UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

LEYDSON MAYCKSON FERRAZ DE BARROS

EXPLORANDO A PROGRAMABILIDADE ADAPTATIVA E COLABORATIVA: UMA ANÁLISE DA INFRAESTRUTURA DE PESQUISA FABRIC PARA AVANÇOS EM REDES DE COMPUTADORES E CIÊNCIA DE DADOS

Recife-PE
2024

LEYDSON MAYCKSON FERRAZ DE BARROS

EXPLORANDO A PROGRAMABILIDADE ADAPTATIVA E COLABORATIVA: UMA ANÁLISE DA INFRAESTRUTURA DE PESQUISA FABRIC PARA AVANÇOS EM REDES DE COMPUTADORES E CIÊNCIA DE DADOS

Monografia apresentada ao Centro de Informática (CIn) da Universidade Federal de Pernambuco (UFPE), como um requisito parcial para conclusão do Curso de Engenharia da Computação.

Orientador: Prof. José Augusto Suruagy Monteiro.

Recife-PE
2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Ferraz de Barros, Leydson Mayckson .
EXPLORANDO A PROGRAMABILIDADE ADAPTATIVA E COLABORATIVA: UMA ANÁLISE DA INFRAESTRUTURA DE PESQUISA FABRIC PARA AVANÇOS EM REDES DE COMPUTADORES E CIÊNCIA DE DADOS / Leydson Mayckson Ferraz de Barros. - Recife, 2024.
59p : il., tab.

Orientador(a): José Augusto Suruagy Monteiro
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2024.
Inclui referências.

1. Fabric. 2. programabilidade adaptativa. 3. redes de computadores. 4. ciência de dados. 5. infraestrutura de pesquisa. I. Suruagy Monteiro, José Augusto . (Orientação). II. Título.

000 CDD (22.ed.)

LEYDSON MAYCKSON FERRAZ DE BARROS

EXPLORANDO A PROGRAMABILIDADE ADAPTATIVA E COLABORATIVA: UMA ANÁLISE DA INFRAESTRUTURA DE PESQUISA FABRIC PARA AVANÇOS EM REDES DE COMPUTADORES E CIÊNCIA DE DADOS

Monografia apresentada ao Centro de Informática
(CIn) da Universidade Federal de Pernambuco (UFPE),
como um requisito parcial para conclusão do Curso de
Engenharia da Computação.

.

Aprovado em: 17/10/2024.

BANCA EXAMINADORA

_____
Prof. Dr.José Augusto Suruagy Monteiro (Orientadora)
Universidade Federal de Pernambuco - UFPE

_____
Prof. Dr. Divanilson Rodrigo de Sousa Campelo (Examinador Interno)
Universidade Federal de Pernambuco - UFPE

AGRADECIMENTOS

Primeiramente, agradeço a Deus, que me deu força, inteligência e sabedoria para trilhar essa caminhada, sempre guiando meus passos. Nessa jornada, jamais estive só. Agradeço à minha família, que é minha base, especialmente à minha mãe e à minha tia, que nunca duvidaram do meu potencial. Agradeço também às pessoas que conheci ao longo do caminho, que me fortaleceram nos momentos difíceis. Em particular, aos meus amigos de curso, que sempre estiveram ao meu lado e contribuíram para o meu crescimento.

Agradeço ainda a todos da Pós-Graduação em Letras da UFPE, onde atuei como bolsista, especialmente ao professor José Alberto Miranda Poza, que desde o início acreditou e confiou na minha competência, oferecendo conselhos e orientações valiosos. Também sou grato ao meu orientador, José Augusto Suruagy Monteiro, pela paciência e pela orientação na construção deste trabalho, sem a qual eu não teria alcançado este momento.

Esta conquista é minha, mas a vitória é de todos!

Resumo

Este trabalho tem como foco o estudo da plataforma de pesquisa Fabric, que oferece uma infraestrutura avançada para experimentos em redes de computadores e ciência de dados, permitindo a realização de experimentos em larga escala, testes de novos protocolos e análise de grandes volumes de dados, sendo uma ferramenta poderosa para pesquisadores. O estudo detalha como a infraestrutura do Fabric é composta por elementos como CPUs, GPUs, FPGAs e conectividade óptica, o que possibilita uma alta capacidade de processamento e armazenamento. Esse trabalho explora o uso do Fabric em projetos práticos, discute os desafios encontrados pelos pesquisadores, como a complexidade na configuração e o uso dos recursos da plataforma, e apresenta uma análise das suas vantagens e limitações. Além disso, é feita uma revisão sistemática da literatura para investigar o impacto da plataforma nas pesquisas de redes de computadores e ciência de dados, destacando o seu papel em avanços científicos e tecnológicos. A pesquisa inclui uma demonstração prática de implementação de uma arquitetura de rede utilizando o Fabric, ilustrando seu potencial e aplicabilidade. Por fim, o trabalho conclui que, embora o Fabric apresente desafios técnicos e uma curva de aprendizado acentuada, ele se configura como uma plataforma essencial para o avanço de pesquisas em redes de computadores e ciência de dados, proporcionando um ambiente robusto para experimentação e inovação.

Palavras-chave: Fabric, programabilidade adaptativa, redes de computadores, ciência de dados, infraestrutura de pesquisa.

Abstract

This paper focuses on the study of the Fabric research platform, which offers an advanced infrastructure for experiments in computer networks and data science, enabling large-scale experiments, testing of new protocols, and analysis of large data volumes, making it a powerful tool for researchers. The study details how Fabric's infrastructure is composed of elements such as CPUs, GPUs, FPGAs, and optical connectivity, allowing for high processing and storage capacity. This work explores the use of Fabric in practical projects, discusses the challenges faced by researchers, such as the complexity in configuring and utilizing the platform's resources, and presents an analysis of its advantages and limitations. Furthermore, a systematic literature review is conducted to investigate the platform's impact on research in computer networks and data science, highlighting its role in scientific and technological advancements. The research includes a practical demonstration of implementing a network architecture using Fabric, illustrating its potential and applicability. Finally, the paper concludes that although Fabric presents technical challenges and a steep learning curve, it stands as an essential platform for advancing research in computer networks and data science, providing a robust environment for experimentation and innovation.

Keywords: Fabric, adaptive programmability, computer networks, data science, research infrastructure.

Lista de Figuras

Lista de Tabelas

Sumário

Páginas

1 Introdução

Neste capítulo, apresenta-se o contexto no qual o tema abordado no estudo está inserido, juntamente com a razão e os propósitos almejados. Adicionalmente, é delineada a organização deste trabalho.

1.1 Motivação

O Fabric é um ambiente para experimentação (*testbed*) que oferece uma infraestrutura avançada e recursos poderosos para pesquisas em uma ampla gama de áreas. A escolha do Fabric como tema para o Trabalho de Conclusão de Curso reflete não apenas a relevância e a complexidade dessa plataforma, mas também demonstra o compromisso em contribuir para o avanço do conhecimento em uma área em constante evolução.

Como uma plataforma altamente escalável e flexível, o Fabric oferece um ambiente propício para experimentação e análise de dados em larga escala. Sua arquitetura avançada e recursos abrangentes permitem a realização de experimentos complexos e a investigação de problemas reais em redes de computadores e ciência de dados.

1.2 Objetivo

O objetivo deste trabalho é explorar a plataforma Fabric, discutindo sua definição, os impactos positivos que oferece, além das dificuldades encontradas em sua aplicação em pesquisas e projetos nas áreas de redes de computadores e ciência de dados. Ao aprofundar as nuances dessa plataforma, buscamos não apenas compreender suas funcionalidades e recursos, mas também avaliar suas aplicações práticas, os desafios enfrentados e suas contribuições para o avanço da pesquisa. Assim, pretendemos fornecer uma documentação clara e uma revisão sistemática da literatura, sintetizando os principais desafios e benefícios relatados por outros pesquisadores ao utilizarem o Fabric.

Com isso em mente, este trabalho traz uma explicação sobre o Fabric, incluindo uma aplicação prática de sua utilização. Além disso, respondemos a duas questões fundamentais: **Quais são os impactos do Fabric nas pesquisas em redes de computadores e ciência de dados?** e **Quais são as principais dificuldades encontradas ao utilizá-lo em projetos de pesquisa?**.

Em última análise, este trabalho busca fornecer uma análise focada do potencial do Fabric como uma ferramenta poderosa para pesquisa em redes de computadores e ciência de dados. Ao explorarmos suas capacidades, desafios e aplicações, esperamos oferecer *insights* valiosos que possam informar e inspirar futuras pesquisas nesta área dinâmica e vital para o avanço da tecnologia.

1.3 Estrutura do documento

Para facilitar a leitura e compreensão do trabalho, a organização foi feita em cinco capítulos, sendo eles:

- **Capítulo 2:** Este capítulo faz uma breve introdução sobre várias plataformas de experimentação e infraestrutura de pesquisa, tomando como foco o Fabric.
- **Capítulo 3:** Este capítulo apresenta uma aplicação prática do desenvolvimento de um projeto no Fabric, além de fornecer um guia para auxiliar outros desenvolvedores na criação de seus próprios projetos na plataforma.
- **Capítulo 4:** Neste capítulo, apresenta-se uma descrição do método utilizado para conduzir a pesquisa desse trabalho. Aqui, são delineadas as etapas envolvidas na pesquisa, os repositórios digitais nos quais os termos de busca foram aplicados, as questões de pesquisa, os termos de busca utilizados e o processo de seleção dos estudos primários.
- **Capítulo 5:** Neste capítulo, são apresentados os resultados da revisão sistemática e do estudos da plataforma Fabric, oferecendo ao leitor as respostas para as questões de pesquisa previamente estabelecidas e uma projeto demostrando sua aplicabilidade. Essas respostas são derivadas da análise minuciosa dos artigos selecionados e do conteúdo oferecido pela base de conhecimento do Fabric.
- **Capítulo 6:** Neste capítulo, são expostas as conclusões finais decorrentes dos resultados obtidos na pesquisa. Adicionalmente, é fornecida uma análise sucinta sobre potenciais direções para trabalhos futuros.

2 Redes para Experimentação

Este capítulo tem como principais objetivos fornecer as definições necessárias sobre diferentes redes de experimentação, com ênfase no Fabric, detalhando seus componentes e incluindo uma demonstração prática da implementação de uma arquitetura de rede utilizando a plataforma.

2.1 Redes para Experimentação

Uma rede para experimentação ou um *testbed* é essencial para a verificação e validação de modelos computacionais emergentes e estruturas de avaliação quantitativa, interrupção e processos de recuperação, sendo plataforma ou ambiente virtual projetado para simular, projetar e analisar eventos adversos. Ele reúne conjuntos de dados, modelos computacionais e ferramentas de análise que funcionam como um espaço virtual onde cenários de desastres são criados e explorados, permitindo que pesquisadores avaliem estratégias de mitigação, resposta e recuperação. Além disso, é utilizado para validar modelos e técnicas de resiliência comunitária antes de sua aplicação no mundo real (ENDERAMI et al., 2023).

Essencialmente, um *testbed* é um ambiente experimental que permite a realização de testes e avaliações por cientistas, engenheiros ou pesquisadores. Ele oferece uma variedade de *hardware* real e virtualizado para conduzir experimentos em diversas áreas de pesquisa. Esses *testbeds* podem ser compartilhados entre usuários ou específicos para projetos individuais. Nesse contexto, abrangem áreas como Redes de Comunicação, Computação em Nuvem, Segurança Cibernética, Internet das Coisas, Robótica, Sistemas Embarcados, entre outros. Usados para validar a funcionalidade e desempenho de novos protocolos, sistemas de segurança e serviços de rede em condições realistas. Funcionam como laboratórios avançados nos quais cientistas e engenheiros podem aprimorar e ajustar suas experiências, garantindo resultados mais seguros e eficientes.

Embora ofereçam resultados repetíveis e isolamento de recursos, os *testbeds* também têm limitações, como restrições de topologia e compatibilidade de *hardware*. São instalações de rede virtualizadas construídas a partir de uma federação coordenada de redes para experimentação que incorporam diversos tipos de *hardware*.

Dentre as redes para experimentação mais conhecidas temos as seguintes:

- **Emulab:** É uma plataforma de teste de redes que oferece diversos ambientes para pesquisadores desenvolverem, depurarem e avaliarem sistemas. É uma instalação física e um *software*, gerido pelo Flux Group, com mais de 24 instalações globais. Também serve como base para outras plataformas como GENI, PRObE e DETER. Usada amplamente em redes e sistemas distribuídos, também apoia o ensino nessas áreas. É gratuita e oferece ambientes para emulação de topologias de rede, experimentos na internet com testbeds como GENI e PlanetLab, redes sem fio 802.11, e rádios definidos por *software*. Todos esses ambientes são unificados em uma interface comum. https://www.emulab.net/classic.php.

- **GENI (Global Environment for Network Innovations):** Foi um laboratório virtual para pesquisa e ensino em redes e sistemas distribuídos. Ele permitiu explorar redes em grande escala, promovendo inovações em ciência de redes, segurança, serviços e aplicativos, é ideal para experimentos de larga escala, conectividade sem IP, programação profunda de redes, reprodutibilidade de resultados e acesso a ferramentas de medição e instrumentação. O GENI encerrou suas operações em 1º de agosto de 2023, sendo integrado ao Fabric. https://www.geni.net/.

- **StarBED:** É um simulador de Internet em grande escala criado pelo Instituto Nacional de Tecnologia da Informação e Comunicação (NICT) em 2002, com o objetivo de apoiar pesquisas e desenvolvimento na área de redes. Ele é composto por centenas de servidores interconectados e permite a realização de simulações e testes de redes em um ambiente controlado. Desde seu início, o projeto passou por várias fases, ampliou o escopo para testar redes ubíquas, com mais de 1.000 servidores, incluiu a próxima geração de redes, segurança, redes mistas e sistemas ciberfísicos e focou na era da IoT, avaliando PCs, celulares e sensores, considerando fenômenos físicos. A plataforma, que atualmente está fase StarBED5, a plataforma é voltada para novos dispositivos, sistemas de TIC (Tecnologia da Informação e Comunicação), redes sem fio avançadas e suporte à pesquisa e desenvolvimento, com foco na emulação de ambientes IoT/CPS e benchmarking de sistemas complexos. https://starbed.nict.go.jp/en/.

- **CloudLab:** Oferece aos pesquisadores controle total sobre os servidores físicos (bare metal), permitindo a criação de nuvens inteiras em poucos minutos. Ele oferece isolamento para diferentes usuários, possibilitando centenas de experimentos simultâneos sem interferência. Os pesquisadores podem usar *software* de nuvem como OpenStack, Hadoop e Kubernetes ou criar suas próprias soluções. O CloudLab utiliza tecnologias do Emulab e GENI, fornecendo uma interface familiar. Com cerca de 1.000 máquinas em três locais nos EUA (Utah, Wisconsin e Carolina do Sul), além de acesso a outras instalações, ele está conectado à infraestrutura da Internet2, permitindo a extensão de redes privadas. https://www.cloudlab.us/.

- **Chameleon:** É uma plataforma experimental de grande escala, altamente reconfigurável, criada para apoiar pesquisas em sistemas de Ciência da Computação. Projetos na plataforma incluem novas arquiteturas de sistemas operacionais, métodos de virtualização, gestão de energia, redes definidas por *software*, inteligência artificial e gerenciamento de recursos.

