plataforma poderosa para testar e validar novas teorias, protocolos de segurança e estratégias de defesa. A colaboração e gestão eficiente de equipes e recursos, aliadas à capacidade de reproduzir cenários realistas, são essenciais para o sucesso dos projetos de pesquisa. Em específico, a implementação do MENTORED *testbed* se destaca no contexto brasileiro devido à sua simplicidade para definir, executar e monitor experimentos por meio de um portal, além da sua infraestrutura baseada no *Cluster* Nacional da RNP.

Como perspectivas futuras para experimentação em cibersegurança, pode-se citar a utilização de tecnologias como aprendizado de máquina e *blockchain*, além de tecnologias emergentes como realidade aumentada e computação quântica, em conjunto com as tecnologias já estabelecidas, bem como a integração com ferramentas modulares relacionadas à segurança. Assim, além dos experimentos atuais de simulação de ataques e defesa, poderá ser ofertado suporte a experimentos mais complexos, como os que envolvem detecção automatizada de ataques, resposta à incidentes automatizada, experimentos de proteção de dados, interoperabilidade segura entre sistemas, privacidade de dados, e gestão de informações sensíveis.

Finalmente, os *testbeds* não só avançam nossa compreensão das ameaças cibernéticas existentes e emergentes, mas também desempenham um papel importante na educação e formação de futuros profissionais de segurança. À medida que enfrentamos desafios cada vez mais sofisticados, a importância desses ambientes experimentais continua a crescer, promovendo um ciclo de inovação contínua que é crucial para manter a segurança e integridade dos sistemas presentes em aplicações reais. Portanto, esses ambientes experimentais são fundamentais para enfrentar os desafios de cibersegurança considerando o atual estado da arte da pesquisa e tecnologias.

Agradecimentos

Este trabalho foi financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (#2018/23098-0 e #2023/06265-8), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES (PROSUC), Rede Nacional de Ensino e Pesquisa (RNP) e Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (#141179/2021-0). Os autores utilizaram a ferramenta ChatGPT para revisar a redação de algumas partes deste capítulo.

Referências

ADJIH, Cedric et al. FIT IoT-LAB: A large scale open experimental IoT testbed. In: IEEE. 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). 2015. P. 459–464.

ALSAEDI, Abdullah et al. TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. **Ieee Access**, IEEE, v. 8, p. 165130–165150, 2020.

BALDIN, Ilya et al. Fabric: A national-scale programmable experimental network infrastructure. **IEEE Internet Computing**, IEEE, v. 23, n. 6, p. 38–47, 2019.

BENZEL, Terry. The Science of Cyber Security Experimentation: The DETER Project. In: PROCEEDINGS of the 27th Annual Computer Security Applications Conference. Orlando, Florida, USA: Association for Computing Machinery, 2011. (ACSAC '11), p. 137–148. ISBN 9781450306720. DOI: 10.1145/2076732.2076752. Disponível em: <https://doi.org/10.1145/2076732.2076752>.

BERMAN, Mark et al. GENI: A federated testbed for innovative network experiments. **Computer Networks**, Elsevier, v. 61, p. 5–23, 2014.

BUDIGIRI, Gerald et al. Network policies in kubernetes: Performance evaluation and security analysis. In: IEEE. 2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit). 2021. P. 407–412.

CHERNYSHEV, Maxim et al. Internet of things (iot): Research, simulators, and testbeds. **IEEE Internet of Things Journal**, IEEE, v. 5, n. 3, p. 1637–1647, 2017.

CHOULIARAS, Nestoras et al. Cyber Ranges and TestBeds for Education, Training, and Research. **Applied Sciences**, v. 11, n. 4, 2021. ISSN 2076-3417. DOI: 10.3390/app11041809. Disponível em: <https://www.mdpi.com/2076-3417/11/4/1809>.

CHUN, Brent et al. Planetlab: an overlay testbed for broad-coverage services. **ACM SIGCOMM Computer Communication Review**, ACM New York, NY, USA, v. 33, n. 3, p. 3–12, 2003.

CLAASSEN, Joris; KONING, Ralph; GROSSO, Paola. Linux containers networking: Performance and scalability of kernel modules. In: IEEE. NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium. 2016. P. 713–717.

CUNHA PONTES, Edgard da et al. FABRIC Testbed from the Eyes of a Network Researcher. In: SBC. ANAIS do II Workshop de Testbeds. 2023. P. 38–49.

DEMEESTER, Piet et al. Fed4fire–the largest federation of testbeds in europe. In: BUILDING the future internet through FIRE. River Publishers, 2022. P. 87–109.

FLANAGAN, Heather et al. Enabling efficient electronic collaboration between LIGO and other astronomy communities using federated identity and COmanage. In: SPIE. SOFTWARE and Cyberinfrastructure for Astronomy II. 2012. v. 8451, p. 530–546.

