Capítulo 4

Gestão de Identidade em Redes Experimentais para a Internet do Futuro

Edelberto Franco Silva (UFF), Natalia Castro Fernandes (UFF), Noemi Rodriguez (PUC-RIO), Luiz Claudio Schara Magalhães (UFF), Débora Christina Muchaluat Saade (UFF)

Abstract

Limitations of the current Internet architecture and protocols have been motivating several research initiatives about alternative solutions for the Future Internet, as it is called. Different research groups have been presenting new proposals that have to be tested and validated in real interconnected network testbeds. In this scenario, the federation concept can be applied to provide collaboration among researchers in a reliable way. Therefore, functionalities related to identity management become crucial. This chapter discusses the main proposals found in the literature for identity management in Future Internet research projects, and also open issues considering virtual organizations and experimentation in software defined network testbeds.

Resumo

As limitações impostas pela arquitetura da Internet atual têm gerado um grande interesse e incentivo à pesquisa relacionada a possíveis arquiteturas alternativas. Desta forma, diversos grupos de estudo ao redor do mundo têm unido esforços para a criação e propostas de novas soluções. A Internet do Futuro, como vem sendo chamada, deverá, antes, ser testada e validada por meio de ambientes mais próximos o possível do real, o que impulsiona a necessidade de união dos esforços da comunidade de pesquisa com relação à interconexão dos ambientes de experimentação de novas arquiteturas e protocolos. Neste contexto, surge a necessidade de emprego do conceito de federação, onde, a partir da colaboração dos participantes, ambientes de experimentação podem ser compartilhados de forma confiável. Sobre este novo paradigma, há a necessidade de gestão de identidade. Este capítulo discute as principais propostas para gestão de identidade em projetos de Internet do Futuro, encontradas na literatura, e também os problemas ainda em aberto, considerando o conceito de organizações virtuais e a experimentação em redes definidas por software.

4.1. Introdução

As limitações impostas pela arquitetura da Internet atual têm gerado um grande interesse e incentivo à pesquisa relacionada a possíveis arquiteturas alternativas. A Internet do Futuro, como vem sendo chamada, deverá, antes, ser testada e validada por meio de ambientes mais próximos o possível do real.

O conceito de virtualização de redes [FGR07, BFH+06a] tem sido empregado para a criação de ambientes de testes de novas arquiteturas e protocolos bastante flexíveis. A flexibilidade resulta da utilização de equipamentos de rede e de técnicas de virtualização, com as chamadas redes definidas por software (*Software Defined Networks* - SDN). Uma tecnologia de SDN que permite a coexistência do tráfego da Internet atual com o tráfego de novos experimentos é o OpenFlow [MAB+08], que obteve grande projeção e vem sendo bastante utilizado em redes de experimentação (*testbeds*) para a Internet do Futuro.

Várias redes experimentais têm sido criadas nos últimos anos [gen13b, fir13]. Diversos grupos de estudo ao redor do mundo vêm recentemente unindo esforços para a criação e propostas de novas soluções, levando à necessidade de interconexão dos ambientes de experimentação de novas arquiteturas e protocolos. A interconexão desses ambientes permite que pesquisadores de diferentes instituições possam utilizar, em seus experimentos, recursos de rede que são administrados por diferentes organizações, que por sua vez seguem diferentes políticas e procedimentos de uso.

O ambiente integrado resultante traz requisitos bastante desafiadores em termos de autenticação de usuários, autorização para uso de recursos e alocação desses recursos, funções relacionadas ao conceito de gestão de identidade.

Este capítulo tem como objetivo principal discutir as principais questões de pesquisa e propostas encontradas na literatura para gestão de identidade em redes experimentais para a Internet do Futuro. Por ser um problema recente, ainda não existe uma única solução bem definida e que resolva as diversas questões envolvidas.

O restante do capítulo está estruturado da seguinte maneira. A segunda seção aborda os principais conceitos relacionados à gestão de identidade, definindo as principais funções dos provedores de identidade e provedores de serviço, e apresentando o conceito de federação de autenticação e autorização (A&A). A terceira seção discute a gestão de identidade em organizações virtuais, tais como as organizações definidas por pesquisadores de diferentes instituições e a utilização de redes experimentais distintas e heterogêneas, apontando os principais desafios. A quarta seção comenta os conceitos básicos relacionados a redes definidas por software e apresenta brevemente a solução de virtualização de redes do OpenFlow. A quinta seção discute os requisitos para federação de ambientes de experimentação para a Internet do Futuro e apresenta de forma sucinta as principais iniciativas americana GENI (*Global Environment for Network Innovations*) [gen13b] e européia FIRE (*Future Internet Research and Experimentation Initiative*) [fir13]. São discutidos os procedimentos de autenticação e autorização para uso de recursos em cada um desses ambientes. A sexta seção discute as propostas encontradas na literatura para integração entre os diversos *testbeds* e projetos de redes experimentais para a Internet do Futuro. São apresentadas as propostas SFA (*Slice-based Federation Architecture*) [PRFC10b], *Panlab Federation (Teagle)* [GBW+06] e NOVI (*Networking Inno-*

vations Over Virtualized Infrastructures) [LGP+12]. Finalmente, a última seção apresenta as considerações finais deste capítulo.

4.2. Gestão de Identidade e Federações de Autenticação e Autorização

A *gestão* (ou gerência) *de identidade* (GId) pode ser entendida como o conjunto de processos e tecnologias usados para garantir a identidade de uma entidade, garantir a qualidade das informações de identidade (identificadores, credenciais e atributos) e utilizar essas garantias em procedimentos de autenticação, autorização e auditoria [IT09, WMB+10]. Ainda que objetos arbitrários possam estar associados a identidades, neste texto, estamos interessados em identidades associadas a usuários finais.

Os mecanismos para construção de sistemas de GId em ambientes circunscritos geográfica e administrativamente estão bastante bem definidos. No contexto de sistemas geograficamente distribuídos, no entanto, ainda há muito a ser entendido e resolvido. O cenário em que estamos interessados é aquele onde usuários de diferentes instituições necessitam acesso a recursos que podem ser mantidos por uma dessas instituições ou por alguma organização externa. Cada uma dessas instituições e organizações constitui um domínio administrativo diferente, com suas próprias equipes e políticas. Para dar suporte a interações envolvendo diferentes domínios administrativos, é necessário que determinados serviços e aplicações aceitem acessos vindos de usuários oriundos de outros domínios. O ambiente de colaboração acadêmica traz ainda requisitos de conveniência bastante próprios.

O uso de certificados digitais e de infraestruturas de chaves públicas (ICPs) vem crescendo bastante nos últimos anos e é um elo importante na criação de infraestruturas de autenticação, garantindo a qualidade de procedimentos de autenticação remotos. No entanto, o uso isolado dessa tecnologia não é suficiente para dar apoio à colaboração entre instituições de forma confiável e conveniente. Uma das questões centrais é que não faz sentido, em termos de escala nem de confiabilidade, que cada domínio administrativo "conheça" individualmente cada um de seus usuários advindos de outros domínios. Por "conhecer" aqui nos referimos a uma série de atividades como registrar usuários, gerenciar e distribuir credenciais, atualizar seus dados quando necessário, e manter registro das autorizações associadas a cada um deles. A replicação desses processos em cada domínio detentor de um serviço de interesse do usuário torna praticamente impossível garantir a consistência entre as bases de dados. Uma outra consideração importante é a privacidade do usuário. Ao estender o processo de registro a usuários externos, os sistemas institucionais tendem a requisitar deles todos os dados exigidos de usuários locais, o que na maioria das vezes não faz sentido.