  O Chameleon permite reconfiguração de *hardware*, oferecendo controle total sobre a pilha de *software*, incluindo privilégios de *root* e customização do *kernel*. Ele também oferece uma pequena porção virtualizada usando KVM para projetos que não exigem tanto isolamento.

A plataforma conta com quase 15.000 núcleos e 5PB de armazenamento, distribuídos entre a Universidade de Chicago e o TACC (Texas Advanced Computing Center), conectados por uma rede de 100 Gbps. O *hardware* inclui uma variedade de tecnologias, como *switches* reconfiguráveis, FPGAs, GPUs e suporte a arquiteturas x86 e ARM.

Chameleon utiliza OpenStack como tecnologia principal, oferecendo uma interface familiar e colaborando com a comunidade de desenvolvedores que ultilizam o Chameleon. A plataforma, lançada em 2015, já apoiou mais de 3.000 usuários em mais de 500 projetos. https://www.chameleoncloud.org/.

- **P4Campus:** É uma iniciativa voltada para redes de campus que tem como objetivo transformar aplicações desenvolvidas em P4 (uma linguagem de programação usada para definir como *switches* e roteadores processam pacotes de dados) de provas de conceito para aplicações de nível de produção. A ideia é levar essas aplicações programáveis para rodarem em *switches* de *hardware* reais, em vez de apenas em simuladores de *software*, ampliando o impacto e a aplicabilidade dessas tecnologias. O P4Campus também promove a troca de código, boas práticas e experiências entre os participantes da comunidade. https://p4campus.cs.princeton.edu/.

São ferramentas essenciais para a pesquisa e desenvolvimento de redes de computadores, fornecendo um ambiente realista para experimentação que ajuda a impulsionar a inovação e a educação na área de tecnologia com o seu realismo e escalabilidade.

2.2 O ambiente Fabric

O Fabric é uma Infraestrutura de Pesquisa Programável e Adaptável para Ciência da Computação e Aplicações Científicas. Ele possui um ambiente de teste composto por elementos de redes programáveis equipados com recursos de computação e armazenamento, além de interconexões de links ópticos dedicados de alta velocidade. A programabilidade do Fabric, combinada com a capacidade de medição de alta fidelidade, fornece uma infraestrutura de pesquisa flexível para apoiar a pesquisa aplicada em redes e o desenvolvimento de fluxos de trabalho científicos (FABRIC. . . , 2021).

O objetivo da plataforma é apoiar a maior variedade possível de experimentos, desde conceitos inovadores para o futuro da Internet até o suporte a novos fluxos de trabalho científicos em produção. A plataforma oferece conectividade sob demanda com redes e infraestruturas de produção, permitindo que os experimentadores combinem recursos e experimentos de maneira que melhor se adequem aos seus projetos. Para um entendimento mais aprofundado, foi realizada uma análise detalhada da base de conhecimento do Fabric, com o objetivo de aprender seus conceitos fundamentais e explorar os recursos que a plataforma oferece. Foram encontrados artigos que abordam desde os conceitos introdutórios e orientações para o primeiro acesso até a criação e o gerenciamento de projetos. Esses materiais fornecem uma visão abrangente dos

componentes e conceitos que sustentam a infraestrutura e a operação do Fabric, incluindo a definição de recursos, bem como a descrição dos agentes e processos envolvidos na gestão e utilização da plataforma.

Os principais tópicos abordados na base de conhecimento são: Gerenciamento, Framework, Ferramentas, Funções, Recursos e Experimentações, Operadores e Administradores, Projetos e Membros, além de Termos Específicos e Adicionais. Esses tópicos podem ser visualizados na Figura 1, abaixo.

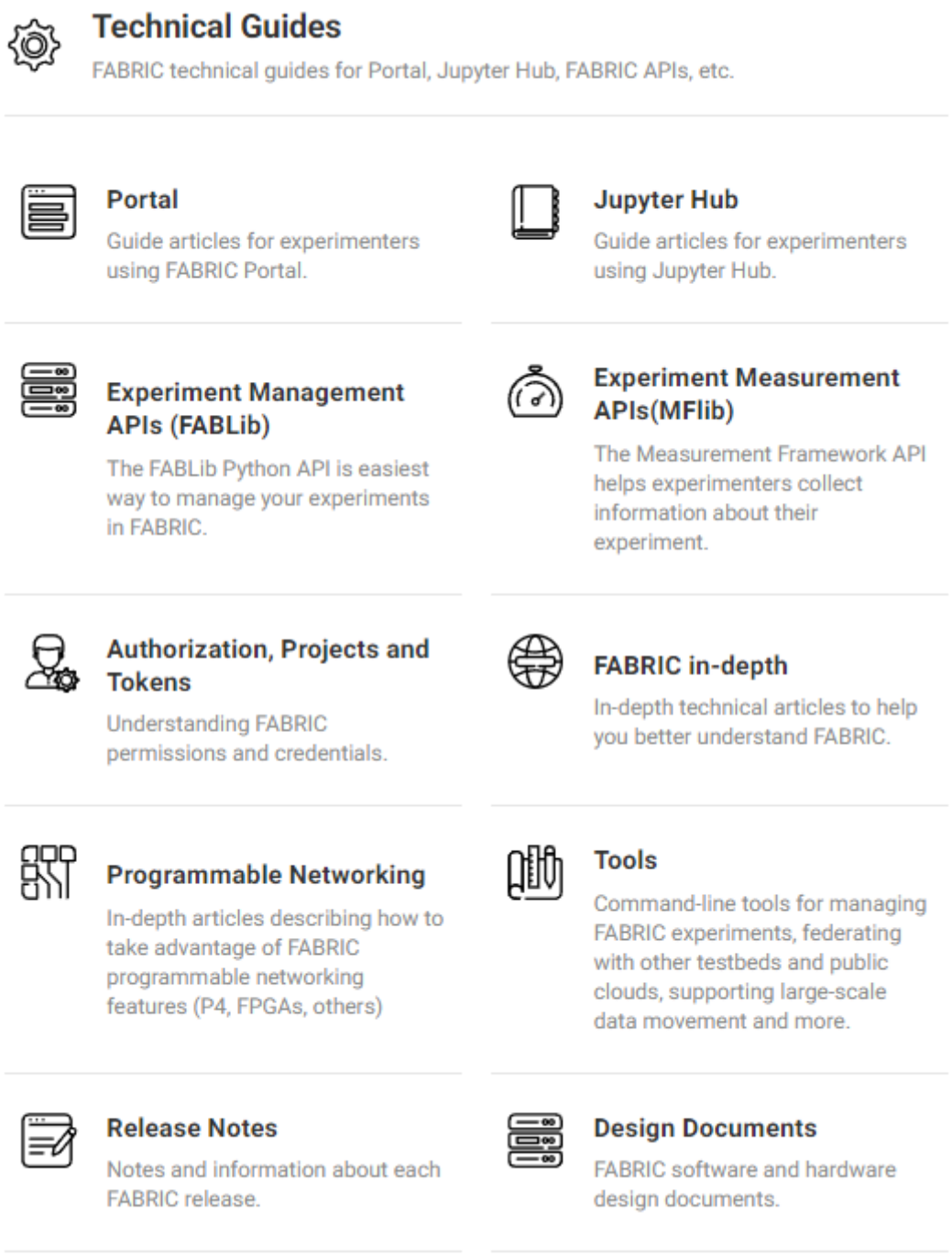


Figura 1. *Base de conhecimento.*

O Fabric não é uma rede isolada para experimentação; ele integra *testbeds* já existentes, como GENI, CloudLab e Chameleon Cloud. Além disso, oferece a possibilidade de se interconectar com diversos recursos experimentais e de produção por meio de colaborações com provedores externos de redes e recursos. Dessa forma, o Fabric pode se conectar a outras redes fora de sua própria infraestrutura, integrando instalações científicas e dados do mundo real. Isso facilita o acesso dos pesquisadores a esses dados e instalações, ampliando a diversidade de recursos, pesquisas e experimentos possíveis. Essas ações dependem de acordos com redes externas e operadores de instalações. A topologia principal do Fabric, mostrada na Figura 2, abrange uma rede Terabit e vários links de 100 Gbps que se estendem por diversas cidades dos Estados Unidos incluindo o HAWI, além de englobar a universidade de Tokio, Universidade de Amsterdam, Bristol e o CERN. Esses sites incorporam nós Fabric interconectados por links de fibra, bem como espaços de colocação provenientes de diversas organizações (GOMEZ et al., 2023).

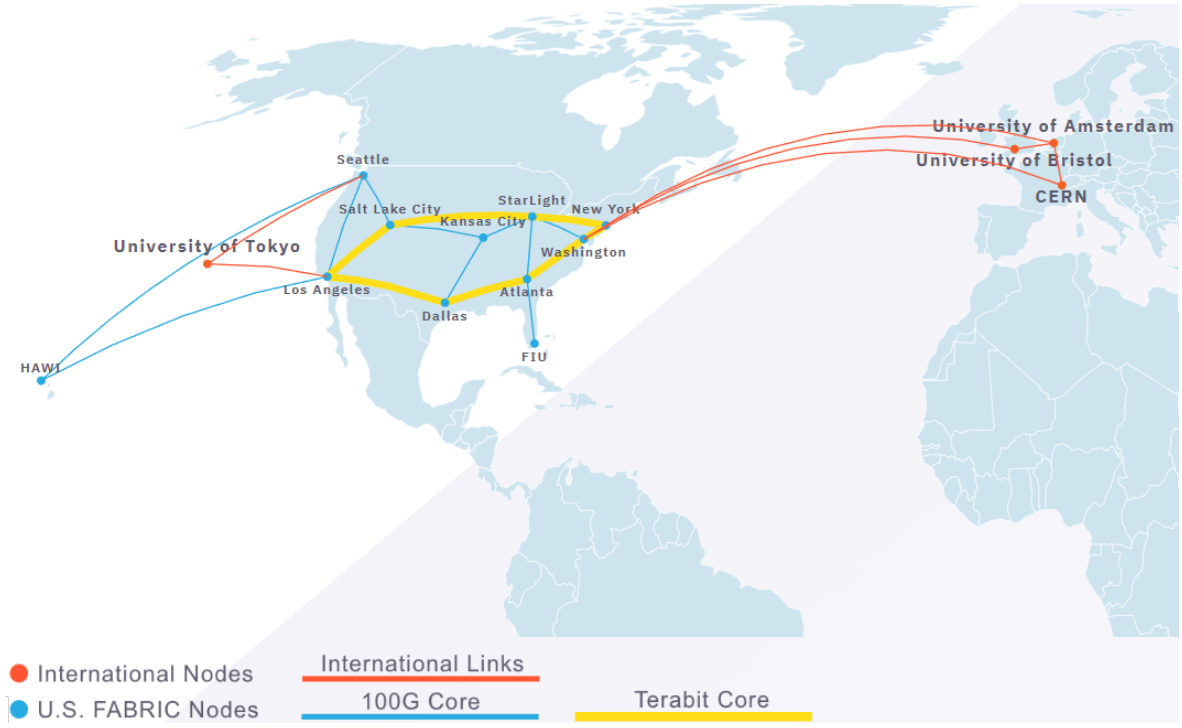


Figura 2. *Topologia FABRIC.*

O Fabric é composto por elementos de rede extensíveis, projetados para serem totalmente programáveis. Ele utiliza links ópticos dedicados de alta velocidade, garantindo uma comunicação rápida e eficiente, ideal para pesquisas em larga escala e experimentos que demandam alta largura de banda e baixa latência. Atualmente existem 40 Nós Fabric distribuídos em pontos de presença da rede ESnet6 (Energy Sciences Network), redes regionais, grandes instalações de ciber infraestrutura e universidades. Um Nó Fabric refere-se a todos os recursos Fabric localizados em um único local de hospedagem. Cada nó é equipado com recursos de armazenamento e recursos computacionais permitindo que os dados sejam processados e armazenados dentro da rede (FABRIC. . . , 2021). Um nó pode ser considerado como nó central, que são os principais componentes da infraestrutura, sendo estrategicamente localizados em diferentes regiões dos Estados Unidos no intuito de garantir cobertura nacional ou nó de borda que são implementados em universidades, laboratórios e em outras instalações, possuindo também recursos computacionais e armazenamento próprios e são menores em níveis de *hardware* em relação aos nós centrais. Esses nós são interconectados por links ópticos dedicados de alta velocidade. Em nível de *hardware* o Fabric oferece máquinas virtuais (VMs) de até 64 núcleos e 384 G de RAM, com dispositivos PCI conectados diretamente a GPUs, Placas de redes, unidades de NVMe (Non-Volatile Memory Express) que é uma interface de comunicação e um protocolo projetado especificamente para aproveitar a alta velocidade de dispositivos de armazenamento baseados em memória não volátil, como SSDs (Solid-State Drives) e FPGA, essas máquinas pode ser colocada em locais de sua escolha e ter grandes volumes de armazenamento e são interconectada por conjuntos de serviços de redes sob demanda, também possui suporte a *switches* P4, placas de redes de alta velocidades, links ópticos, dispositivos de firewall, IDS/IPS e de criptografia (FREQUENTLY. . . , 2024). Vale ressaltar que VMs são "porções" de *hardware* que um determinado nó oferece organizadas em fatias que representam topologias de experimentos individuais. As fatias podem ser criadas, modificadas e excluídas. As fatias têm uma vida útil finita, mas normalmente podem ser estendidas antes de expirarem. Uma fatia pode ser conectar a outros *testbeds* ou instalações usando Facility Ports. Toda a infraestrutura continua em desenvolvimento e é atualizada sempre que um recurso fica disponível, geralmente sem atrapalhar os experimentos já em curso e essas atualizações acontecem a cada 3 a 4 meses.

O Fabric também possui uma infraestrutura de gerenciamento interno, conforme mostrado na Figura 3. Esta infraestrutura é baseada na arquitetura ORCA e utiliza tecnologias modernas para comunicação e segurança, garantindo uma operação segura e eficiente para usuários e experimentadores. A estrutura consiste em três tipos básicos de componentes Aggregate Manager (AM), Broker e Orchestrator que auxiliam continuamente de forma sutil no gerenciamento dos recursos.

- **Aggregate Manager (AM):** Responsável por gerenciar recursos específicos que pertencem a um proprietário, criando delegações de recursos disponíveis e envia para o Broker, responsável também pelo monitoramento do ciclo de vida das aplicações (FABRIC. . . , 2022).

- **Broker:** Coleta delegações de múltiplos AMs e utiliza essas informações para gerar reservas de recursos para os *Orchestrator*, conforme suas solicitações (FABRIC. . . , 2022).
- **Orchestrator:** Incorpora topologias de usuários nos recursos existentes, convertendo essas topologias em reservas e resgatando essas reservas com os AMs, que provisionam os recursos necessários (FABRIC. . . , 2022).