FORUM, World Economic. **Global Risks Report 2024**. Jan. 2024. https://www.weforum.org/publications/global-risks-report-2024/.

GEMMER, Davi Daniel et al. A Scalable Cyber Security Framework for the Experimentation of DDoS Attacks of Things. In: PROCEEDINGS of IEEE/IFIP Network Operations and Management Symposium (NOMS). Miami, FL, EUA: IEEE/IFIP, mai. 2023.

GOMEZ, Jose et al. A survey on network simulators, emulators, and testbeds used for research and education. **Computer Networks**, v. 237, p. 110054, 2023. ISSN 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2023.110054. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128623004991>.

GOODFELLOW, Ryan; THURLOW, Lincoln; RAVI, Srivatsan. Merge: An Architecture for Interconnected Testbed Ecosystems. **CoRR**, abs/1810.08260, 2018. arXiv: 1810.08260. Disponível em: <http://arxiv.org/abs/1810.08260>.

KORONIOTIS, Nickolaos; MOUSTAFA, Nour; SCHILIRO, Francesco et al. The sair-iiot cyber testbed as a service: A novel cybertwins architecture in iiot-based smart airports. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 24, n. 2, p. 2368–2381, 2021.

KORONIOTIS, Nickolaos; MOUSTAFA, Nour; SITNIKOVA, Elena et al. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. **Future Generation Computer Systems**, Elsevier, v. 100, p. 779–796, 2019.

MEYER, Bruno Henrique; GEMMER, Davi Daniel; SANTANA, Khalil G. Q. de et al. Criação e análise de *datasets* de ataque de negação de serviço usando o Mentored Testbed. In: SBC. ANAIS do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. 2024.

MEYER, Bruno Henrique; GEMMER, Davi Daniel; SCHWARZ, Marcos et al. MENTORED: The Brazilian Cybersecurity Testbed. In: DEMO sessions of IEEE Global Communications Conference (GLOBECOM). Rio de Janeiro, RJ: IEEE, dez. 2022.

MIRKOVIC, Jelena; BENZEL, Terry. Teaching cybersecurity with DeterLab. **IEEE Security & Privacy**, IEEE, v. 10, n. 1, p. 73–76, 2012.

MOUSTAFA, Nour. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. **Sustainable Cities and Society**, Elsevier, v. 72, p. 102994, 2021.

NGUYEN, Xuan et al. **Network isolation for Kubernetes hard multi-tenancy**. 2020. Diss. (Mestrado).

PRATES JR, Nelson G et al. Um ambiente de experimentaçao em cibersegurança para internet das coisas. In: SBC. ANAIS do VI Workshop do testbed FIBRE. 2021. P. 68–79.

SÁEZ-DE-CÁMARA, Xabier et al. Gotham testbed: a reproducible IoT testbed for security experiments and dataset generation. **IEEE Transactions on Dependable and Secure Computing**, IEEE, 2023.

SAYFAN, Gigi. **Mastering kubernetes**. Packt Publishing Ltd, 2017.

SCHWAB, Stephen; KLINE, Erik. Cybersecurity experimentation at program scale: Guidelines and principles for future testbeds. In: IEEE. 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). 2019. P. 94–102.

SIATERLIS, Christos; GARCIA, Andres Perez; GENGE, Bela. On the Use of Emulab Testbeds for Scientifically Rigorous Experiments. **IEEE Communications Surveys & Tutorials**, v. 15, n. 2, p. 929–942, 2013. ISSN 1553-877X. DOI: 10.1109/SURV.2012.0601112.00185. Disponível em: <http://ieeexplore.ieee.org/document/6226792/>. Acesso em: 24 abr. 2023.

SIATERLIS, Christos; GENGE, Bela; HOHENADEL, Marc. EPIC: A testbed for scientifically rigorous cyber-physical security experimentation. **IEEE Transactions on Emerging Topics in Computing**, IEEE, v. 1, n. 2, p. 319–330, 2013.

SIBONI, Shachar et al. Security testbed for Internet-of-Things devices. **IEEE transactions on reliability**, IEEE, v. 68, n. 1, p. 23–44, 2018.

THOM, Jay et al. Casting a wide net: An internet of things testbed for cybersecurity education and research. In: IEEE. 2021 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). 2021. P. 1–8. DOI: 10.23919/SPECTS52716.2021.9639278. DOI: https://doi.org/10.23919/SPECTS52716.2021.9639278.

WROCLAWSKI, John et al. DETERLab and the DETER Project. **The GENI Book**, Springer, p. 35–62, 2016.

YALTIRAKLI, Gokberk. **Low bandwidth DoS tool. Slowloris rewrite in Python**. 2015. https://github.com/gkbrk/slowloris.