Para manter um processo escalável de autenticação e autorização nesses ambientes de colaboração distribuída, é interessante que cada informação (de identidade, atributos ou autorizações) seja mantida apenas em um ponto do sistema, de forma a facilitar sua consistência e qualidade. Ao acessar um determinado recurso, o usuário pode autorizar, explícita ou implicitamente, a transferência de informações necessárias para esse acesso entre o domínio que as mantém e o domínio responsável pelo recurso. Sempre que possível, o procedimento de autenticação deve ser realizado entre um usuário e um único ponto (serviço) no sistema.

Modelos de identidade que procuram satisfazer esses requisitos vêm sendo bastante estudados nos últimos anos [JP05, WMB+10]. *Modelos centralizados* definem uma única autoridade central que fica responsável pela autenticação e outras informações (atributos) de todos os usuários, enviando as garantias e atributos necessários para diferentes serviços. *Modelos federados* definem uma associação entre domínios, na qual podem existir diferentes autoridades de autenticação e de atributos, e diferentes provedores de serviços. Essa segunda arquitetura, que utiliza o conceito de federação, reflete naturalmente o contexto de colaboração entre instituições de ensino e pesquisa, onde cada organização já é naturalmente candidata a autoridade de informações sobre seus usuários.

Uma *federação*, em nosso contexto, é uma infraestrutura de autenticação e autorização interdomínios, muitas vezes chamada de *infraestrutura de autenticação e autorização (A&A) federada*. O objetivo dessa infraestrutura é permitir que um usuário tenha seus dados armazenados e gerenciados em uma única instituição — em nosso contexto a instituição acadêmica à qual pertence, chamada aqui de *instituição de origem do usuário* (*home organization*) — mas que ao mesmo tempo possa ter acesso a recursos oferecidos em diferentes domínios. A manutenção de tal infraestrutura envolve não só a utilização de tecnologias e serviços, mas também a gerência de acordos e contratos entre as entidades envolvidas.

Uma infraestrutura de autenticação e autorização federada é constituída por dois tipos de elementos principais:

provedores de identidade: responsáveis pela manutenção e fornecimento das informações sobre usuários e por sua autenticação;

provedores de serviço: responsáveis por um ou mais serviços (ou recursos) oferecidos.

No caso de um serviço acessado via web, o suporte à autenticação federada se dá da seguinte forma: ao direcionar seu navegador para determinado serviço, o usuário recebe um pedido para selecionar seu provedor de identidade dentro de uma lista de provedores. O usuário escolhe então sua instituição de origem, e seu navegador é redirecionado para o provedor de identidade dessa instituição. A seguir, o provedor de identidade realiza o procedimento de autenticação do usuário, utilizando credenciais e métodos de sua escolha, que não precisam ser os mesmos utilizados por outros provedores na mesma federação. Após autenticar o usuário, o provedor de identidade repassa o resultado dessa autenticação ao provedor de serviço e cria uma sessão de uso associada ao usuário, A Figura 4.1 ilustra as interações realizadas durante um acesso típico, via navegador, a um serviço federado. O fluxo apresentado assume que nenhuma informação sobre o usuário é conhecida pelo provedor do serviço, e que este é o primeiro acesso do usuário a um serviço federado. Acessos a novos serviços dentro da mesma sessão de uso — cujo tempo de vida depende de parâmetros de configuração — não geram nova necessidade de autenticação do usuário (conceito de *single sign-on*).

Cenários típicos de aplicação dessa infraestrutura são serviços oferecidos a toda a comunidade acadêmica brasileira, como o repositório de periódicos da CAPES ou os serviços de conferência web e vídeo sob demanda da RNP (Rede Nacional de Ensino e

Figura 4.1. Acesso a serviços federados via navegador web

Pesquisa). Em muitos países da Europa, as editoras acadêmicas assinam contratos diretamente com universidades, e oferecem o acesso a seus acervos através das federações nacionais. Um outro exemplo de serviço que também utiliza o conceito de federação é o serviço eduroam (*education roaming*) [edu13], que oferece acesso seguro a rede sem fio para a comunidade acadêmica internacional.

Além da garantia de autenticação, o provedor de serviço poderá necessitar de outras informações (atributos) relativas ao usuário. Um caso típico é a solicitação de algum identificador único, associado ao usuário, que, ainda que não permita conhecer a identidade real do usuário, permita identificá-lo entre visitas consecutivas, para manutenção de preferências e histórico. Um outro caso especialmente relevante é o de uso de atributos para definição de autorizações. O controle de acesso deve ser definido no domínio do serviço, que tipicamente sequer conhece a identidade do usuário. A associação de autorizações a identidades específicas não seria recomendável de qualquer forma, pois traz dificuldades de atualização e volta a onerar a administração do serviço. Federações de A&A utilizam o modelo de autorização baseada em atributos (ABAC - *attribute-based access control*). Nesse modelo, um serviço define as autorizações de acesso em termos de atributos do usuário, como por exemplo a natureza de seu vínculo com a instituição (estudante, professor, funcionário, etc). Caso um atributo relevante seja alterado, automaticamente isso se refletirá em suas autorizações. Depois que um usuário é autenticado, um provedor de serviço pode requisitar, a uma *autoridade de atributos*, os atributos necessários para definir suas autorizações. A maior parte das infraestruturas de autenticação e autorização federada atuais inclui apenas uma autoridade de atributos por usuário, autoridade essa que reside no provedor de identidade.

A privacidade e controle individual de dados pessoais é um dos objetivos do desenvolvimento de federações de autenticação e autorização. A transferência de dados entre provedores de identidade e de serviço depende de acordos bilaterais entre essas en-

tidades. Além disso, as tecnologias utilizadas como base para as federações permitem que o usuário seja consultado sobre sua concordância com a transferência de determinado dado.

Exemplos de serviços que podem se beneficiar do uso de atributos são sistemas de vendas com descontos para professores ou alunos, ou sistemas de prontuários médicos onde a informação esteja disponível apenas para médicos e enfermeiros.

O protocolo mais usado em federações acadêmicas é o SAML [OAS05]. Há várias implementações do SAML disponíveis, dentre as quais se destacam, em termos de uso, o *Shibboleth* [Int13], desenvolvido no âmbito do projeto *Internet2*, nos EUA, e o SimpleSAML, desenvolvido pela UNINNETT, na Noruega.

Um elemento básico de uma federação é a relação de confiança entre provedores de serviço e provedores de identidade. Provedores de serviço têm que confiar na qualidade dos dados que lhes são fornecidos pelos provedores de identidade. Por outro lado, provedores de identidade têm que confiar nos provedores de serviço a quem fornecem dados de usuários, que devem garantir que utilizarão esses dados apenas para os fins combinados. Assim, um aspecto da arquitetura de uma federação é a gerência de informações sobre provedores de identidade e de serviços que têm sua chancela, chamados de *participantes* da federação. A gerência de informações sobre provedores de identidade e de serviços que têm a chancela de uma federação é feita através da manutenção de *metadados*: a operação da federação mantém disponíveis, para os participantes, os dados (endereço do servidor que atua como provedor de identidade, certificado de servidor, etc) sobre cada uma das instituições participantes.

Os metadados da federação também são usados para alimentar o *serviço de descoberta*, que permite ao usuário identificar sua instituição de origem (também conhecido como WAYF – *Where Are You From*). Esse serviço permite a exibição da lista de provedores de identidade para que o usuário selecione sua instituição de origem. Em um serviço web, quando o usuário direciona seu navegador para um serviço federado, na realidade seu navegador é redirecionado para o WAYF, que por sua vez o redireciona para a instituição escolhida. (Por simplicidade, a Figura 4.1 não mostra esse passo.)