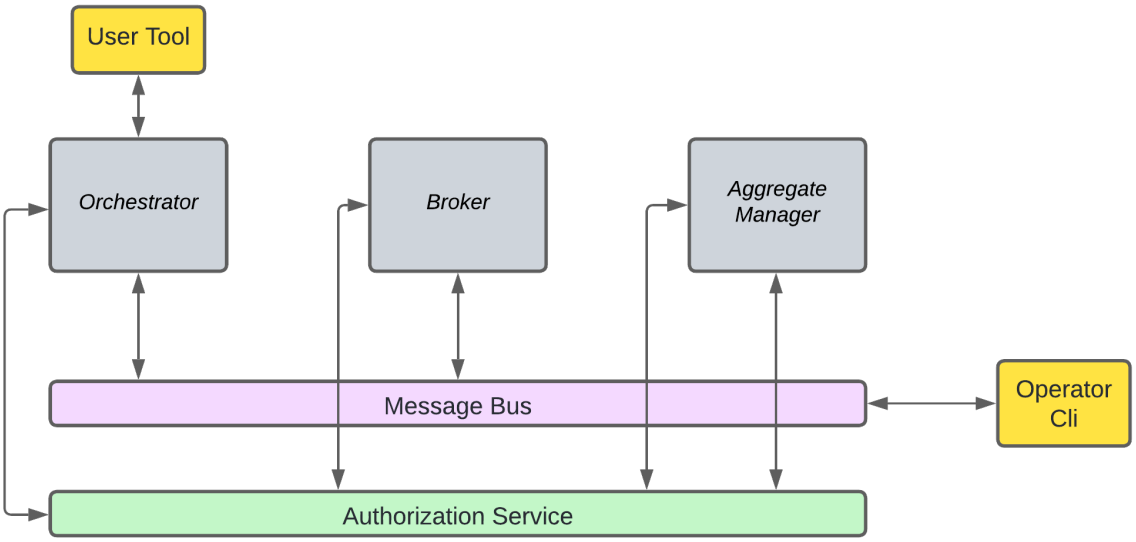


Figura 3. *Estrutura de Gerenciamento interno do Fabric.*

O processo de reserva de recursos envolve várias etapas:

- **Solicitação de Recursos:** O usuário faz uma solicitação para criar uma fatia (slice) de recursos através do Orquestrador. Nesta etapa, a topologia dos recursos necessários é definida, especificando os tipos e quantidades de *hardware* desejados, como CPUs, memória e armazenamento.
- **Mapeamento:** O Orquestrador mapeia a solicitação do usuário nos recursos disponíveis. Neste ponto, o Orquestrador coordena com os Brokers e Aggregate Managers (AMs) para identificar quais recursos estão disponíveis e podem ser alocados para atender à solicitação.
- **Reserva de Recursos:** Com o mapeamento feito, o Broker gera reservas de recursos baseadas nas delegações dos AMs e nas solicitações do Orquestrador. O Orquestrador então aloca essas reservas para a fatia do usuário.
- **Uso de Recursos:** Com os recursos alocados, o usuário utiliza-os para experimentos, com monitoramento e ajustes contínuos para garantir desempenho e eficiência.
- **Liberação de Recursos:**Após a conclusão do uso, o usuário libera os recursos. Os AMs desprovisionam os recursos e atualizam o estado para refletir a disponibilidade.

E para o desenvolvedor no processo de criação dos seus projetos podemos interagir com o Fabric pelo portal, uma aplicação web que roda no navegador, onde se pode gerenciar projetos, chaves SSH, *tokens* de API de forma gráfica e visual. Além disso, o mapa de recursos e suas atualizações são mostrados na sua página inicial como pode ser visto na Figura 4.

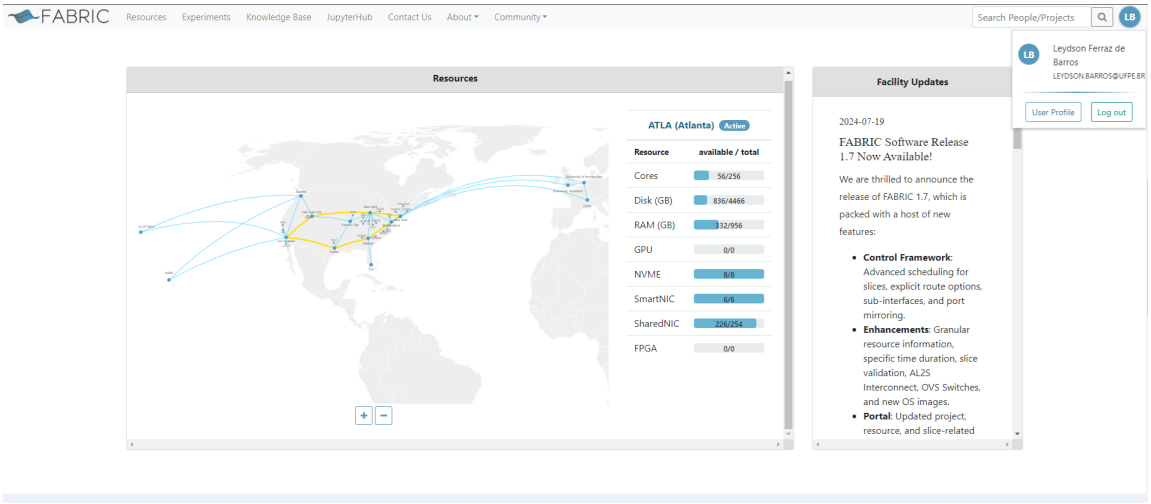


Figura 4. *Mapas de recursos.*

O portal também dá acesso ao JupyterHub, uma espécie de Jupyter Notebooks que utiliza APIs do Fabric para executar experimentos e topologias diretamente no navegador. É recomendado iniciar os experimentos com o JupyterHub, e para o desenvolvimento prático deste trabalho, optou-se por usá-lo. O JupyterHub é um contêiner privado que inclui um sistema de arquivos onde você pode armazenar seus cadernos de experimentos Fabric. Ele inclui um conjunto de notebooks de exemplo que demonstram o uso da API Fabric *Python* como padrão.

Outra forma de interagir com o Fabric é através das APIs do Fabric, permitindo que o usuário utilize seu próprio *laptop* ou *desktop* para realizar as ações mencionadas. As bibliotecas da API Fabric estão disponíveis para download no PyPi.

E para interagir com o Portal, JupyterHub e API o Fabric tem 4 tipos de usuários diferentes:

- **Project Lead:** é um usuário especialmente privilegiado do FABRIC com permissão para criar novos projetos, designar Proprietário(s) do Projeto e convidar membros regulares para um projeto. Como essa função vem com muitas responsabilidades, apenas professores com estabilidade ou equipe sênior de TI podem solicitar essa função. Isso pode ser feito na página Perfil do Usuário no portal (FREQUENTLY..., 2024).

- **Facility Operator:** O *Facility Operator* é um superusuário autorizado a manipular qualquer projeto. Esta função é reservada para a equipe FABRIC (FREQUENTLY..., 2024).

- **Project Owner:** Um Proprietário de Projeto pode ser designado a um projeto específico por um Líder de Projeto ou outro Proprietário de Projeto. Sua responsabilidade é gerenciar Membros do Projeto – adicioná-los ou removê-los (FREQUENTLY..., 2024).

- **Project Member:** Este é um membro regular do projeto sem privilégios especiais para gerenciar o projeto. Eles podem criar fatias dentro do projeto (FREQUENTLY…, 2024).

Em conclusão, podemos considerar o Fabric como um instrumento científico e um laboratório distribuído, projetado para apoiar pesquisas em infraestrutura cibernética, incluindo arquiteturas de Internet, segurança, protocolos distribuídos e aplicações.

3 Explorando o Fabric

Neste capítulo, inicia-se a exploração prática da plataforma Fabric, que é o principal objeto de estudo desta monografia. Com uma abordagem passo a passo, serão mostradas as principais etapas envolvidas na criação e configuração de um ambiente de experimentação dentro do Fabric.

O capítulo está organizado de maneira a guiar o leitor pelos principais processos necessários para o desenvolvimento de um projeto na plataforma, destacando desde a criação de fatias (slices) até a configuração de componentes e a execução. O objetivo é demonstrar na prática a flexibilidade e o potencial oferecido pelo Fabric.

Para a prática deste trabalho, optou-se por usar o JupyterHub, por ser mais intuitivo para descrever a programação e implementar a arquitetura que desejo exemplificar. Foi usado o FABlib, uma biblioteca Python criada para interagir com o ambiente Fabric, onde sua documentação pode ser acessada no https://fabric-fablib.readthedocs.io/en/latest/.

Ao estudar a documentação do FABlib, aprendemos a construir e manipular slices, nós, componentes, interfaces, *switches*, serviços de rede, facility ports e outros recursos oferecidos pelo Fabric. Ressalto que a documentação do Fablib é de fácil leitura e entendimento. Ao longo das seções, será possível entender o funcionamento de elementos como nós, switches programáveis e interfaces de rede, além da importância do uso de bibliotecas FABlib, para a manipulação e controle dos recursos disponíveis.

3.1 Passo 1: Criação do Ambiente

Antes de iniciar o desenvolvimento, foi necessário criar o ambiente de desenvolvimento no JupyterHub. A seguir, descreverei todas as etapas do processo de criação do ambiente, começando pela geração das chaves SSH. Elas são essenciais para conceder acesso às suas fatias. Para isso, acessando o portal do Fabric, vá em "*User Profile* > *My SSH KEYS*"e crie um par de chaves *slice/sliver*. Nesse momento, uma chave pública e outra privada serão geradas. O Fabric mantém apenas as chaves SSH públicas, enquanto o armazenamento das chaves SSH privadas é de responsabilidade do experimentador.

Com as chaves criadas, podemos configurar o ambiente de desenvolvimento. Acesse o JupyterHub, abra a pasta *"fabric_config"*, importe as chaves SSH criadas e edite o arquivo *"fabric_config/fabric_rc"*.

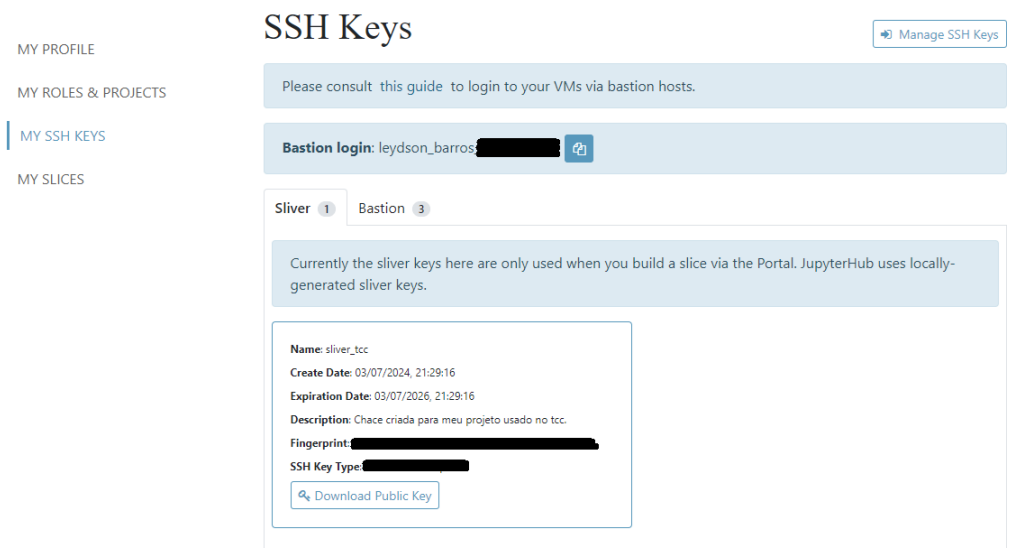


Figura 5. *SSH Keys.*

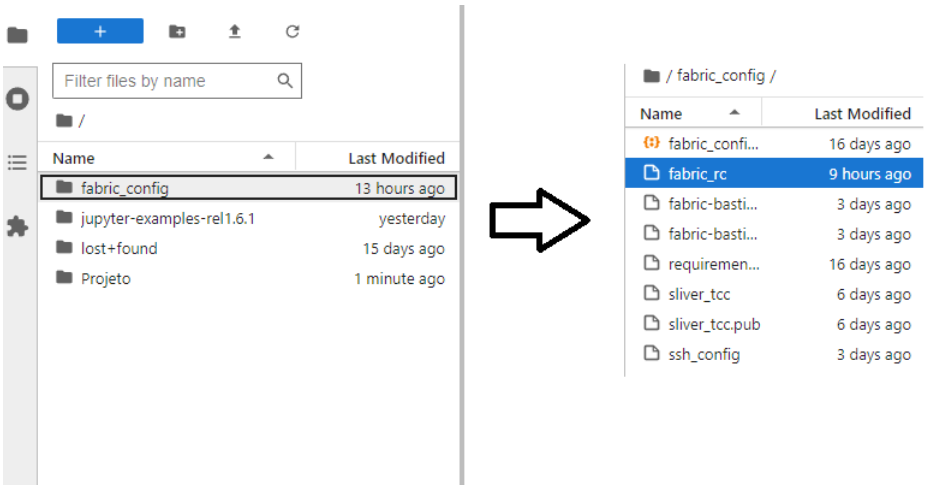


Figura 6. *Diretórios jupyterhub.*

Ao abrir o arquivo, veremos as seguintes linhas de comando:

```
export FABRIC_ORCHESTRATOR_HOST=orchestrator.fabric-testbed.net
export FABRIC_CREDMGR_HOST=cm.fabric-testbed.net
export FABRIC_CORE_API_HOST=uis.fabric-testbed.net
export FABRIC_TOKEN_LOCATION=/home/fabric/.tokens.json
export FABRIC_PROJECT_ID=74f461a8-035d-40df-8c4a-f3c4be6a5714
export FABRIC_BASTION_HOST=bastion.fabric-testbed.net
export FABRIC_BASTION_USERNAME=<Nome de Usuário>
export FABRIC_BASTION_KEY_LOCATION=/home/fabric/work/fabric_config/fabric-bastion-key
export FABRIC_SLICE_PUBLIC_KEY_FILE=/home/fabric/work/fabric_config/<Chave Publica>
export FABRIC_SLICE_PRIVATE_KEY_FILE=/home/fabric/work/fabric_config/<Chave privada>
export FABRIC_AVOID=
export FABRIC_SSH_COMMAND_LINE=ssh -i {{ _self_.private_ssh_key_file }} -F /home/fabri
c/work/fabric_config/ssh_config {{ _self_.username }}@{{ _self_.management_ip }}
export FABRIC_LOG_LEVEL=INFO
export FABRIC_LOG_FILE=/tmp/fablib/fablib.log
export FABRIC_BASTION_SSH_CONFIG_FILE=/home/fabric/work/fabric_config/ssh_config
```

Figura 7. *Arquivo de configuração de acesso.*

Será necessário indicar o nome do usuário. Para encontrar o nome do seu usuário, acesse o portal e vá em "*User Profile* > *MY PROFILE*". Na linha export *FABRIC_BASTION_USERNAME=<Nome de Usuário>*, faça o apontamento para o seu nome de usuário.

Além disso, especifique as chaves criadas anteriormente nas linhas correspondentes: export FABRIC_SLICE_PUBLIC_KEY_FILE=/home/fabric/work/fabric_config/<Chave Publica> e export FABRIC_SLICE_PRIVATE_KEY_FILE=/home/fabric/work/fabric_config/<Chave privada>

Depois desse procedimento salve o arquivo e a próxima etapa será configurar o ID do projeto no ambiente.

Configuração do ID do projeto no ambiente

Depois de criar as chaves e configurá-las no arquivo *fabric_config/fabric_rc*, podemos apontar o ID do projeto no nosso ambiente de desenvolvimento. O ID do projeto pode ser encontrado na página de informações básicas do projeto desejado. Para isso, acesse o portal e vá até a aba *"Experiments"*, onde você terá acesso aos projetos dos quais faz parte. Ao abrir a página do projeto, você encontrará as informações básicas, incluindo o ID do projeto.