4.3. Gestão de Identidade em Organizações Virtuais — Desafios

Um conjunto de pesquisadores e a utilização de um ambiente de experimentação de Internet do Futuro formam, tipicamente, uma *organização virtual* (OV), uma organização cujos membros e recursos estão espalhados por diferentes instituições e domínios administrativos [Gem06]. A participação de membros em uma organização virtual é iniciada ou cancelada segundo regras externas às organizações às quais são diretamente ligados. A importância de organizações virtuais vem crescendo continuamente, com exemplos como sociedades científicas, grupos de pesquisa multi-institucionais, comitês interinstitucionais formados para prover pareceres médicos ou acadêmicos, e muitos outros.

Dada a evolução na oferta do controle de acesso federado, é natural pensar em utilizar a mesma infraestrutura para o controle de acesso nas muitas organizações virtuais presentes na comunidade acadêmica. Se o acesso desejado é feito via web, o procedimento de autenticação de usuários pode facilmente ficar por conta da federação. No

entanto, a autorização de acesso depende de atributos definidos pela OV e não pela instituição de origem do usuário, sendo o mais básico entre esses atributos a própria participação do usuário na organização virtual em questão. A instituição de origem do usuário, que tipicamente atua como sua autoridade de atributos, não detém essa informação. Surge então a questão de como integrar organizações virtuais nas infraestruturas federadas disponíveis hoje.

Uma forma de resolver essa questão seria associar a cada organização virtual seu próprio provedor de identidade, responsável pelos atributos específicos dessa OV. No entanto, isso traria de volta todos os problemas associados a redundância: novas credenciais para cada usuário perante cada organização virtual e manutenção de informação replicada, uma vez que a autorização perante um recurso qualquer pode necessitar tanto de informações específicas à organização virtual, como a participação do pesquisador, como de outras disponíveis em sua instituição de origem, como por exemplo a natureza de seu vínculo.

Para evitar essa duplicação, é interessante que serviços oferecidos por organizações virtuais possam confiar a autenticação de usuários a suas instituições de origem, porém que possam consultar diferentes autoridades de atributo para consolidar a informação necessária para determinar as autorizações de um usuário. Nesse modelo de múltiplas autoridades de atributo [Fer12], cada organização virtual manteria os atributos que lhe dizem respeito. No entanto, não é trivial fazer a ligação entre os diversos cadastros de um usuário sem utilizar explicitamente identificadores que comprometam a privacidade. Discutiremos esse problema na Seção 4.3.1.

Ainda uma outra questão a levar em consideração em ambientes de experimentação é que o acesso via web pode não ser a forma mais conveniente de configurar testes. Pode ser desejável que aplicações de teste possam configurar diretamente a rede, via APIs, ao invés de sua execução depender sempre de uma etapa anterior de configuração manual via um portal web. Há projetos em andamento para dar suporte à execução de aplicações não web em infraestruturas federadas de autenticação e autorização, mas ainda não há soluções amplamente disponíveis.

4.3.1. Ligação de Identidades e Agregação de Atributos

Na arquitetura básica de federações, todos os atributos de um usuário estão disponíveis em um único ponto da federação, que também é responsável pela autenticação do usuário. No entanto, à medida que federações crescem e se tornam mais complexas, há uma série de situações que requerem que cadastros de um usuário sejam mantidos em mais de um ponto da federação.

Em alguns cenários, um usuário tem sua identidade registrada e autenticada em mais que uma instituição, pois cada uma dessas não pode ou não admite utilizar o processo de autenticação de outra. Um exemplo seria um estudante universitário possuidor de cartão de crédito. A universidade à qual o usuário está ligado e a instituição financeira que mantém o cartão mantêm, tipicamente, procedimentos independentes de autenticação. Nesse cenário, pode ser importante para um provedor de serviço receber atributos vindos das duas instituições com a garantia de que se referem a um mesmo indivíduo. Por exemplo, um vendedor de livros pode requisitar o status de estudante para fornecer um desconto

e a informação de crédito para completar a venda.

Em outros cenários, pode ser viável que o procedimento de autenticação seja realizado pela instituição de origem do usuário, mas é necessário que outras instituições agreguem novos atributos àqueles mantidos pelo provedor de identidade de origem. Por exemplo, um estudante universitário que pertence a sociedades acadêmicas pode manter credenciais de autenticação apenas perante sua instituição de origem. Uma sociedade acadêmica pode perfeitamente delegar a autenticação de seus membros a suas instituições acadêmicas, mas precisará manter atributos não fornecidos por aquelas, como por exemplo a própria participação na sociedade. Esse seria também um cenário de organizações virtuais, compostas por usuários que possuem um vínculo principal, mas que são participantes de um projeto ou grupo de pesquisa bem definido.

Nesses dois casos surge uma mesma questão. Como garantir que as informações que são mantidas em diferentes instituições se referem a um mesmo indivíduo mantendo, ao mesmo tempo, garantias de privacidade? Esse problema, conhecido como *agregação de atributos*, vem recebendo atenção nos últimos anos [CIK10]. No entanto, não existem ainda soluções padronizadas ou completamente incorporadas nas tecnologias de federação atuais.

4.4. Ambientes de experimentação em Internet do Futuro

As pesquisas relacionadas ao tema Internet do Futuro têm como objetivo o desenvolvimento de novas arquiteturas que permitam um uso da rede mais eficiente e funcional [BC01]. Entre as principais vertentes de pesquisa, estão as propostas *clean-slate*, que propõem que a nova arquitetura da rede deveria ser pensada de forma independente da arquitetura atual. Com base nos conhecimentos sobre problemas e soluções para a Internet, um pesquisador poderia propor uma nova solução que não precisasse ser compatível com a arquitetura atual. Outra vertente de pesquisa apoia que uma nova solução deva ser estabelecida com base em uma evolução da Internet atual [RD10]. Em ambas as vertentes, existe o consenso de que todas as novas propostas precisam ser testadas em larga escala e sob diferentes cenários antes de se propor uma modificação no ambiente atual. Além disso, é necessário desenvolver um ambiente para a transição entre a Internet atual e a nova arquitetura.

Nesse contexto, surgiu o conceito da virtualização de redes [FGR07, BFH+06a]. De fato, para testar uma nova arquitetura de forma escalável, seria necessário utilizar, de alguma forma, os equipamentos de rede já existentes. Além disso, se fosse possível conciliar o funcionamento da rede atual com uma nova arquitetura de rede sobre o mesmo equipamento, o ambiente para a migração estaria automaticamente construído. Assim, foram iniciadas as pesquisas para a construção de *experimental facilities*, que seriam redes de teste (*testbeds*) extremamente flexíveis e baseadas em virtualização. A ideia é que a rede de teste fosse tão flexível que seria possível testar qualquer tipo de arquitetura. A escalabilidade viria da utilização de equipamentos da rede atual por meio de técnicas de virtualização. Dentro dessas iniciativas, destacou-se a da Universidade de Stanford, que construiu uma nova arquitetura de rede dentro da universidade, a qual permitia grande flexibilidade no controle e gerenciamento da rede, além de permitir que o tráfego experimental e o tráfego de produção coexistissem sobre o mesmo equipamento de forma

independente. Essa nova arquitetura foi chamada de OpenFlow e, devido às suas características de programabilidade, ganhou grande projeção [MAB+08].

A seguir, são apresentados os principais conceitos das redes OpenFlow, dando um destaque a como é feita a virtualização desse tipo de rede.

4.4.1. A arquitetura do OpenFlow

As redes definidas por software (*Software Defined Networks* - SDN) são baseadas na separação dos planos de controle e de dados, de tal forma que o plano de dados se torne programável. A existência de um plano de controle centralizado permite um controle e uma gerência da rede de forma mais simples. De fato, a existência de uma visão central da rede e de uma API de alto nível, ambas providas pelo plano de controle centralizado, facilita o desenvolvimento das aplicações de rede.