Com o ID em mãos, basta rodar o código abaixo apontando o ID para configurar o ambiente para o projeto em questão.

```
1  from fabrictestbed_extensions.fablib.fablib import FablibManager as fablib_manager
2
3  # Id do projeto que se deseja desenvolver
4  project_id = "74f461a8-035d-40df-8c4a-f3c4be6a5714"
5
6  # Criando um obejto Fabric.
7  fablib = fablib_manager(project_id=project_id)
```

Figura 8. *Apontamento do ID e criação do Projeto.*

Ao rodar o código acima, estaremos configurando o ambiente para o projeto no qual desejamos construir nossa arquitetura de teste. Para garantir que tudo ocorreu corretamente, podemos verificar as configurações do projeto executando o comando apresentado na Figura 9. É uma boa prática sempre verificar as configurações do projeto, especialmente se o desenvolvedor tiver acesso a vários projetos, para evitar quaisquer equívocos.

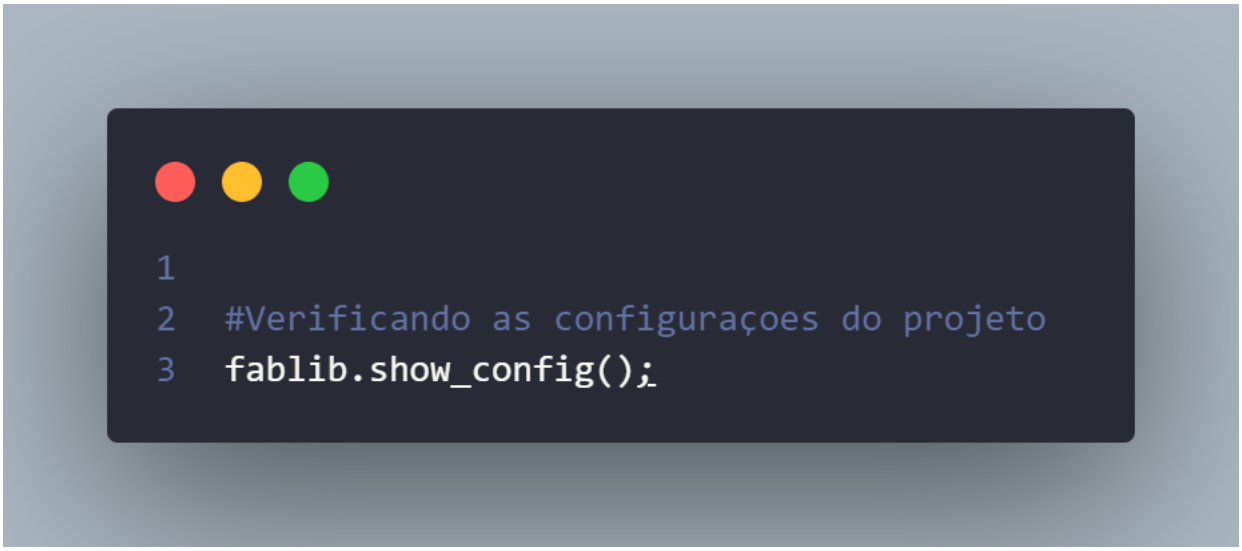


Figura 9. *Verificação das configuração do Projeto.*

Ao rodar o código acima teremos a Figura 10 como retorno. Ela contém informações pertinentes para o qual fizemos o apontamento.

Orchestrator	orchestrator.fabric-testbed.net
Credential Manager	cm.fabric-testbed.net
Core API	uis.fabric-testbed.net
Token File	/home/fabric/.tokens.json
Project ID	74f461a8-035d-40df-8c4a-f3c4be6a5714
Bastion Host	bastion.fabric-testbed.net
Bastion Username	leydson_barros_0000193045
Bastion Private Key File	/home/fabric/work/fabric_config/fabric-bastion-key
Slice Public Key File	/home/fabric/work/fabric_config/sliver_tcc.pub
Slice Private Key File	/home/fabric/work/fabric_config/sliver_tcc
Sites to avoid	
SSH Command Line	ssh -i {{ _self_.private_ssh_key_file }} -F /home/fabric/work/fabric_config/ssh_config {{ _self_.username }}@{{ _self_.management_ip }}
Log Level	INFO
Log File	/tmp/fablib/fablib.log
Bastion SSH Config File	/home/fabric/work/fabric_config/ssh_config
Version	1.6.2
Data directory	/tmp/fablib

Figura 10. *Tabela de configurações.*

Depois de verificar todas as informações o próximo passo é validar as informações acima, para isso executamos o código da Figura 11 que verifica se toda a configuração necessária existe para que a fatia funcione com sucesso.

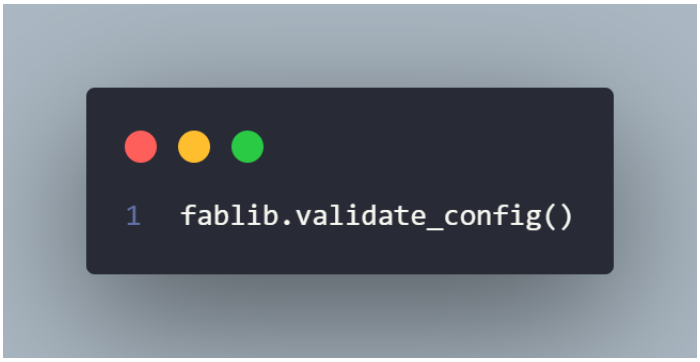


Figura 11. *Validando o ambiente de desenvolvimento.*

Se tudo estiver certo teremos a Figura 12 como resposta.

```
User: LEYDSON.BARROS@UFPE.BR bastion key is valid!
Bastion SSH Config file already exists, not making updates!
Configuration is valid and please save the config!
```

Figura 12. *Resposta da validação.*

Se tudo correu bem nas execuções acima podemos salvar as configurações feitas, executando o código na Figura 13.

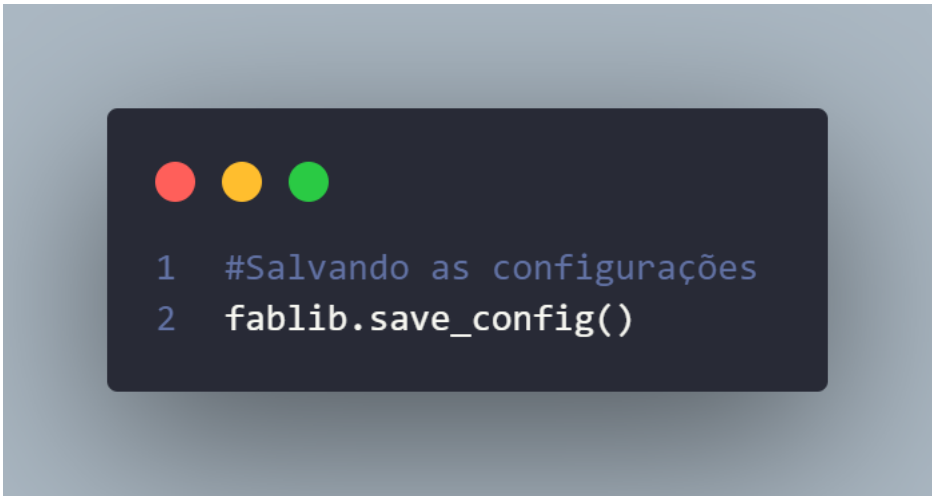


Figura 13. *Salvando a configuração do ambiente.*

3.2 Passo 2: Criando a Fatia (Slice)

Realizando todas as etapas acima teremos o nosso ambiente configurado, podemos começar a desenvolver nosso cenário de testes no Fabric que usará 2 servidores e um switch programável P4. Utilizaremos um switch P4 BMv2, que é muito utilizado para definir o comportamento dos pacotes em dispositivos de rede programáveis. Tomei como referência o laboratório 'lab1_creating_a_slice_with_a_P4_switch'.

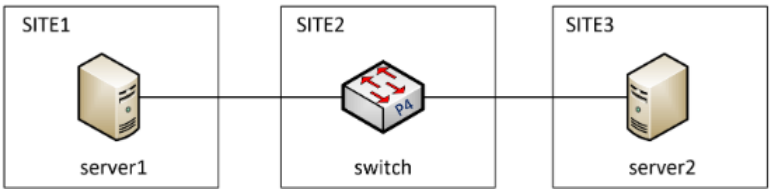


Figura 14. *Infraestrutura proposta.*

Demonstrarei a implementação de forma didática e prática para exemplificar a criação de um projeto no Fabric. Modelarei um passo a passo que servirá como uma "receita"para ajudar a guiar qualquer projeto que se deseje criar na plataforma Fabric.

Para começar a desenvolver nosso projeto de teste, precisaremos criar uma fatia/slice. Uma fatia é uma coleção de recursos logicamente relacionados que representam uma única execução de um experimento (ou uma parte de um experimento, pois várias fatias podem estar envolvidas). Normalmente, uma fatia representa uma topologia conectada de recursos conhecidos como *slivers*. Cada fatia faz parte de um e somente um projeto.

Para criar uma fatia, execute o código da Figura 15:

```
1  # O código abaixo cria uma nova fatia/slice
   com o nome "Tcc_Leydson"
2  slice = fablib.new_slice(name="Tcc_Leydson")
```

Figura 15. *Criação de Fatia (Slice).*

3.3 Passo 3: Busca por recursos

Antes de começar a implementação, é importante verificar quais sites possuem os recursos necessários para o seu projeto. Isso garantirá sucesso na solicitação e alocação do *hardware*.

No projeto, utilizarei 4 CPU cores, 8GB RAM, 20GB para cada servidor, além do switch de 16 CPU, 8GB RAM, 40GB.

Para utilizar dados programáveis em P4, configurarei um switch de *software*, ou switch virtual, que funciona como um switch de *hardware* tradicional. Este trabalho tem como foco principal a demonstração da ferramenta Fabric, então não entrarei em detalhes sobre a linguagem P4

Uso o código abaixo para listar os sites que podem suportar a quantidade de *hardware* que desejo alocar. Como vou utilizar processamento, memória RAM, armazenamento e placas de rede, uso os seguintes nomes para compor o filtro nas buscas: *cores available*, *ram available*, *disk available*, e nic basic available. Esses nomes podem ser ajustados conforme as necessidades da sua implementação.

```
1  #filtro de componentes
2  fields=['name','cores_available','ram_available','disk_availa
   ble','nic_basic_available']
3  #Buscas por componetes disponíveis nos sites
4  output_table = fablib.list_sites(fields=fields, latlon=False)
```

Figura 16. *Filtro de componentes.*

Ao executar o código receberemos como retorno uma tabela como podemos ver na Figura 17, que mostra os componentes disponíveis solicitados em cada site do Fabric.

Sites				
Name	**Cores Available**	**RAM Available**	**Disk Available**	**Basic NIC Available**
CLEM	238	962	52079	312
RUTG	452	1692	102683	543
AMST	260	950	60198	241
MAX	324	242	104423	564
BRIST	276	1266	54878	363
TOKY	2	514	60258	309
STAR	452	2196	103216	691
UTAH	292	1150	100861	592

Figura 17. *Exemplo de retorno do filtro.*

Para esse exemplo os sites selecionados foram GATECH, RUTG, EDUKY.

3.4 Passo 4: Criando os Componentes/Nós

O código abaixo cria três nós: server1, server2 e o switch. O server1 será criado no site GATECH e o server3 será criado no site EDUKY e o Switch será criado no site RUTG.

```
server1 = slice.add_node(name="server1",
                         site=site1,
                         cores=4,
                         ram=8,
                         disk=20,
                         image='default_ubuntu_20')

server2 = slice.add_node(name="server2",
                         site=site3,
                         cores=4,
                         ram=8,
                         disk=20,
                         image='default_ubuntu_20')

switch = slice.add_node(name="switch",
                         site=site2,
                         cores=32,
                         ram=16,
                         disk=40,
                         image='default_ubuntu_20')
```

Figura 18. *Criação de Servidores e Switch.*

3.5 Passo 5: Configuração dos componentes

O código abaixo faz a conexão entre os componentes, adicionando uma placa de interface de rede (NIC) a cada servidor e 2 placas NIC ao switch, criando uma rede site a site entre o site1 e o site3 conectando o servidor1 e o servidor2 e o switch P4.

```
# Adicionando uma placa de interface de rede (NIC) a cada servidore
server1_iface = server1.add_component(model='NIC_Basic').get_interfaces()[0]
server2_iface = server2.add_component(model='NIC_Basic').get_interfaces()[0]

# Adiciona duas placas de interface de rede (NICs) ao switch.
switch_iface1 = switch.add_component(model='NIC_Basic', name='net1_nic').get_interfaces()[0]
switch_iface2 = switch.add_component(model='NIC_Basic', name='net2_nic').get_interfaces()[0]

#Criando uma rede site a site entre o site1 e o site2 conectando o servidor1 e o switch P4
net1 = slice.add_l2network(name='net1', interfaces=[server1_iface, switch_iface1])

#Crie uma rede site a site entre site2 e site3 conectando o switch P4 e o servidor2
net2 = slice.add_l2network(name='net2', interfaces=[switch_iface2, server2_iface])
```

Figura 19. *Adicionando Placa de Redes e fazendo o links entre os sites pelo switch.*

Após executar o código acima, nossa infraestrutura estará conectada da seguinte forma:

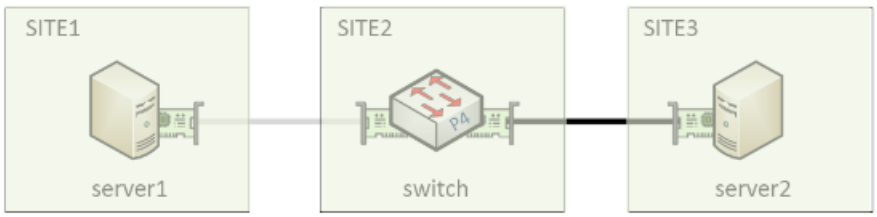


Figura 20. *Infraestrutura Conectada.*

3.6 Passo 6: Solicitando a construção da fatia

Após descrever e conectar a infraestrutura que desejamos criar no Fabric, podemos agora fazer a alocação propriamente dita do *hardware* desejado nos sites, e para isso executaremos o código da Figura 21 abaixo.

```
1  # O código abaixo envia a solicitação da fatia.
2  slice.submit();
```

Figura 21. *Solicitando a alocação da fatia.*

Se tudo ocorrer bem e os sites tiverem o *hardware* disponível, teremos as saídas apontando sucesso para a fatia, os componentes e as interfaces, conforme mostrado nas figura 22, 23 e 24 abaixo:

Slice	
ID	032c9729-3917-4a49-a600-a0918906c526
Name	Tcc_Leydson
Lease Expiration (UTC)	2024-07-12 16:53:59 +0000
Lease Start (UTC)	2024-07-11 16:54:01 +0000
Project ID	74f461a8-035d-40df-8c4a-f3c4be6a5714
State	StableOK

Figura 22. *Status da Alocação da Fatia.*

Nodes

ID	Name	Cores	RAM	Disk	Image	Image Type	Host	Site	Username	Management IP	State	Error
ff2cc72b-b316-4e5c-8564-56a200006e60	server1	4	8	100	default_ubuntu_20	qcow2	gatech-w4.fabric-testbed.net	GATECH	ubuntu	2610:148:1f00:9f01:f816:3eff:fe89:5b59	Active	
b21eea3c-07eb-4cf8-a291-444844388334	server2	4	8	100	default_ubuntu_20	qcow2	eduky-w12.fabric-testbed.net	EDUKY	ubuntu	2610:1e0:1700:206:f816:3eff:fe1c:d94	Active	
5679e197-8062-4801-ad92-d7bd658b875a	switch	32	16	100	default_ubuntu_20	qcow2	rutg-w2.fabric-testbed.net	RUTG	ubuntu	2620:0:d61:4101:f816:3eff:fe24:fb8b	Active	

Figura 23. *Status da Alocação dos Nós.*

Networks

ID	Name	Layer	Type	Site	Subnet	Gateway	State	Error
5a90ee59-867f-4a15-90a8-dbc74231c054	net1	L2	L2STS	None	None	None	Active	
664fadea-0042-4cac-a3ed-7933459e6244	net2	L2	L2STS	None	None	None	Active	

Interfaces

Name	Short Name	Node	Network	Bandwidth	Mode	VLAN	MAC	Physical Device	Device	IP Address	Numa Node
server2-None-p1	p1	server2	net2	100	config		02:3E:FE:EC:F4:77	enp7s0	enp7s0	None	1
switch-net1_nic-p1	p1	switch	net1	100	config		46:3C:D4:E8:5E:54	enp7s0	enp7s0	None	6
switch-net2_nic-p1	p1	switch	net2	100	config		4A:64:8A:28:E7:3A	enp8s0	enp8s0	None	6
server1-None-p1	p1	server1	net1	100	config		02:90:C6:F3:FC:7A	enp7s0	enp7s0	None	4

Figura 24. *Status das alocação da Network e Interface*

3.7 Passo 7: Configurando a InfraEstrutura criada

Quero destacar que, desde a etapa 1 até a etapa 6, mapeei um passo a passo aplicável a todos os projetos que desejamos criar no Fabric. Esse processo inclui configurar o ambiente, verificar a disponibilidade de *hardware* nos sites, especificar o *hardware* desejado, definir as configurações necessárias e realizar a alocação.