A implementação mais conhecida de SDNs é o OpenFlow, que define um protocolo padrão para determinar as ações de encaminhamento de pacotes em dispositivos de rede, como, por exemplo, comutadores, roteadores e pontos de acesso sem fio. As regras e ações instaladas no hardware de rede são responsabilidade de um elemento externo, denominado controlador, que pode ser implementado em um servidor comum. A característica que possibilita este controle diferenciado no OpenFlow é a divisão entre os planos de controle e dados, como pode ser observado na Figura 4.2. Nesta figura, o comutador OpenFlow se comunica através de um canal seguro implementado em software (TCP/-TLS) com o controlador, enquanto que as tabelas de encaminhamento são implementadas em hardware ou software.

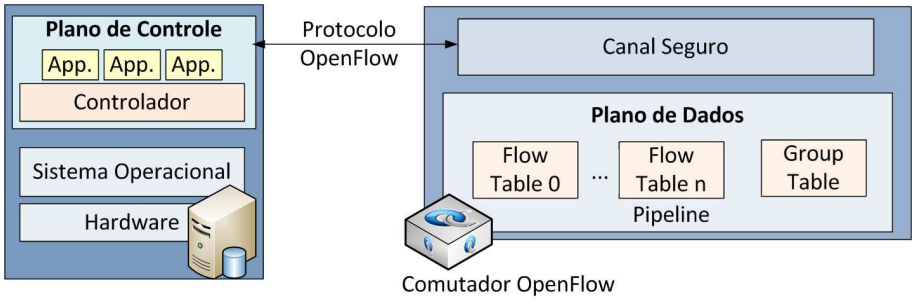


Figura 4.2. Visão geral da arquitetura do OpenFlow.

O plano de dados é formado por um conjunto de tabelas que formam o *pipeline* do OpenFlow. O encaminhamento de pacotes é feito com base em regras simples associadas a cada entrada das tabelas de encaminhamento do comutador de pacotes. As regras de ação para os pacotes são definidas como: encaminhá-lo para uma porta específica do dispositivo, alterar parte de seus cabeçalhos, descartá-lo, ou encaminhá-lo para outra tabela ou para inspeção por um controlador da rede. Em dispositivos dedicados, o plano de dados pode ser implementado em hardware utilizando os elementos comuns a roteadores e switches atuais. Já o módulo de canal seguro permite ao controlador da rede programar as entradas da tabela de encaminhamento com padrões que identifiquem fluxos de interesse e regras associadas a eles. O elemento controlador pode ser um módulo de software

implementado de forma independente em algum ponto da rede [GVV+12]. Um exemplo de uma entrada na tabela de fluxos do OpenFlow pode ser visto na Figura 4.3.

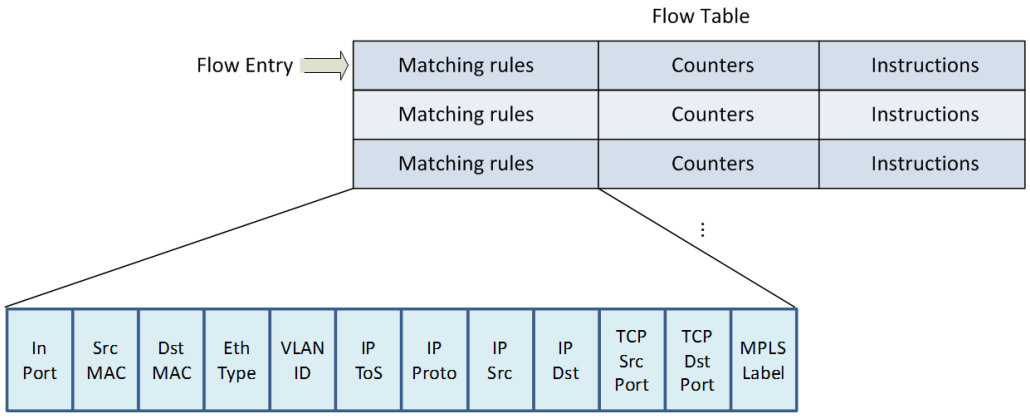


Figura 4.3. Estrutura de uma entrada de tabela de fluxos do OpenFlow, a qual contém uma regra, contadores e instruções.

4.4.1.1. O controlador da rede

O controlador da rede OpenFlow funciona como um sistema operacional de rede, sobre o qual rodam as aplicações definidas pelo usuário para realizar monitoração e encaminhamento de pacotes. O controlador é responsável por manter uma visão topológica da rede atualizada, além de receber e tratar os eventos que ocorrem na rede.

O NOX [GKP+08] é um dos principais controladores para redes OpenFlow. A fim de tratar os eventos da rede, como a chegada de um novo fluxo, o NOX apresenta uma interface de programação da tabela de encaminhamento dos comutadores OpenFlow. A Figura 4.4 mostra um esquema de uma rede OpenFlow com o NOX.

Devemos destacar que há diversas outras soluções de controladores de rede existentes, como o Trema, Maestro, Beacon, Onix etc., cada uma com funcionalidades e desempenho diferentes [GVV+12].

4.4.1.2. Virtualização de uma rede OpenFlow

A virtualização é uma abstração que permite dividir um recurso em diversas partes (também chamadas de *slices*). Tal abstração é usualmente implementada como uma camada de software que provê uma interface virtual muito semelhante à interface real do recurso [FMM+11]. A coexistência de diversos *slices* sobre o mesmo hardware só é possível porque a camada de virtualização quebra a ligação entre os recursos físicos e a camada superior, que, no caso de computadores, é o sistema operacional.

A camada de virtualização de computadores é implementada como um módulo que é executado sobre o hardware, chamado de hipervisor ou monitor de máquinas virtuais. O hipervisor provê uma interface para as máquinas virtuais que é semelhante à

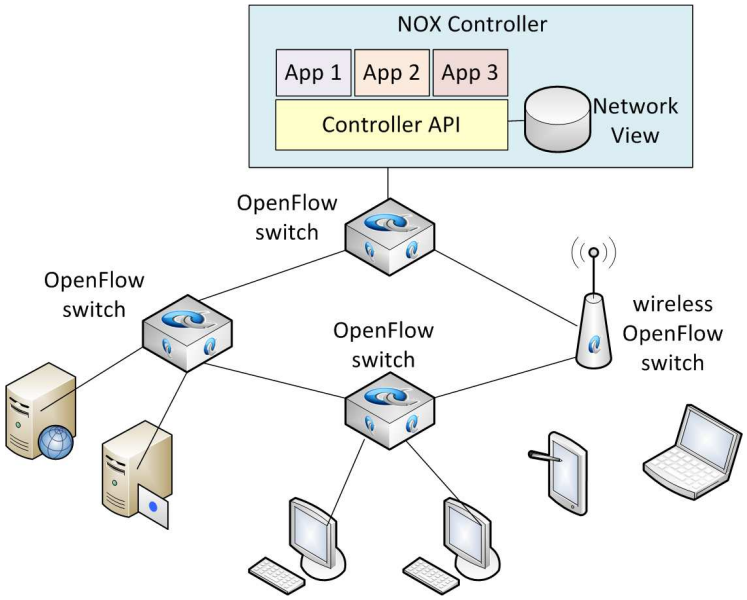
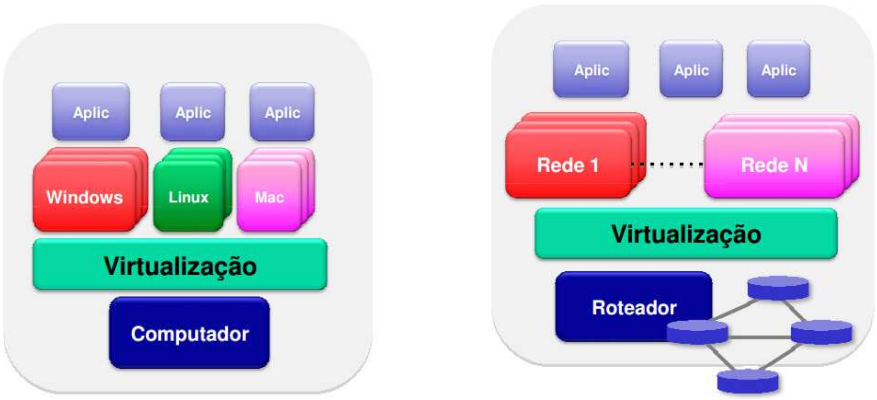


Figura 4.4. Esquema do controlador NOX em uma rede OpenFlow.

interface do hardware. Assim, a máquina virtual acredita estar usando de forma exclusiva a CPU, a memória, o disco e todos os outros elementos de hardware físico, embora esteja compartilhando esses recursos com diversas outras máquinas. Além disso, o hipervisor também é responsável por garantir o isolamento entre as máquinas virtuais, de tal forma que o funcionamento de um ambiente virtual não interfira no funcionamento dos demais ambientes. O isolamento garante que o comportamento de uma máquina virtual será o mesmo independente dos demais ambientes virtuais que coexistem sobre o mesmo hardware. A Figura 4.5(a) mostra um esquema com as camadas envolvidas na virtualização de um computador.

O conceito de virtualização de redes advém do conceito de virtualização de computadores. Ao se criar uma máquina virtual, cria-se um ambiente emulado que simula o hardware. Assim, uma máquina virtual funciona exatamente como uma máquina real, embora possa estar compartilhando o hardware com outras máquinas virtuais. Da mesma forma, a virtualização de rede permite que elementos de rede sejam usados por diferentes redes em paralelo. Assim, é possível a coexistência de diferentes pilhas de protocolo sendo executadas sobre o mesmo hardware. A Figura 4.5(b) mostra uma visão esquemática da virtualização de redes. É possível observar que, assim como na virtualização de computadores, a virtualização de redes também demanda uma camada entre o hardware e o software de rede, com as mesmas funcionalidades de um hipervisor.

Quando a rede possui o controle distribuído, como é o caso da Internet, virtualizar a rede significa compartilhar o plano de dados e criar um plano de controle virtual sobre cada elemento de rede para cada uma das redes virtuais. Nesse caso, o compartilhamento do plano de dados é feito de forma semelhante ao compartilhamento do hardware na virtualização de computadores. Contudo, o OpenFlow apresenta um modelo no qual o plano de controle é centralizado. Nesse caso, a virtualização da rede precisa apenas fazer



(a) Visão esquemática da virtualização de computadores.

(b) Visão esquemática da virtualização de redes.

Figura 4.5. Comparação entre entre os esquemas de virtualização de computadores e de virtualização de redes. Fonte: [MFCD09].

com que o plano de dados em cada elemento de rede seja compartilhado, ou seja, que diferentes planos de controle possam inserir suas regras nesses planos de dados. Assim, planos de controle que são executados em diferentes máquinas, reais ou virtuais, podem interferir no funcionamento da rede.

Contudo, o plano de dados dos comutadores OpenFlow, na versão atual, não permite o uso de mais de um controlador [Ope11]. Para contornar esse problema e permitir a virtualização da rede, foi criado um hipervisor centralizado, chamado de FlowVisor [SCC+10]. O FlowVisor funciona como um *proxy* entre os comutadores OpenFlow e os controladores de cada uma das redes virtuais. Assim, o FlowVisor é o controlador da rede para os comutadores OpenFlow e, para os controladores, o FlowVisor funciona como se fosse o comutador que interliga o controlador com o resto da rede. Assim, o FlowVisor gerencia a virtualização da rede de forma transparente tanto para os comutadores quanto para os controladores. Portanto, o FlowVisor provê uma interface virtual para os planos de controle, de tal forma que os controladores acreditam controlar sozinhos toda a rede.

Como um hipervisor de rede, o FlowVisor também precisa garantir o isolamento entre as redes virtuais. Uma vez que os planos de controle são executados em máquinas diferentes, não existe necessidade de medidas adicionais para garantir o isolamento entre eles. Contudo, o plano de dados é compartilhado, de tal forma que, se mecanismos de isolamento adicionais não forem aplicados, uma rede virtual irá interferir nas demais.

Para garantir o isolamento no acesso ao plano de dados, o FlowVisor apresenta um esquema semelhante ao da Figura 4.6. Nesse esquema, o FlowVisor multiplexa as mensagens enviadas para os comutadores e para os controladores de acordo com a definição do *slice*. Assim, se o controlador envia uma mensagem para configurar o plano de dados de um comutador, o controlador envia, de fato, a mensagem para o FlowVisor. O FlowVisor, por sua vez, verifica se os fluxos afetados pela mensagem do controlador pertencem ao *slice* do controlador. Caso pertençam, o FlowVisor encaminha a mensagem para os comutadores correspondentes. Da mesma forma, quando um comutador envia uma mensagem

para o controlador, notificando algum evento na rede ou enviando dados de monitoração, o FlowVisor intercepta a mensagem para verificar qual(is) controlador(es) deve(m) receber aquela mensagem.

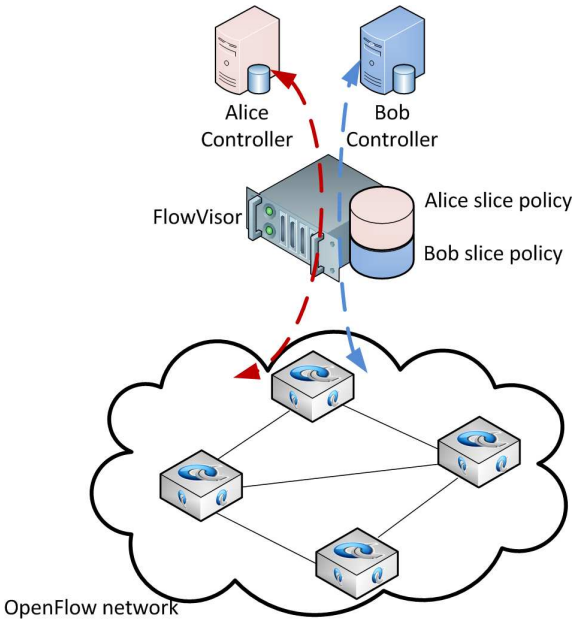


Figura 4.6. Interconexão lógica entre os comutadores OpenFlow e os controladores.

Como mostrado na Figura 4.6, cada rede virtual possui um conjunto de políticas configuradas no FlowVisor. Essas políticas definem o *slice* de cada uma das redes virtuais, ou seja, quais recursos de rede estão atribuídos a cada controlador. No FlowVisor, um *slice* é definido pelas seguintes características:

- **Banda** - Cada *slice* recebe uma banda limitada. Isso garante que cada *slice* terá, pelo menos, uma banda mínima garantida. Essa característica, contudo, só pode ser implementada se o plano de dados der suporte às primitivas de qualidade de serviço.
- **Topologia** - A topologia virtual vista por um controlador pode não incluir todos os comutadores e todos os enlaces da rede. Assim, o FlowVisor precisa garantir que o controlador só receba mensagens a respeito dos comutadores e enlaces reais que participam da rede virtual.
- **Tráfego** - O plano de virtualização precisa determinar qual conjunto de fluxos pertence a cada rede virtual de forma a garantir a consistência da rede.
- **CPU** - A CPU do comutador é compartilhada por todos os planos de controle. Assim, é importante garantir que ela não é sobrecarregada por uma rede virtual, reduzindo o desempenho das demais redes virtuais. Assim, o plano de virtualização deve ser capaz de inferir o uso de CPU e de controlar o compartilhamento da CPU entre as redes.