A partir da etapa 7, os conhecimentos sobre como operar o Fabric se combinam com as habilidades de gerenciamento e operação de *hardware* e serviços. Isso inclui configurar

servidores e seus serviços, ajustar *switches*, entre outras tarefas. É essencial saber operar e configurar os *hardwares* solicitados para aproveitar ao máximo os recursos oferecidos.

O código da Figura 25 realiza a instalação do BMv2 no switch criado, além de configurar as ferramentas de rede necessárias nos servidores e no *switch*.

```
# Instalando o BMv2 no switch criado anteriormente.
# Para fazer a instalaçao usarei um script install_bmv2.sh
switch = slice.get_node(name="switch")
switch.upload_file('install_bmv2.sh', 'install_bmv2.sh')
stdout, stderr = switch.execute(f'chmod +x install_bmv2.sh &&  ./install_bmv2.sh',quiet=True)

# Script que foi usado para instalar o BMv2 no Switch criado
#!/bin/bash

. /etc/os-release && echo "deb https://download.opensuse.org/repositories/home:/p4lang/xUbuntu_${VERSION_ID}/ /" | sudo tee /etc/apt/sources.list.d/home:p4lang.list && curl -L "https://download.opensuse.org/repositories/home:/p4lang/xUbuntu_${VERSION_ID}/Release.key" | sudo apt-key add - && sudo apt-get update && sudo apt install -y p4lang-p4c

# Instalando net-tools
# O pacote net-tools será instalado nos nós switch, server1 e server2.
# Este pacote nos permitirá usar os comandos ifconfig e arp

server1 = slice.get_node(name="server1")
server2 = slice.get_node(name="server2")
stdout, stderr = server1.execute(f'sudo apt-get install -y net-tools', quiet=True)
stdout, stderr = server2.execute(f'sudo apt-get install -y net-tools', quiet=True)
stdout, stderr = switch.execute(f'sudo apt-get install -y net-tools', quiet=True)

```

Figura 25. *Instalação do Switch P4 e os serviços de redes.*

Atribuição de endereços IP e MAC

O código da Figura 26 é responsável por nomear as interfaces para que possamos atribuir endereços IP a elas.

```
node1_iface = server1.get_interface(network_name='net1')
server1_iface_name = node1_iface.get_device_name()
print(f'server1_iface: {server1_iface_name}')

node2_iface = server2.get_interface(network_name='net2')
server2_iface_name = node2_iface.get_device_name()
print(f'server2_iface: {server2_iface_name}')

switch_iface1 = switch.get_interface(network_name='net1')
switch_iface1_name = switch_iface1.get_device_name()
print(f'switch_iface1: {switch_iface1_name}')

switch_iface2 = switch.get_interface(network_name='net2')
switch_iface2_name = switch_iface2.get_device_name()
print(f'switch_iface2: {switch_iface2_name}')
```

Figura 26. *Nomeando as interfaces.*

Quando executar o código da Figura 26 recebermos as seguintes respostas:

server1_iface: enp7s0
server2_iface: enp7s0
switch_iface1: enp7s0
switch_iface2: enp8s0

O próximo passo é ativar todas as interfaces. Para isso, vamos usar o comando ip link, que é utilizado para configurar interfaces de baixo nível ou para ajustes de protocolos, como VLANs, ARP ou MTUs, além de poder habilitar ou desabilitar uma interface.

```
#Ativando todas as interfaces
#Ativando as interface com o comando ip link
stdout, stderr = server1.execute(f'sudo ip link set dev {server1_iface_name} up', quiet=True)
stdout, stderr = server2.execute(f'sudo ip link set dev {server2_iface_name} up', quiet=True)
stdout, stderr = switch.execute(f'sudo ip link set dev {switch_iface1_name} up', quiet=True)
stdout, stderr = switch.execute(f'sudo ip link set dev {switch_iface2_name} up', quiet=True)

### Codificando endereço MAC
server1_iface_MAC = '00:00:00:00:00:01'
switch_iface1_MAC = '00:00:00:00:00:02'
switch_iface2_MAC = '00:00:00:00:00:03'
server2_iface_MAC = '00:00:00:00:00:04'
```

Figura 27. *Ativação das interfaces.*

Agora, vamos configurar os endereços IP e MAC nos servidores e no switch usando o código da Figura 28.

```python
#Configurando os endereços IP e MAC em server1_iface e switch_iface1
server1_switch_subnet = "192.168.1.0/24"
server1_ip = '192.168.1.10/24'
switch_ip1 = '192.168.1.1/24'

stdout, stderr = server1.execute(f'sudo ifconfig {server1_iface_name} {server1_ip}')
stdout, stderr = switch.execute(f'sudo ifconfig {switch_iface1_name} {switch_ip1}')

stdout, stderr = server1.execute(f'sudo ifconfig {server1_iface_name} hw ether {server1_iface_MAC}')
stdout, stderr = switch.execute(f'sudo ifconfig {switch_iface1_name} hw ether {switch_iface1_MAC}')

#### Configuring the IP and MAC addresses on switch_iface2 and server2_iface
server2_switch_subnet = "192.168.2.0/24"
server2_ip = '192.168.2.10/24'
switch_ip2 = '192.168.2.1/24'

stdout, stderr = server2.execute(f'sudo ifconfig {server2_iface_name} {server2_ip}')
stdout, stderr = switch.execute(f'sudo ifconfig {switch_iface2_name} {switch_ip2}')

stdout, stderr = server2.execute(f'sudo ifconfig {server2_iface_name} hw ether {server2_iface_MAC}')
stdout, stderr = switch.execute(f'sudo ifconfig {switch_iface2_name} hw ether {switch_iface2_MAC}')
```

Figura 28. *Configuração das interfaces nos Servidores e Switch.*

Ao executar todo o código da Figura 28 teremos deixando nossa infraestrura configurada como a Figura 29:

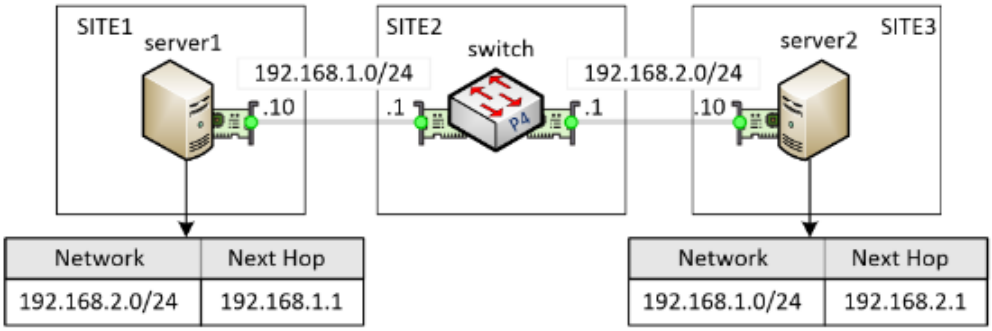


Figura 29. *Infraestrutura ativada.*

Vamos usar no switch com o comando sudo **sysctl -w net.ipv4.ip_forward=1**, visando ativar o encaminhamento de pacotes IPv4 no kernel do Linux. Dessa forma, permitimos que o sistema atue como um roteador, encaminhando pacotes de rede de uma interface para outra. O comando será executado no dispositivo switch, conforme a Figura 30.

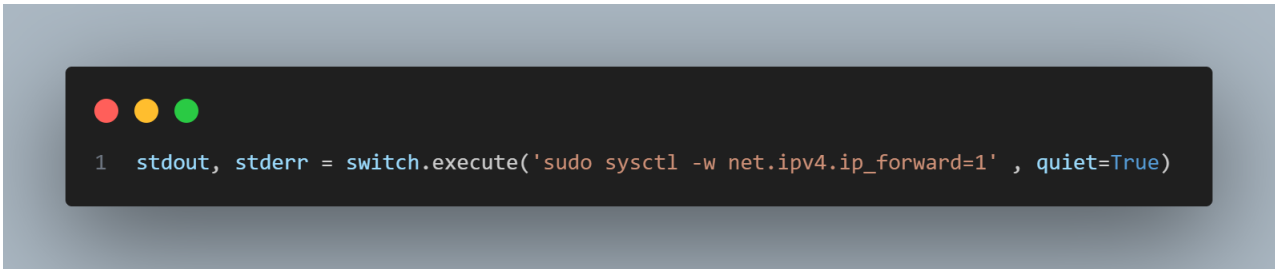


Figura 30. *Habilitando o encaminhamento Switch.*

Vamos excluir as entradas de roteamento para forçar o tráfego a passar pelo switch BMv2. Ao deletar as rotas, os pacotes serão direcionados pelo switch BMv2, em vez de serem encaminhados pelo *kernel*.

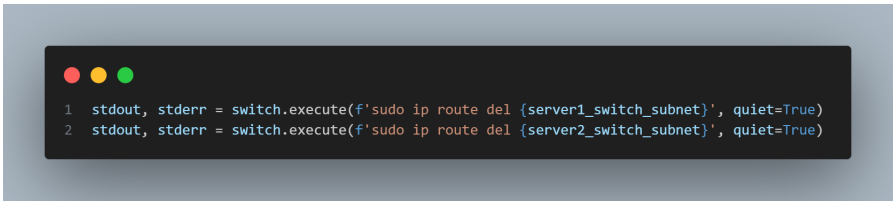


Figura 31. *Excluindo entradas de roteamento.*

Nesta etapa, configuraremos rotas estáticas em server1 e server2. Para server1, adicionaremos uma rota para chegar à rede **192.168.2.0/24** via **192.168.1.1**. Para server2, adicionaremos uma rota para chegar à rede **192.168.1.0/24** via **192.168.2.1**.

Figura 32. *Configuração de Roteamento.*

Nesta etapa configuraremos entradas ARP estáticas em server1 e server2. A razão pela qual estamos fazendo isso é porque o switch não processa pacotes ARP a menos que seja programado para isso. Para garantir que os pacotes ARP não sejam enviados para o switch, codificaremos os MACs nos servidores.

Para cada servidor, adicionaremos uma entrada ARP à interface vizinha do *switch*.

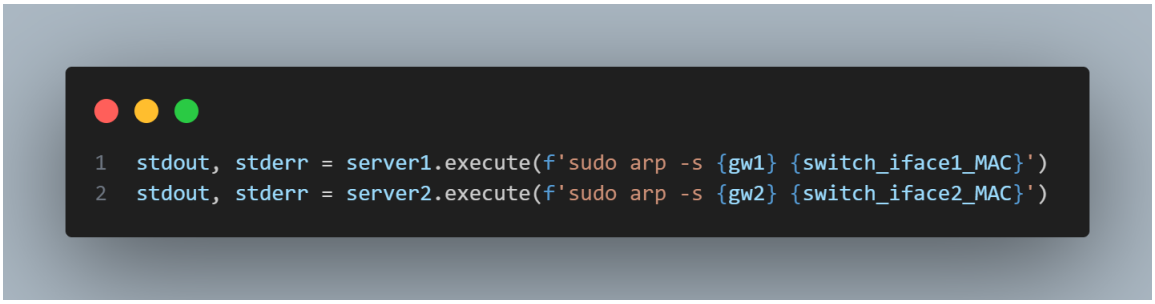


Figura 33. *Configuração ARP.*

Compilando e iniciando um programa de roteamento P4 simples

Nesta etapa, compilaremos o programa P4 e iniciaremos o *switch* usando o compilador p4c. Foi usado um programa P4 exemplo que tem a finalidade de configurar um *switch* para analisar, processar e encaminhar pacotes IPv4, além de gerenciar os endereços MAC e TTL dos pacotes encaminhados. Esse programa foi desenvolvido em uma IDE externa e depois do script pronto fazemos o upload do arquivo e executamos o comando da Figura 34 para refletir no nosso switch.

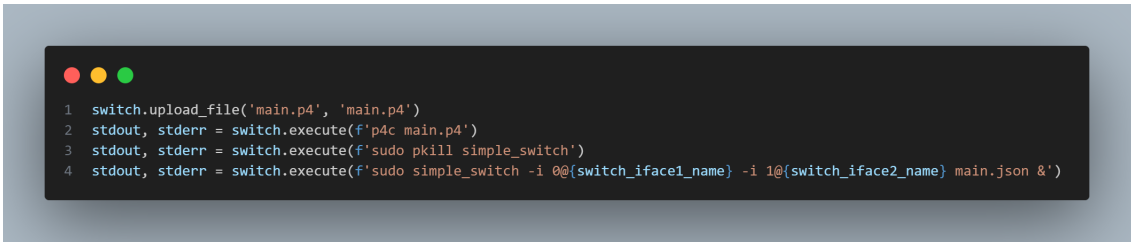


Figura 34. *Iniciando o Switch P4.*

Se tudo ocorreu bem teremos a resposta abaixo:

Calling target program-options parser
Adding interface enp7s0 as port 0
Adding interface enp8s0 as port 1

E teremos o *switch* configurado da segunte maneira:

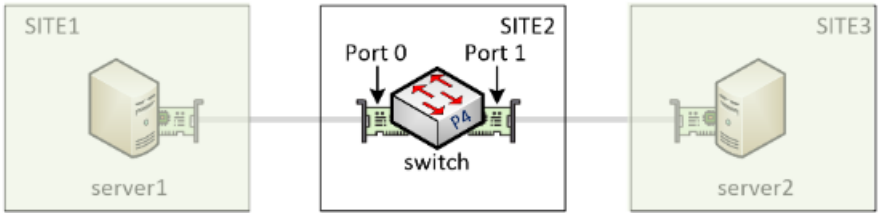


Figura 35. *Switch P4 configurado.*

Agora, vamos configurar a tabela de encaminhamento. Seguiremos os mesmos passos: criaremos o script, faremos o upload para o *switch* e, em seguida, o executaremos conforme o código da Figura 36.

```
1  #Tabela de Encaminhamento
2  #Para preencher a tabela irei rodar um script.
3  switch.upload_file('rules.sh', 'rules.sh')
4  stdout, stderr = switch.execute('chmod +x rules.sh && ./rules.sh')
```

Figura 36. *Preenchendo a tabela de encaminhamento.*

Se tudo ocrreu bem, teremos a seguite saída do comando acima.

"Obtaining JSON from switch...

Done

Control utility for runtime P4 table manipulation

RuntimeCmd: Adding entry to lpm match table MyIngress.ipv4_lpm

match key: LPM-c0:a8:01:00/24

action: MyIngress.forward

runtime data: 00:00:00:00:00:01 00:00

Entry has been added with handle 0

RuntimeCmd:

Obtaining JSON from switch...

Done

Control utility for runtime P4 table manipulation

RuntimeCmd: Adding entry to lpm match table MyIngress.ipv4_lpm

match key: LPM-c0:a8:02:00/24

action: MyIngress.forward

runtime data: 00:00:00:00:00:04 00:01

Entry has been added with handle 1

RuntimeCmd:"

Para testar a conexão fazemos um teste ping. O código da Figura 37 realiza um teste de ping do servidor 1 para o servidor 2, para verificar se ambos estão se enxergando na rede.

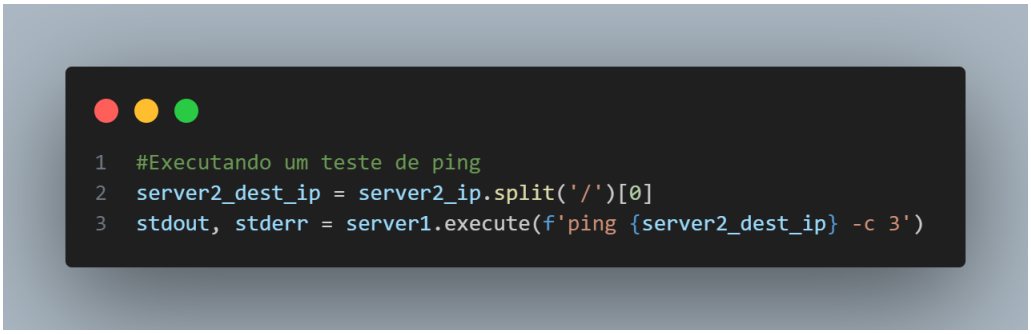


Figura 37. *Teste Ping.*

Saída do teste ping feito:

PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.

64 bytes from 192.168.2.10: icmp_seq=1 ttl=63 time=47.4 ms

64 bytes from 192.168.2.10: icmp_seq=2 ttl=63 time=47.0 ms

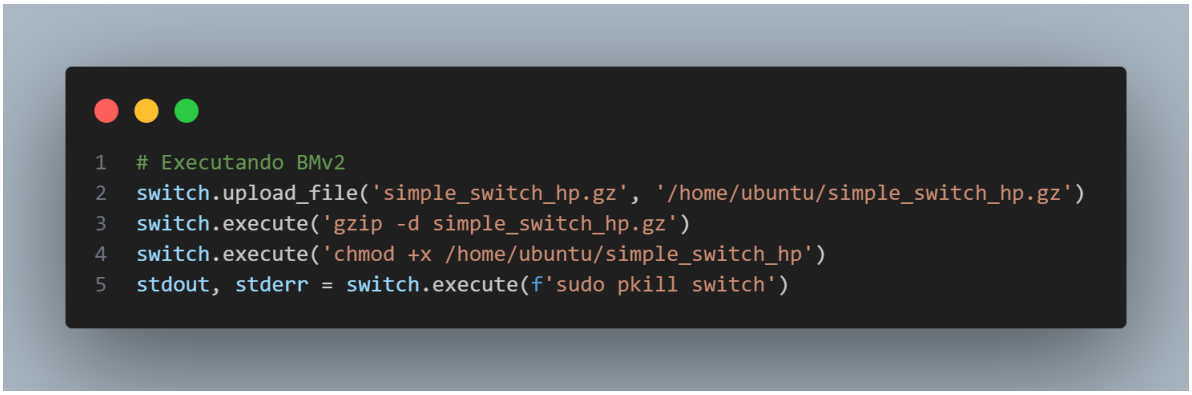
64 bytes from 192.168.2.10: icmp_seq=3 ttl=63 time=46.9 ms

— 192.168.2.10 ping statistics —

3 packets transmitted, 3 received, 0

rtt min/avg/max/mdev = 46.936/47.131/47.445/0.223 ms

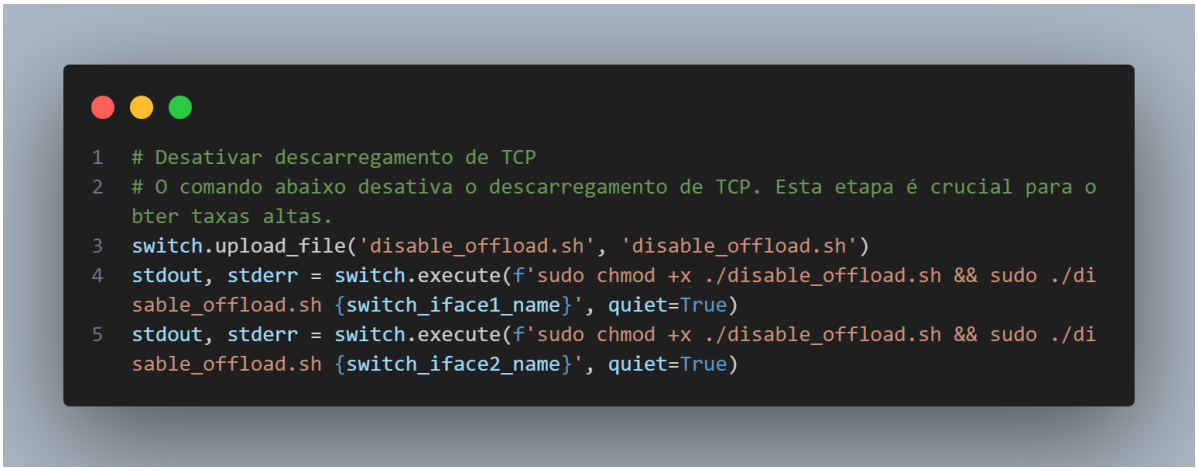
No trecho de código da Figura 38 executa um *switch* simples HP em nosso switch virtual P4.



```
# Executando BMv2
switch.upload_file('simple_switch_hp.gz', '/home/ubuntu/simple_switch_hp.gz')
switch.execute('gzip -d simple_switch_hp.gz')
switch.execute('chmod +x /home/ubuntu/simple_switch_hp')
stdout, stderr = switch.execute(f'sudo pkill switch')
```

Figura 38. *Executando o switch BMv2.*

Desativando download do TCP:



```
# Desativar descarregamento de TCP
# O comando abaixo desativa o descarregamento de TCP. Esta etapa é crucial para o
bter taxas altas.
switch.upload_file('disable_offload.sh', 'disable_offload.sh')
stdout, stderr = switch.execute(f'sudo chmod +x ./disable_offload.sh && sudo ./di
sable_offload.sh {switch_iface1_name}', quiet=True)
stdout, stderr = switch.execute(f'sudo chmod +x ./disable_offload.sh && sudo ./di
sable_offload.sh {switch_iface2_name}', quiet=True)
```

Figura 39. *Desativando descarregamento de TCP.*

Executando o *Switch* novamente depois de ter desabilitado o descarregamento TCP. Isso iniciará o switch de alto desempenho e alocará as interfaces e, em seguida, enviará as regras do plano de controle.

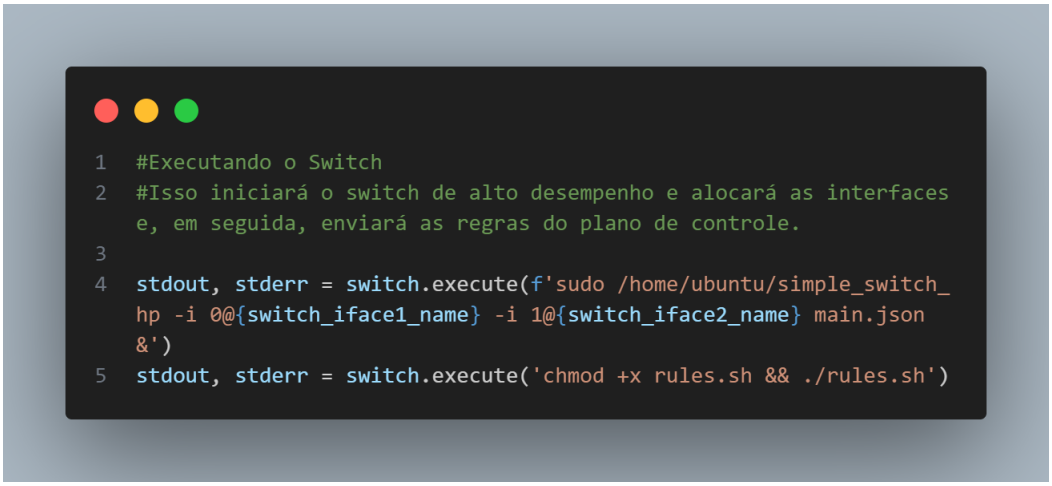


Figura 40. *Executando o Switch de alto desempenho.*

Ao executar o código acima teremos a seguinte resposta:

Calling target program-options parser

Adding interface enp7s0 as port 0

Adding interface enp8s0 as port 1

Obtaining JSON from switch...

Done

Control utility for runtime P4 table manipulation

RuntimeCmd: Adding entry to lpm match table MyIngress.ipv4_lpm

match key: LPM-c0:a8:01:00/24

action: MyIngress.forward

runtime data: 00:00:00:00:00:01 00:00

Entry has been added with handle 0

RuntimeCmd:

Obtaining JSON from switch...

Done

Control utility for runtime P4 table manipulation

RuntimeCmd: Adding entry to lpm match table MyIngress.ipv4_lpm

match key: LPM-c0:a8:02:00/24

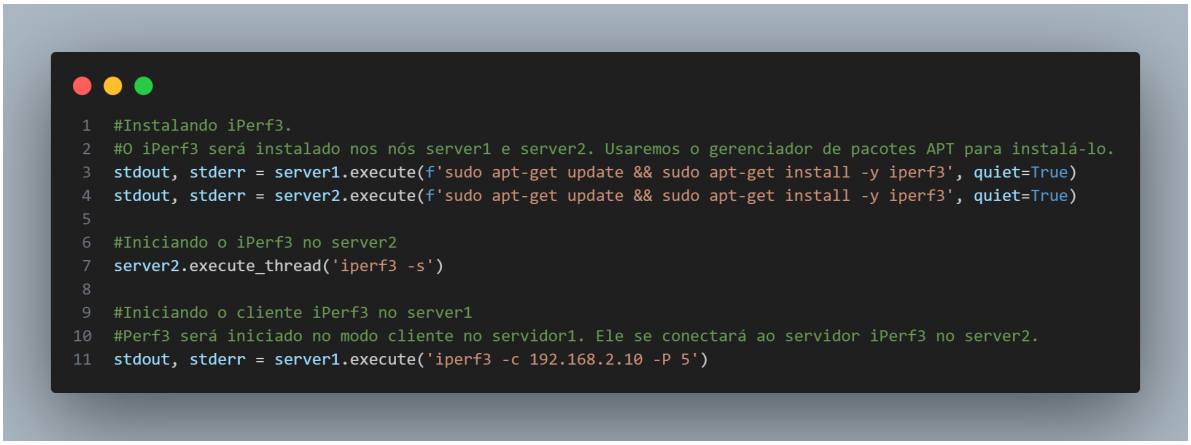
action: MyIngress.forward

runtime data: 00:00:00:00:00:04 00:01

Entry has been added with handle 1

RuntimeCmd:

Instalando iPerf3, que é uma ferramenta de teste de desempenho de rede que mede a largura de banda de uma conexão de rede entre dois pontos. Ele é amplamente utilizado para diagnosticar problemas de rede, testar a capacidade de largura de banda e verificar a qualidade da conexão. O iPerf3 será instalado nos nós server1 e server2. Usaremos o gerenciador de pacotes APT para instalá-lo.



```
#Instalando iPerf3.
#O iPerf3 será instalado nos nós server1 e server2. Usaremos o gerenciador de pacotes APT para instalá-lo.
stdout, stderr = server1.execute(f'sudo apt-get update && sudo apt-get install -y iperf3', quiet=True)
stdout, stderr = server2.execute(f'sudo apt-get update && sudo apt-get install -y iperf3', quiet=True)

#Iniciando o iPerf3 no server2
server2.execute_thread('iperf3 -s')

#Iniciando o cliente iPerf3 no server1
#Perf3 será iniciado no modo cliente no servidor1. Ele se conectará ao servidor iPerf3 no server2.
stdout, stderr = server1.execute('iperf3 -c 192.168.2.10 -P 5')
```

Figura 41. *Instalando iPerf3.*

Ao executar o código da Figura 41 teremos como resposta o teste feito pelo iPerf3; Em resumo temos um desempenho fluente com taxa de transferência variando consideravelmente ao longo dos intervalos, sugerindo variações nas condições da rede ou no comportamento do tráfego. Um certo número relativamente alto de retransmissões (829 no total) podendo indicar problemas de qualidade na rede, como congestionamento ou interferência. Existe também uma pequena diferença entre os dados reportados pelo remetente e pelo receptor, o que é comum devido a fatores como buffer de rede e latência.