- **Tabelas de encaminhamento** - Existe um limite físico para o número de entradas na tabela de encaminhamento em um comutador OpenFlow. Assim, o plano de virtualização deve garantir que uma rede virtual não use mais entradas do que um determinado limiar do *slice*.

Entre essas características, destaca-se a que define o tráfego de cada uma das redes virtuais. É importante definir adequadamente o *slice* de cada rede, de forma que dois controladores diferentes não atuem sobre o mesmo fluxo. Isso é importante para garantir que as regras e políticas de encaminhamento de cada rede sejam respeitadas e que o isolamento seja mantido.

4.5. Redes Experimentais para a Internet do Futuro

Existem diversas redes experimentais desenvolvidas para a avaliação de novas arquiteturas e mecanismos para uma nova Internet. Devido à grande praticidade do OpenFlow para a programação da rede e devido à simplicidade para criar ambientes virtuais, essa tecnologia está sendo usada em diversas redes de teste.

Nessa seção, são descritas algumas das redes de teste que utilizam o OpenFlow, dando especial ênfase a como é feita a gerência de identidade nessas redes programáveis. Além disso, são descritos, em linhas gerais, os mecanismos para federação das redes de teste, de tal forma que identidades e recursos locais de cada rede possam ser compartilhados com um conjunto de outras redes. Essa integração promovida pela federação pode ser feita entre redes de teste semelhantes ou distintas, o que implica um grau de complexidade maior. Maiores detalhes serão providos durante a descrição de cada rede de teste.

4.5.1. Conceitos gerais de redes experimentais

Para entender a estrutura geral das redes experimentais para Internet do Futuro, é importante compreender quais são os usuários dessas redes [OFE13b, PRFC10b]:

- **Administrador ou dono** - O administrador é o responsável pela instalação, manutenção e suporte para recursos e processos da sua rede de teste, ou seja, seu domínio administrativo. O administrador também determina políticas de alto nível relativas ao uso da sua rede. Caso exista colaboração entre diferentes redes, essas políticas precisam ser respeitadas ao se disponibilizar os recursos para usuários de outras redes de teste. Dependendo da estrutura e porte, a manutenção e o suporte da rede podem ser atribuídos pelo administrador a um operador.

- **Operador** - Responsável por manter a plataforma funcional, por prover serviços para pesquisadores e por prevenir ataques ou uso indevido das instalações.

- **Pesquisador ou experimentador** - É o usuário que utiliza os recursos físicos e de controle da rede de teste para realizar experimentos.

Além disso, também é preciso definir os elementos básicos que constituem uma rede experimental:

- **Componente** - É o bloco mais básico de uma arquitetura de rede de teste, correspondendo a um computador, roteador, ponto de acesso, entre outros. Um componente encapsula um conjunto de recursos, sejam eles físicos, como CPU, memória, disco ou banda, lógicos, como descritores de arquivos e números de porta, ou sintéticos, como caminhos para encaminhamento de pacotes. Os recursos podem estar em um único dispositivo físico ou distribuídos por um conjunto de dispositivos físicos.
- **Agregado** - É um conjunto de recursos.
- ***Sliver*** - Consiste de um recurso instanciado pelo gerenciador de agregados, ou seja, uma parte de um componente que é alocado para um pesquisador. Esse conceito só é usado nas redes de teste baseadas em virtualização.
- ***Slice*** - Consiste do conjunto de recursos e suas configurações que formam um experimento. Esse conceito só é usado nas redes de teste que podem ser usadas simultaneamente por mais de um pesquisador. Da perspectiva do pesquisador, o *slice* é uma rede de recursos computacionais e de comunicação capaz de executar um experimento. Para o operador, o *slice* é uma abstração primária para contabilização de recursos usados e para a responsabilização sobre a interação com elementos externos à rede de teste.

Com base nesses conceitos, é possível definir os principais módulos para o gerenciamento da rede experimental:

- **Gerenciador de recursos** - É a entidade responsável por gerenciar um recurso, mantendo o registro sobre o estado do recurso, ou seja, se o recurso está alocado ou disponível por um determinado slot de tempo. Caso o recurso esteja em uso, o seu estado pode ser em execução, pausado, parado, entre outros.
- **Gerenciador de agregados** - É a entidade que gerencia diversos agregados ou diversos gerenciadores de recursos. Assim, essa entidade pode gerenciar recursos de um tipo específico, como máquinas virtuais, comutadores, entre outros. Se um agregado contém apenas um componente, também pode ser chamado de gerenciador de componentes. O gerenciador de agregados define quais são as operações disponíveis para os serviços de usuários para gerenciar a alocação de recursos.
- **Âncoras de identidade ou provedores de identidade** - Entidade existente em redes de teste federadas. É responsável por prover autenticação e atributos, de tal forma que as redes de teste possam usar esses atributos para determinar as autorizações para alocação e uso de recursos.
- **Gerenciador de autoridade** - Insere as decisões do administrador na rede de teste, através do estabelecimento das políticas de alocação de agregados para usuários.

4.5.2. Requisitos para a federação de redes experimentais

O objetivo da federação de redes experimentais para a Internet do Futuro é permitir que diversas redes sejam unidas de forma a criar um ambiente de teste de maior escala

e de maior diversidade de equipamentos. Com isso, um pesquisador pode alocar seu experimento utilizando recursos de diferentes redes de teste. Para tanto, é necessário tratar o compartilhamento de recursos, de tal forma que o pesquisador possa verificar quais são os recursos disponíveis em todas as redes de teste, assim como a autenticação e a autorização. Nesse caso, é preciso garantir que a autenticação de um usuário em uma rede de teste sirva como identificação para o uso dos recursos de qualquer outra rede de teste federada, desde que o usuário atenda às políticas locais de cada uma das redes.

Uma vez que cada rede de teste apresenta um arcabouço (*framework*) de controle responsável pelo gerenciamento da rede e pela interface de serviços disponível para o pesquisador instanciar seu experimento, a federação é melhor compreendida quando estudada segundo o ponto de vista do arcabouço de controle [JB10].

De forma geral, os principais desafios da federação de redes experimentais podem ser classificados em três categorias:

- **Gerenciamento de identidade e de autoridade** - Cada rede de teste pode utilizar uma forma de identificação e de autorização diferentes. Se os usuários são identificados de forma diferente, torna-se mais complexa a construção de um sistema de gestão de identidade que permita que os usuários tenham credenciais que sejam válidas na alocação de recursos de qualquer rede de teste. Nesse caso, a autorização torna-se um problema complexo.

- **Procedimentos de controle** - Os procedimentos de controle são utilizados para a descoberta e alocação de recursos. Eles tratam de questões relacionadas ao controle de fluxos e ao uso de diferentes interfaces ou APIs. Os procedimentos de controle de fluxos especificam os mecanismos para descoberta e alocação de recursos, criação de *slices* e gerenciamento de experimentos. Mesmo quando dois *testbeds* adotam os mesmos procedimentos de controle de fluxos, suas APIs podem ser distintas, definindo parâmetros diferentes. Assim sendo, pode não ser possível acessar diretamente recursos em outros *testbeds* sem modificar ou adaptar seus procedimentos de controle e as APIs utilizadas. Portanto, sistemas de federação para *testbeds* heterogêneos devem tratar da tradução dos procedimentos de controle e suas APIs, permitindo uma disseminação e alocação de recursos correta.

- **Descrição dos recursos e do experimento** - Cada rede de teste pode usar um formato diferente para a descrição de recursos e do experimento. Isso se torna um problema ao se combinar diferentes redes de teste. Primeiramente, é preciso que o mecanismo de federação traduza as diferentes descrições de recursos. Essa tarefa pode se tornar complexa, pois os campos utilizados para cada uma das descrições podem ser incompatíveis, de tal forma que pode não ser possível criar uma descrição completa dos recursos segundo a visão de cada uma das redes de teste. Com relação ao experimento, a ausência de um mecanismo de descrição comum a todas as redes traz grande dificuldade aos pesquisadores. De fato, se cada rede de teste possui uma interface de descrição de experimento diferente, um pesquisador precisaria criar diferentes scripts de configuração, um para cada rede de teste envolvida no experimento. Para evitar essa complexidade, que pode inviabilizar a realização do experimento, em especial, se um número grande de redes heterogêneas estiver em uso,

o mecanismo de federação deve prover uma forma de homogeneizar a descrição do experimento.

Todos esses aspectos devem ser levados em conta ao se considerar a construção de ambientes federados. Contudo, nem sempre todos esses desafios são tratados, devido à alta complexidade do problema. De fato, as redes de teste ainda estão em fase de desenvolvimento, de tal forma que novas funcionalidades vêm sendo constantemente adicionadas à medida que se observam as dificuldades para o uso da rede.

4.5.3. Principais redes experimentais baseadas em SDN

A seguir, são descritas algumas das principais iniciativas para construção de redes experimentais que fazem uso de redes definidas por software. São destacados aspectos relativos à gestão de identidade e à federação de redes em cada um dos casos.

4.5.3.1. GENI

A rede GENI (*Global Environment for Network Innovations*) [gen13b] vem sendo desenvolvida nos Estados Unidos desde 2005 e tem como objetivo servir como base aos experimentos de uma nova Internet. É uma iniciativa do NSF CISE (*National Science Foundation - Directorate for Computer and Information Science and Engineering*), apoiada pelas universidades e pela indústria. O GENI é um projeto de longa duração, cujo objetivo é criar uma rede de teste de larga escala através da federação de outras iniciativas como a Internet2 e o National LambdaRail (NLR). Participantes da indústria são principalmente desenvolvedores de hardware, como Juniper, NEC e Cisco, e empresas de telecomunicações, como AT&T.

O desenvolvimento da GENI é feito em espirais, de forma que a rede de teste possa evoluir, disponibilizando novas ferramentas de acordo com os requisitos observados a cada nova versão.

Controle, gerência e federação na GENI

A GENI é uma rede que integra redes de teste de natureza diferente, o que implica diferentes arcabouços de controle e gerência, diferentes tipos de credenciais, diferentes tipos de política de uso, entre outros desafios.

Um dos pontos que merece destaque ao se estudar a GENI é a existência de diferentes arcabouços de controle e gerência. De fato, as redes que integram a GENI apresentam ferramentas próprias de controle, o que gera grande heterogeneidade dentro da rede federada. Dentre os arcabouços usados na GENI, destacam-se:

- *PlanetLab Control Framework* - Usado em todas as redes PlanetLab da GENI, esse arcabouço compreende ferramentas de controle desenvolvidas por diversas universidades.
- *ProtoGENI Control Framework* - Ferramenta desenvolvida para uma integração em larga escala dos sistemas existentes e em construção. Esse arcabouço provê funcionalidades chave para a GENI.

- *ORCA Control Framework* - O arcabouço *Open Resource Control Architecture* (ORCA) foi desenvolvido para atender a uma infraestrutura de computação em rede sob demanda.
- *COntrol and Management Framework* (OMF) - Desenvolvido inicialmente para uso no ORBIT, esse arcabouço é amplamente utilizado nas redes de teste sem fio, embora possa ser utilizado para controlar experimentos com qualquer tipo de dispositivo.
- *DETER Control Framework* - Desenvolvido inicialmente para trabalhar com a rede *cyber-DEfense Technology Experimental Research laboratory* (DETER), é um arcabouço que visa à federação e à integração de modelos de segurança.

Dada essa diversidade de controles e gerência, a federação na GENI também se torna um desafio. Os requisitos gerais para a federação da GENI já foram estabelecidos e, em sua maior parte, se traduzem na estrutura da *GENI Clearinghouse* [JB10,fed13,Fal11].

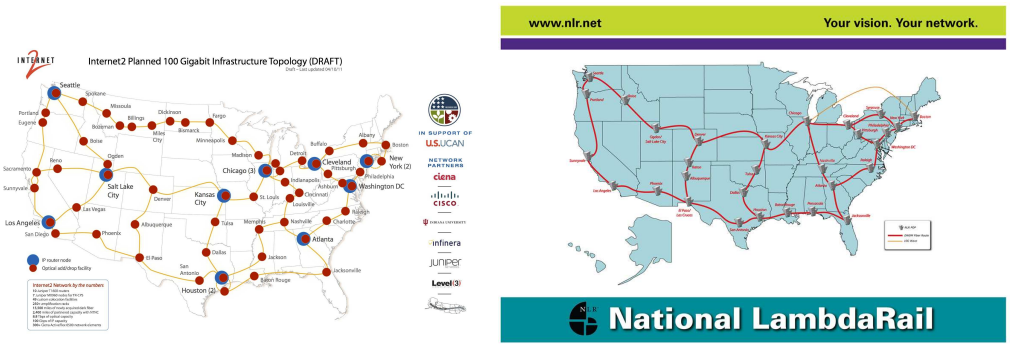
A *GENI Clearinghouse* é uma coleção de serviços relacionados com o objetivo de dar suporte à federação para experimentadores, agregados e o *GENI Meta-Operations Center* (GMOC). Assim, a *clearinghouse* funciona como uma entidade que provê a base da confiança entre todas as demais entidades de software na federação da GENI, de tal forma que qualquer elemento no qual a *clearinghouse* confie deve ser confiado por qualquer outro membro da federação da GENI. Nesse contexto, se a *clearinghouse* confia no elemento, então o elemento possui credenciais certificadas pela *clearinghouse* mostrando os atributos do elemento. Dessa forma, o certificado da *clearinghouse* deve ser aceito como um certificado raiz por todos os elementos da federação.

Entre os principais serviços providos pela *clearinghouse* relacionados às credenciais, destaca-se o *Provedor de Identidade* (*Identity Provider* - IdP), o qual é responsável por gerar certificados e material criptográfico para os experimentadores, tornando-os usuários da federação GENI. Outro serviço importante é a *Autoridade de Projeto*, a qual registra projetos e o papel de cada membro dentro desse projeto. A *clearinghouse* também define uma *Autoridade de Slices*, a qual provê, aos experimentadores, credenciais de *slice*, as quais são usadas na comunicação com os gerenciadores de agregados. O *Serviço de Autorização* tem especial importância por determinar quais ações podem ser realizadas na *clearinghouse* e nos agregados de acordo com as políticas da federação. Outros dois serviços de destaque são o *Serviço de registro*, que provê um catálogo de todos os serviços confiáveis da GENI, e o *Portal Single-Sign-on*, o qual provê uma interface web para autenticação e acesso a serviços da *clearinghouse*. Além desses serviços, a *clearinghouse* é responsável por manter logs dos acessos de forma a permitir a responsabilização dentro da federação.

Backbone da GENI

O *backbone* da GENI é composto basicamente de duas redes: a Internet 2 e a National LambdaRail (NLR), ambas de abrangência nacional nos Estados Unidos, como mostrado na Figura 4.7.