Connecting to host 192.168.2.10, port 5201

[5] local 192.168.1.10 port 43996 connected to 192.168.2.10 port 5201

[7] local 192.168.1.10 port 44004 connected to 192.168.2.10 port 5201

[9] local 192.168.1.10 port 44016 connected to 192.168.2.10 port 5201

[11] local 192.168.1.10 port 44022 connected to 192.168.2.10 port 5201

[13] local 192.168.1.10 port 44030 connected to 192.168.2.10 port 5201

[ID] Interval Transfer Bitrate Retr Cwnd

[5] 0.00-1.00 sec 27.6 MBytes 231 Mbits/sec 247 2.54 MBytes

[7] 0.00-1.00 sec 18.1 MBytes 151 Mbits/sec 95 1.06 MBytes

[9] 0.00-1.00 sec 13.5 MBytes 114 Mbits/sec 358 874 KBytes

[11] 0.00-1.00 sec 9.29 MBytes 77.9 Mbits/sec 48 502 KBytes

[13] 0.00-1.00 sec 12.6 MBytes 106 Mbits/sec 1 727 KBytes

[SUM] 0.00-1.00 sec 81.0 MBytes 680 Mbits/sec 749

- -

[5] 1.00-2.00 sec 30.0 MBytes 252 Mbits/sec 56 1.33 MBytes

[7] 1.00-2.00 sec 20.0 MBytes 168 Mbits/sec 1 823 KBytes

[9] 1.00-2.00 sec 16.2 MBytes 136 Mbits/sec 1 675 KBytes

[11] 1.00-2.00 sec 8.75 MBytes 73.4 Mbits/sec 2 274 KBytes

[13] 1.00-2.00 sec 12.5 MBytes 105 Mbits/sec 2 561 KBytes

[SUM] 1.00-2.00 sec 87.5 MBytes 734 Mbits/sec 62

- -

[5] 2.00-3.00 sec 20.0 MBytes 168 Mbits/sec 2 826 KBytes

[7] 2.00-3.00 sec 17.5 MBytes 147 Mbits/sec 0 877 KBytes

[9] 2.00-3.00 sec 12.5 MBytes 105 Mbits/sec 1 595 KBytes

[11] 2.00-3.00 sec 5.00 MBytes 41.9 Mbits/sec 0 303 KBytes

[13] 2.00-3.00 sec 11.2 MBytes 94.4 Mbits/sec 0 598 KBytes

[SUM] 2.00-3.00 sec 66.2 MBytes 556 Mbits/sec 3

- -

[5] 3.00-4.00 sec 15.0 MBytes 126 Mbits/sec 2 379 KBytes

[7] 3.00-4.00 sec 16.2 MBytes 136 Mbits/sec 1 649 KBytes

[9] 3.00-4.00 sec 11.2 MBytes 94.4 Mbits/sec 0 551 KBytes

[11] 3.00-4.00 sec 6.25 MBytes 52.4 Mbits/sec 0 317 KBytes

[13] 3.00-4.00 sec 12.5 MBytes 105 Mbits/sec 0 619 KBytes

[SUM] 3.00-4.00 sec 61.2 MBytes 514 Mbits/sec 3

- -

[5] 4.00-5.00 sec 7.50 MBytes 62.9 Mbits/sec 0 419 KBytes

[7] 4.00-5.00 sec 13.8 MBytes 115 Mbits/sec 0 700 KBytes

[9] 4.00-5.00 sec 12.5 MBytes 105 Mbits/sec 0 584 KBytes

[11] 4.00-5.00 sec 7.50 MBytes 62.9 Mbits/sec 0 322 KBytes

[13] 4.00-5.00 sec 13.8 MBytes 115 Mbits/sec 0 631 KBytes

[SUM] 4.00-5.00 sec 55.0 MBytes 461 Mbits/sec 0

- -

[5] 5.00-6.00 sec 8.75 MBytes 73.4 Mbits/sec 0 443 KBytes

[7] 5.00-6.00 sec 15.0 MBytes 126 Mbits/sec 0 734 KBytes

[9] 5.00-6.00 sec 10.0 MBytes 83.9 Mbits/sec 1 441 KBytes

[11] 5.00-6.00 sec 6.25 MBytes 52.4 Mbits/sec 0 332 KBytes

[13] 5.00-6.00 sec 10.0 MBytes 83.9 Mbits/sec 1 474 KBytes

[SUM] 5.00-6.00 sec 50.0 MBytes 419 Mbits/sec 2

- -

[5] 6.00-7.00 sec 10.0 MBytes 83.9 Mbits/sec 0 455 KBytes

[7] 6.00-7.00 sec 11.2 MBytes 94.4 Mbits/sec 1 557 KBytes

[9] 6.00-7.00 sec 8.75 MBytes 73.4 Mbits/sec 0 472 KBytes

[11] 6.00-7.00 sec 5.00 MBytes 41.9 Mbits/sec 1 267 KBytes

[13] 6.00-7.00 sec 7.50 MBytes 62.9 Mbits/sec 1 365 KBytes

[SUM] 6.00-7.00 sec 42.5 MBytes 357 Mbits/sec 3

- -

[5] 7.00-8.00 sec 6.25 MBytes 52.4 Mbits/sec 1 351 KBytes

[7] 7.00-8.00 sec 11.2 MBytes 94.4 Mbits/sec 1 419 KBytes

[9] 7.00-8.00 sec 10.0 MBytes 83.9 Mbits/sec 0 491 KBytes

[11] 7.00-8.00 sec 6.25 MBytes 52.4 Mbits/sec 1 214 KBytes

[13] 7.00-8.00 sec 5.00 MBytes 41.9 Mbits/sec 1 284 KBytes

[SUM] 7.00-8.00 sec 38.8 MBytes 325 Mbits/sec 4

- -

[5] 8.00-9.00 sec 7.50 MBytes 62.9 Mbits/sec 0 372 KBytes

[7] 8.00-9.00 sec 8.75 MBytes 73.4 Mbits/sec 0 457 KBytes

[9] 8.00-9.00 sec 10.0 MBytes 83.9 Mbits/sec 0 499 KBytes

[11] 8.00-9.00 sec 3.75 MBytes 31.5 Mbits/sec 0 232 KBytes

[13] 8.00-9.00 sec 6.25 MBytes 52.4 Mbits/sec 1 219 KBytes

[SUM] 8.00-9.00 sec 36.2 MBytes 304 Mbits/sec 1

- -

[5] 9.00-10.00 sec 6.25 MBytes 52.4 Mbits/sec 1 287 KBytes

[7] 9.00-10.00 sec 10.0 MBytes 83.9 Mbits/sec 0 479 KBytes

[9] 9.00-10.00 sec 10.0 MBytes 83.9 Mbits/sec 1 383 KBytes

[11] 9.00-10.00 sec 5.00 MBytes 41.9 Mbits/sec 0 240 KBytes

[13] 9.00-10.00 sec 3.75 MBytes 31.5 Mbits/sec 0 238 KBytes

[SUM] 9.00-10.00 sec 35.0 MBytes 294 Mbits/sec 2

- -

[ID] Interval Transfer Bitrate Retr

[5] 0.00-10.00 sec 139 MBytes 116 Mbits/sec 309 sender

[5] 0.00-10.05 sec 136 MBytes 114 Mbits/sec receiver

[7] 0.00-10.00 sec 142 MBytes 119 Mbits/sec 99 sender

[7] 0.00-10.05 sec 139 MBytes 116 Mbits/sec receiver

[9] 0.00-10.00 sec 115 MBytes 96.3 Mbits/sec 362 sender

[9] 0.00-10.05 sec 111 MBytes 93.1 Mbits/sec receiver

[11] 0.00-10.00 sec 63.0 MBytes 52.9 Mbits/sec 52 sender

[11] 0.00-10.05 sec 60.4 MBytes 50.4 Mbits/sec receiver

[13] 0.00-10.00 sec 95.1 MBytes 79.8 Mbits/sec 7 sender

[13] 0.00-10.05 sec 92.7 MBytes 77.4 Mbits/sec receiver

[SUM] 0.00-10.00 sec 554 MBytes 464 Mbits/sec 829 sender

[SUM] 0.00-10.05 sec 540 MBytes 451 Mbits/sec receiver

iperf Done.

3.8 Passo 8: Exclusão da Fatia/Slice

Este último passo é executado após a conclusão dos testes e estudos de caso. Assim, liberamos os *hardwares* alocados para que outros desenvolvedores possam utilizá-los em seus projetos.

Para excluir o projeto, execute o seguinte código:

```
# Excluir a fatia
from fabrictestbed_extensions.fablib.fablib import FablibManager as fablib_manager
fablib = fablib_manager()
slice = fablib.get_slice(name="Tcc_Leydson")
slice.delete()
```

Figura 42. *Excluindo a Fatia.*

Fica claro que Fabric representa um avanço significativo no campo da pesquisa experimental. Com uma variedade de recursos, como CPUs, GPUs, FPGAs e armazenamento conectados por links ópticos de alta velocidade, oferecendo aos pesquisadores a capacidade de explorar novas fronteiras tecnológicas e abordar problemas complexos em diversas áreas.

No entanto, é importante reconhecer que a utilização do Fabric apresenta desafios que exigem uma compreensão profunda de sua proposta e a habilidade de gerenciar seus recursos para implementar topologias experimentais de forma eficaz. Para o desenvolvimento deste projeto, além de aprender a utilizar o Fabric e entender seu funcionamento, foi necessário ter conhecimentos prévios em redes, bem como em configuração de *switches* e servidores, um aspecto fundamental é o conhecimento técnico necessário para operar e configurar esses recursos. Competências em sistemas operacionais de redes, configuração de servidores e *switches*, e programação são essenciais para aproveitar ao máximo o potencial do Fabric. A curva de aprendizado pode ser maior devido à novidade da plataforma, com poucas referências e um fórum ainda pouco ativo. Além disso, o acesso restrito a indivíduos vinculados a instituições acadêmicas pode causar atrasos na execução de projetos.

Apesar dos desafios, o potencial do Fabric para impulsionar a inovação é enorme. Com seu compromisso com a colaboração e o avanço da pesquisa, ele pode se tornar uma ferramenta essencial para a comunidade acadêmica e científica. No entanto, acredito que alguns pontos podem ser aprimorados neste momento: a oferta de treinamentos adequados para novos desenvolvedores, o fortalecimento da colaboração entre os usuários e a utilização mais eficiente dos recursos de suporte oferecidos pela equipe do Fabric. Essas melhorias poderiam reduzir a curva de aprendizado e facilitar a exploração do potencial da plataforma, transformando-a em um catalisador ainda mais poderoso para o progresso na compreensão e aplicação das tecnologias de rede.

4 Revisão sistemática da literatura

Neste capítulo, exploraremos o método relacionado ao Roteiro de Avaliação Sistemática da Literatura, desenvolvido para detectar os estudos mais relevantes e apropriados relacionados à plataforma de prototipagem Fabric.

Nesta pesquisa, adotou-se uma Revisão Sistemática da Literatura (RSL). Conforme descrito por Kitchenham e Charters (2007), essa metodologia visa identificar e avaliar, por meio da análise de estudos primários, as informações pertinentes encontradas na literatura referentes a uma questão de pesquisa específica. Dessa forma, a avaliação sistemática é caracterizada como um estudo secundário.

A abordagem de Kitchenham é estruturada em três fases distintas: o planejamento da revisão, a execução da revisão e a etapa de relato dos resultados. Nas seções deste capítulo, serão delineados os procedimentos referentes à seleção dos artigos (KITCHENHAM, 2007).

4.1 Processo de planejamento

Na etapa de planejamento, adaptamos o modelo de Kitchenham e Charters (2007), inicialmente identificando duas questões de pesquisa para direcionar nosso estudo, delineando assim o escopo da pesquisa.

Após realizar uma pesquisa preliminar sobre o tema, explorando informações associadas à plataforma Fabric no próprio portal da plataforma e no Google, estabelecemos duas questões fundamentais: *Quais são os impactos do Fabric nas pesquisas de redes de computadores e ciência de dados?* e *Quais as principais dificuldades para se usar o Fabric em seus projetos de pesquisa?* As duas perguntas visam aprofundar a compreensão da aplicabilidade do Fabric em contextos de pesquisa.

A pesquisa se concentrará na análise de artigos que abordem o uso do Fabric no desenvolvimento de suas pesquisas, buscando reunir evidências sobre a plataforma e sua contribuição na implementação de soluções e no avanço de pesquisas como um todo.

4.2 Estratégias de busca

4.2.1 Perguntas de pesquisa

A justificativa para a escolha do tema desta pesquisa foi apresentada no Capítulo 1. Conforme mencionado anteriormente, o objetivo deste trabalho é abordar o Fabric, trazendo sua definição, seus impactos positivos e as dificuldades encontradas na aplicação em pesquisas e projetos. As questões centrais que orientaram o estudo foram: 'Quais são os impactos do Fabric nas pesquisas de redes de computadores e ciência de dados?' e 'Quais as principais dificuldades no uso do Fabric em projetos de pesquisa?'. Com base nessas perguntas, a pesquisa foi direcionada para

identificar artigos com aplicações práticas envolvendo o Fabric, de modo a obter respostas mais precisas.

Questão 1 - Q1: Quais são os impactos do Fabric nas pesquisas de redes de computadores e ciência de dados? **Questão 2 - Q2:**: Quais as principais dificuldades para se usar o Fabric em seus projetos de pesquisa?

Tabela 1. *Tabela de perguntas*

4.2.2 Definição de palavras-chave

Para podermos criar as expressões de busca, após a pesquisa inicial, definimos algumas palavras chave relacionadas ao tema deste estudo (ver Tabela 2).

Palavras Chave
FABRIC testbed
Adaptive Programmable Research
Computer Networks
Networking Technology
Communication Infrastructure
Data Science
Data Analytics
Machine Learning

Tabela 2. *Definição das palavras-chave.*

4.2.3 Expressão de busca

A pesquisa foi conduzida em três bases de dados, utilizando os motores de busca específicos de cada uma: IEEE, Scopus e ScienceDirect. Para efetuar essas buscas, elaboramos uma *string* de pesquisa para as quatro questões de pesquisa, utilizando as palavras-chave previamente estabelecidas como base para a construção da *string*.

String de busca: ((“FABRIC testbed” OR “Adaptive Programmable Research”) AND (“Computer Networks” OR “Networking Technology” OR "Communication Infrastructure") AND (“Data Science” OR "Data Analytics"OR "Machine Learning"))

Tabela 3. *Definição da string de busca*

De acordo com os parâmetros estabelecidos no modelo desenvolvido por Kitchenham e Charters (2007), formulamos critérios de inclusão e exclusão para os artigos obtidos nas fontes selecionadas, visando restringir a análise apenas aos artigos pertinentes ao escopo do estudo. Esses critérios específicos estão detalhados na Tabela 4.

Critérios de inclusão
CI1 - Estudos que usaram a plataforma Fabric
CI2 - Trabalhos escritos no idioma Inglês e Português
CI3 - Estudos sobre simuladores e testbeds

Tabela 4. *Definição dos critérios de inclusão.*

Critérios de exclusão
CE1 - Artigos que não estejam no período de 2018 a 2024
CE2 - Artigos secundários (outra RSL ou MS)
CE3 - Artigos duplicados ou similares
CE4 - Artigos que não estejam relacionados com as perguntas de pesquisa
CE5 - Artigos que não estejam em Inglês e Português
CE6 - Artigos que não seja sobre o FABRIC

Tabela 5. *Definição dos critérios de exclusão.*

4.3 Avaliação da qualidade dos artigos

A fim de analisar a excelência dos documentos selecionados, foram estabelecidos padrões de avaliação previamente estabelecidos antes das investigações, os quais foram aprimorados ao longo do processo de pesquisa, durante o qual adquiriu-se uma compreensão mais aprofundada sobre o tópico. De acordo com Kitchenham e Charters (2007), a avaliação da qualidade dos documentos é considerada crucial, uma vez que não apenas auxilia na definição mais precisa de critérios para inclusão e exclusão, mas também contribui para esclarecer eventuais disparidades nos resultados ao comparar dois estudos.

Critérios de avaliação
Contexto claro
Metodologia bem definida
Aplicação Prática
Discussão relevantes e consistentes

Tabela 6. *Definição dos critérios de avaliação.*

Cada documento deve ser atribuído com uma "classificação de qualidade", e para avaliar os critérios, adotaremos a seguinte escala: de "0 - não satisfatório"a "1.0 - totalmente satisfatório".

4.4 Processo de seleção de estudos primários

Para realizar a escolha dos artigos que atendem aos critérios mencionados anteriormente, seguimos um processo dividido em quatro fases:

1. Registrar as informações relacionadas aos estudos identificados. Posteriormente à aplicação das expressões de busca nas fontes investigadas.

2. Examinar o título, resumo e palavras-chave, aplicando os critérios de escolha.

3. Analisar a introdução e conclusão, levando em consideração os critérios de inclusão e exclusão.

4. Examinar todo o conteúdo do artigo, aplicando os critérios de inclusão e exclusão. Devem ser descartados trabalhos que não tenham relevância para as questões investigadas.

4.5 Execução do método

Após estabelecer o protocolo a ser empregado no estudo sistemático, deu-se início à implementação da pesquisa. Durante este período, foram conduzidas as buscas, aplicados filtros, realizada a leitura, classificação e análise dos textos. Posteriormente a essa etapa, procedeu-se à avaliação dos resultados da pesquisa.

Conforme mencionado anteriormente, foram empregadas três bases, a Scopus, IEEE e ScienceDirect, como fonte de artigos. A *string* de busca previamente definida foi aplicada, seguindo o processo de seleção de artigos descrito no capítulo anterior. A quantidade de artigos identificados em cada etapa do processo de seleção está detalhada nos quadros subsequentes.

Etapas do processo de seleção na base Scopus			
1ª etapa	**2ª etapa**	**3ª etapa**	**4ª etapa**
1	0	0	0

Tabela 7. *Tabela que descreve a quantidade de artigos selecionados em cada etapa do processo na base do Scopus*

Etapas do processo de seleção na base IEEE			
1ª etapa	**2ª etapa**	**3ª etapa**	**4ª etapa**
90	4	4	4

Tabela 8. *Tabela que descreve a quantidade de artigos selecionados em cada etapa do processo na base do IEEE*

Etapas do processo de seleção na base ScienceDirect			
1ª etapa	**2ª etapa**	**3ª etapa**	**4ª etapa**
1	1	1	1

Tabela 9. *Tabela que descreve a quantidade de artigos selecionados em cada etapa do processo na base do ScienceDirect*

Para calcularmos a nota média baseada nos critérios de avaliação, separamos essa média por bases, podemos verificar na Tabela 10 que a base Scopus teve uma média 0,00, pois não trouxe artigos elevantes para a pesquisa, já a base IEEE e ScienceDirect teveram uma média próximo a 1,00.

Média baseada nos critérios de aceitação		
Scopus	**IEEE**	**ScienceDirect**
0,00	0,90	1,00

Tabela 10. *Tabela que descreve a média baseada nos critérios de aceitação*

4.6 Tabelas com os artigos selecionados

A seguir são aprensentados em formato de tabela, os artigos selecionados:

Estudos selecionados				
Título	**Ano**	**Autor**	**Base**	**ID**
Network Traffic as a Federated Testbed Service	2022	Jack Brassil	IEEE	1
Experiments on Network Services for Video Transmission using FABRIC Instrument Resources	2023	Alicia Esquivel Morel, Prasad Calyam, Chengyi Qu, Durbek Gafurov, Cong Wang, Komal Thareja	IEEE	2
Network Services Management using Programmable Data Planes for Visual Cloud Computing	2023	Alicia Esquivel Morel, Prasad Calyam,Chengyi Qu, Durbek Gafurov, Cong Wang	IEEE	3
Evaluating SciStream (Federated Scientific Data Streaming Architecture) on FABRIC	2022	Chengyi Qu, Joaquin Chung, Zhengchun Liu, Tekin Bicer, Rajkumar Kettimuthu	IEEE	4
A survey on network simulators, emulators, and testbeds used for research and education	2023	Jose Gomez, Elie F. Kfoury, Jorge Crichigno, Gautam Srivastava	ScienceDirect	5

Tabela 11. *Tabela dos artigos selecionados*

4.7 Síntese do capítulo

Neste capítulo, foram apresentados os detalhes do método adotado na investigação, contemplando as atividades realizadas com base em técnicas sugeridas por Kitchenham. Explorou-se a abordagem utilizada para formular as perguntas de pesquisa derivadas da questão central, assim como o procedimento de elaboração do termo de busca. Abordaram-se também as peculiaridades da estratégia de pesquisa, como a significativa diminuição no número de estudos após as filtragens realizadas, destacando a importância da aplicação de critérios de qualidade e exclusão.

5 Análise e resultados

Neste capítulo, são apresentados os resultados obtidos a partir da seleção de cinco artigos mediante uma revisão sistemática da literatura nas bases de dados Scopus, IEEE e ScienceDirect. Esses resultados são complementados com os conteúdos didáticos oferecidos pelo portal Fabric para seus desenvolvedores. As próximas seções abordarão as soluções para as perguntas de pesquisa, com o objetivo de analisar os resultados identificados.

5.1 Quais os seus impactos voltados nas pesquisas de redes de computadores e ciência de dados?

Os *testbeds* dedicados à pesquisa em redes de computadores e ciência de dados desempenham um papel crucial em impulsionar avanços nessas áreas. Eles oferecem ambientes controlados onde os pesquisadores podem conduzir experimentos complexos, permitindo testes práticos de novos algoritmos, protocolos e tecnologias. Isso é fundamental, pois permite que as descobertas teóricas sejam validadas em um ambiente prático, ajudando a garantir que possam ser aplicadas na vida real.

Além disso, os testbeds promovem a inovação e o desenvolvimento ao permitir que pesquisadores e desenvolvedores testem novas ideias em um ambiente seguro e controlado. Isso leva ao desenvolvimento de soluções mais eficientes e avançadas, contribuindo para o avanço das redes de computadores e da ciência de dados.

Outro aspecto importante dos testbeds é o seu uso para fins educacionais. Eles fornecem uma plataforma onde estudantes e profissionais podem aprender sobre redes de computadores e ciência de dados por meio de experiências práticas. Isso é essencial para o desenvolvimento de habilidades práticas e para preparar a próxima geração de profissionais nessas áreas.

Além disso, os testbeds facilitam a colaboração e o compartilhamento de recursos entre diferentes instituições e comunidades de pesquisa. Muitos testbeds são projetados para serem acessíveis a pesquisadores de diferentes organizações, promovendo a colaboração e o intercâmbio de ideias e recursos.

No contexto específico do Fabric, suas contribuições são ainda mais profundas. O Fabric fornece uma infraestrutura de testbed avançada para explorar métodos e técnicas para superar as limitações arquiteturais da Internet. Ele permite experimentos em larga escala em redes de produção simuladas, o que possibilita testar novas estratégias de segurança e otimização de desempenho em um ambiente realista, mas controlado.

Além disso, não se limita apenas a redes de computadores, mas também suporta experimentos em ciência de dados, aprendizado de máquina e aplicações científicas. Isso abre novas oportunidades para a inovação e o avanço nessas áreas, contribuindo para uma compreensão mais profunda e uma aplicação mais eficaz dessas tecnologias (GOMEZ et al., 2023).

Em resumo, os testbeds, incluindo o FABRIC, desempenham um papel fundamental no

avanço das redes de computadores e da ciência de dados, proporcionando uma plataforma para experimentação, inovação, colaboração e educação. Suas contribuições são essenciais para impulsionar o progresso nessas áreas e para enfrentar os desafios complexos que enfrentamos na era digital. Para ilustrar essa importância, a seguir são apresentados alguns artigos nos quais o Fabric foi aplicado para o desenvolvimento da pesquisa e trouxe resultados significativos.

No artigo chamado **Network Traffic as a Federated Testbed Service**, é descrito o desenvolvimento e a implementação de um sistema chamado *Science Traffic as a Service (STAAS)*, projetado para facilitar a experimentação em testbeds de pesquisa experimental em redes de comunicação. O STAAS é uma plataforma descentralizada e cooperativa que coleta, filtra e distribui dados de tráfego real e sintético para a comunidade global de pesquisadores de redes (BRASSIL, 2022).

O sistema oferece tráfego sob demanda por meio de um painel web, proporcionando aos pesquisadores uma maneira eficiente de acessar e utilizar diferentes tipos de tráfego para suas investigações. Ele aborda desafios enfrentados pelos pesquisadores na criação de tráfego realista, especialmente com o aumento da velocidade das redes de backbone para 100-1000 Gbps e usou o Fabric como a plataforma principal para o desenvolvimento e implementação do protótipo, contando como uma escolha atraente para realizar experimentos em redes de comunicação de próxima geração (BRASSIL, 2022).

Já o artigo **"Evaluating SciStream (Federated Scientific Data Streaming Architecture) on FABRIC"** avalia uma arquitetura de streaming de dados científicos federados, que foi testada na infraestrutura de rede nacional Fabric, fornecendo um ambiente necessário para testar o SciStream em escala, com sua rede de Terabit e supercore de 100 Gbps, além de grande capacidade de computação e armazenamento. Contribuindo também com a criação de topologias de rede variadas, incluindo diferentes valores de tempo (RTT) de ida e volta, ambientes LAN quanto WAN. Dessa forma, foi possível avaliar o desempenho do SciStream sob diversas configurações de rede, testando diferentes implementações e medindo sua eficácia em termos de taxa de transferência e jitter. O SciStream foi desenvolvido para permitir o streaming eficiente e seguro de dados entre produtores e consumidores de dados científicos, que geralmente estão localizados em diferentes domínios de segurança e não possuem conectividade direta de rede. Essa solução é importante para fluxos de trabalho científicos modernos, que exigem a redução de dados, detecção de características e controle de experimentos em tempo real (QU et al., 2022).

O artigo **"Network Services Management using Programmable Data Planes for Visual Cloud Computing"** explora o uso de planos de dados programáveis, especialmente utilizando a linguagem P4, para melhorar a gestão de serviços de rede em aplicações de computação em nuvem visual (VCC). Envolvendo o processamento de grandes volumes de dados de vídeo, que são transmitidos através de infraestruturas de borda-nuvem. O objetivo principal é proporcionar um controle mais refinado e monitoramento em tempo real para garantir a qualidade da experiência dos usuários (QoE) ao lidar com tarefas críticas, como o reconhecimento facial ou

a análise de vídeos de drones (MOREL et al., 2023). Os experimentos foram realizados no Fabric, onde os pesquisadores utilizaram metadados de telemetria por pacote, como rotas de encaminhamento, marcas de tempo de entrada/saída, ocupação de filas em nós específicos e utilização de portas de egressão. Esses experimentos permitiram validar a eficácia das técnicas propostas, demonstrando como a implementação em P4 pode melhorar o desempenho das redes para aplicações de computação em nuvem visual.

Os resultados desses experimentos são significativos para demonstrar o poder do FABRIC no estudo e teste de novas tecnologias. Trabalhos futuros podem explorar ainda mais as capacidades do Fabric, aprimorando a qualidade dos resultados. Esses estudos destacam o papel crucial do Fabric na pesquisa e desenvolvimento de infraestruturas de rede avançadas e na experimentação de aplicações em larga escala.

5.2 Quais as principais dificuldades para se usar o Fabric em seus projetos de pesquisa?

Embora o Fabric ofereça uma infraestrutura avançada e recursos poderosos para pesquisa em redes de computadores e ciência de dados, ainda não está amplamente disponível para todos os pesquisadores. Isso pode dificultar o acesso de indivíduos e instituições que não fazem parte de redes de colaboração ou que não possuem os recursos necessários para participar de suas iniciativas (GOMEZ et al., 2023).

Existem algumas dificuldades que os pesquisadores podem enfrentar ao utilizá-lo em seus projetos, a configuração dos ambientes experimentais Fabric pode ser complexa, especialmente quando se trabalha com dispositivos programáveis, como *switches* P4. A instalação de sistemas operacionais de rede, dependências e a configuração do ambiente para desenvolvimento requerem habilidades avançadas, o que pode ser um obstáculo para muitos experimentadores, exigindo conhecimentos técnicos especializados para configurar e utilizar efetivamente seus recursos. Isso pode representar uma barreira para pesquisadores menos experientes ou com habilidades específicas limitadas (GOMEZ et al., 2023).

Devido à sua complexidade, pode haver uma curva de aprendizado maior para os pesquisadores que estão começando a usar a plataforma pela primeira vez. Compreender completamente os recursos disponíveis, as melhores práticas de uso e a configuração adequada como os ambientes de desenvolvimento e o chaveamento entre um projeto e outro pode levar tempo.

No entanto, muitas dessas dificuldades podem ser superadas com o tempo, treinamento adequado, colaboração com outros pesquisadores e aproveitando os recursos de suporte oferecidos pela equipe do Fabric e pela comunidade de usuários.

6 Conclusão

Neste trabalho, exploramos o Fabric como uma infraestrutura de pesquisa, destacando suas vantagens e limitações, além de sua aplicabilidade prática. A análise realizada demonstra como a plataforma facilita a experimentação em larga escala, viabilizando testes de novos protocolos e a análise eficiente de grandes volumes de dados.

Além disso, este estudo oferece como uma de suas contribuições uma documentação clara e didática sobre o uso do Fabric, podendo ser de grande utilidade para pesquisadores tanto iniciantes quanto experientes que desejam explorar o potencial da plataforma. A metodologia prática para implementação e configuração apresentada aqui serve como uma referência, adaptável a diferentes projetos e áreas de pesquisa.

Outra contribuição significativa deste trabalho é a revisão sistemática da literatura, que sintetiza os principais desafios e benefícios relatados por outros pesquisadores ao utilizar o Fabric. Essa análise destaca a importância da plataforma no avanço de tecnologias de redes e ciência de dados, além de propor direções futuras para otimizar seu uso. Ao documentar as dificuldades e propor soluções práticas, o trabalho enriquece o entendimento sobre plataformas de experimentação em rede, facilitando o desenvolvimento de novos protocolos e arquiteturas de rede.

Por fim, este estudo contribui para o debate sobre o futuro das plataformas de experimentação, como o Fabric, promovendo sua aplicação em campos emergentes como inteligência artificial e segurança cibernética. Esperamos que as descobertas e práticas aqui documentadas sirvam como base para futuros trabalhos, impulsionando a inovação e incentivando a pesquisa colaborativa em redes de computadores e ciência de dados.

Referências

BRASSIL, J. Network traffic as a federated testbed service. In: **2022 IEEE Future Networks World Forum (FNWF)**. [S.l.: s.n.], 2022. p. 450–455.

ENDERAMI, S. A. et al. Virtual testbeds for community resilience analysis: step-by-step development procedure and future orientation. **Resilient Cities and Structures**, v. 2, n. 2, p. 42–56, 2023. ISSN 2772-7416. Integrated Modeling of Cities to Improve Natural Hazards Resilience. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2772741623000376>.

FABRIC Control Framework Design. v. 0.6, p. 41, 2022. Disponível em: <https://learn.fabric-testbed.net/knowledge-base/design-documents/>.

FABRIC Network Services and Peering Design. v. 1.0, p. 47, 2021. Disponível em: <https://learn.fabric-testbed.net/knowledge-base/design-documents/>.

FREQUENTLY Asked Questions FAQ. 2024. Disponível em: <https://learn.fabric-testbed.net/knowledge-base/frequently-asked-starter-questions/>.

GOMEZ, J. 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. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128623004991>.

KITCHENHAM, C. **Kitchenham B., Charters S., Guidelines for performing systematic literature reviews in software engineering**. [S.l.], 2007.

MOREL, A. E. et al. Network services management using programmable data planes for visual cloud computing. In: **2023 International Conference on Computing, Networking and Communications (ICNC)**. [S.l.: s.n.], 2023. p. 130–136.

QU, C. et al. Evaluating scistream (federated scientific data streaming architecture) on fabric. In: **2022 IEEE/ACM International Workshop on Innovating the Network for Data-Intensive Science (INDIS)**. [S.l.: s.n.], 2022. p. 25–31.