A Internet 2 é uma rede de pacotes óptica, híbrida e dinâmica de 100Gbps, que permite a criação de topologias em camada 2 usando VLANs. A rede da GENI tem

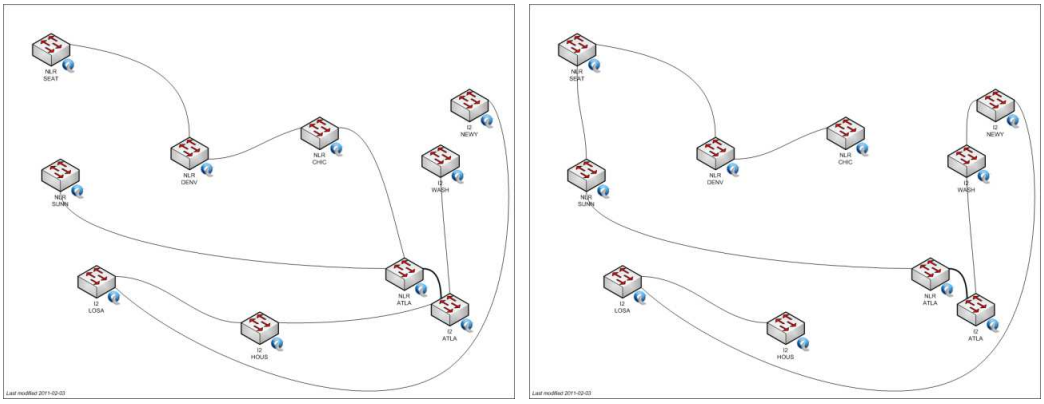


(a) Mapa da Internet 2. (b) Mapa da NLR.

Figura 4.7. Redes que compõem o *backbone* da rede federada GENI. Fonte: [gen13a]

acesso a uma banda dedicada de 1Gbps através da Internet 2. Já a NLR consiste de uma rede de alta velocidade óptica criada para testes composta por mais de 280 universidades e laboratórios. Dentro dessa rede, a GENI tem acesso a uma banda de 30Gbps não dedicada. Assim como na Internet 2, os usuários podem criar sua própria topologia em camada 2 utilizando VLANs.

Dentro desse contexto, cabe destacar que tanto a Internet 2 quanto a NLR possuem comutadores OpenFlow em seu backbone, o que permite a programabilidade da rede. A união dos recursos OpenFlow nessas duas redes forma o chamado *GENI OpenFlow core*, o qual consiste de dez comutadores configurados com duas VLANs. O uso desses recursos é liberado para os usuários da GENI, sem necessidade de notificações às administrações das redes Internet 2 e NLR. Contudo, a integração entre as VLANs do núcleo e as redes universitárias e regionais fica a cargo do usuário. A Figura 4.8 mostra as topologias do núcleo OpenFlow da GENI definidas pelas duas VLANs.



(a) Topologia definida pela VLAN 3715. (b) Topologia definida pela VLAN 3716.

Figura 4.8. Topologia do *GENI OpenFlow core*, o qual inclui nós das redes NLR e Internet 2. (Fonte: [gen13a])

A rede GENI integra, por meio do seu backbone, uma grande quantidade de redes de testes que são classificadas em quatro categorias, sendo elas: redes de *hosts* pro-

gramáveis, redes programáveis, redes sem fio e redes de propósito específico.

Redes de *hosts* programáveis

Nessa categoria, encontram-se dois tipos de redes, sendo elas as redes baseadas em PlanetLab (PlanetLab e GPO Lab myPLC) e as redes baseadas em ProtoGENI (Utah ProtoGENI, Kentucky ProtoGENI, GPO Lab ProtoGENI e *Million Node GENI*).

O PlanetLab apresenta uma arquitetura que cria uma rede de teste composta por computadores localizados nas instituições que integram o consórcio PlanetLab. Nessa arquitetura, toda a rede é formada por computadores localizados nas bordas da Internet. Por utilizar computadores, o PlanetLab provê grande flexibilidade na programação dos nós. Por outro lado, devido à rede de teste ser um *overlay* na Internet, certas características de desempenho de rede não são bem avaliadas nesse tipo de rede de teste.

Os atores que compõem o ambiente do PlanetLab na visão da gestão de identidade são: os pesquisadores das universidades e pesquisadores da indústria. Resumidamente, a facilidade que o PlanetLab fornece é a de se criar um ambiente virtualizado a partir de máquinas virtuais em servidores dedicados. A junção de várias máquinas é a composição de um PLC (PlanetLab Central). Cada PLC é composto por um gerenciador de *slices* (*SM - Slice Manager*), responsável pelo gerenciamento da fatia de tempo dos experimentos, um gerenciador de agregados (*AM - Aggregate Manager*), responsável pela gerência e informação dos recursos disponíveis, uma base de registros (*R - Registry*), onde os usuários estão cadastrados, e um gerenciador de componentes (*CM - Component Manager*), que gerencia os recursos das máquinas. É possível visualizar cada um desses componentes na Figura 4.9 [PRFC10b].

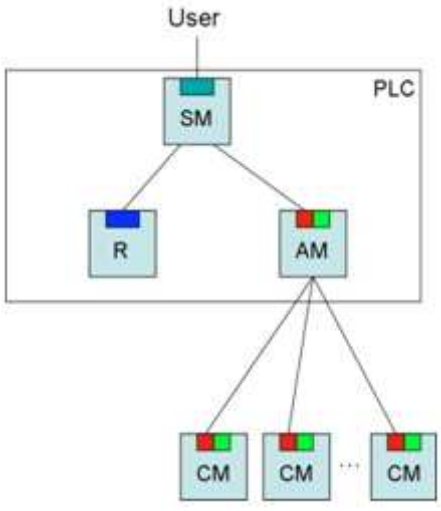


Figura 4.9. Visão esquemática da integração dos elementos do PlanetLab. Adaptado de [sfa13a].

Para a alocação de um experimento, o usuário deverá consultar o gerenciador de *slices*, que verificará em sua base de registros se o usuário tem permissão para o acesso. Uma vez obtendo sucesso nessa consulta, o gerenciador de *slices* é responsável por verifi-

car, junto ao gerenciador de agregados, se há condições disponíveis para os experimentos do usuário. Essa consulta de recursos é realizada pelo gerenciador de agregados nos gerenciadores de componentes e devolvida ao gerenciador de *slices* em forma de resposta. Se a resposta for positiva, o experimento já se encontra alocado e liberado no gerenciador de *slices* e o usuário pode acessar as máquinas de experimentação e realizar seus testes.

Pode-se notar que cada instituição parceira do PlanetLab terá sua base de registros de usuários autorizados a realizarem experimentação. Ou seja, as bases são criadas de forma independente em cada PLC, sendo possível ao gerenciador de *slices* consultar somente as bases que se encontram sob seu domínio. Para garantir acesso ao usuário em cada máquina, o gerenciador de *slices* envia uma liberação contendo a chave pública daquele usuário. Depois de obtida a liberação de acesso, o usuário acessa diretamente cada máquina através de SSH. A administração dos usuários é realizada por meio de uma interface web com HTTPS. Esse ambiente de administração se chama MyPLC [CCR+03].

A federação do PlanetLab ainda está em fase inicial, embora os planos para seu desenvolvimento estejam de acordo com o projeto do *Slice-based Facility Architecture* (SFA) [PRFC10b], descrito em mais detalhes na Seção 4.6.1. A ideia é conseguir fazer a federação não apenas entre redes de teste PlanetLab, mas também com outros tipos de rede de teste, o que está de acordo com o plano de desenvolvimento da GENI. Muitos esforços para a federação também estão sendo feitos através da rede PlanetLab europeia [pla13].

O segundo tipo de redes de *hosts* programáveis da GENI são as redes baseadas em ProtoGENI. O ProtoGENI é uma implementação protótipo de arcabouço de controle para a GENI desenvolvida pela Universidade de Utah. Sendo amplamente baseado no software do Emulab[1], o ProtoGENI é usado, atualmente, no maior conjunto de projetos integrados da GENI.

O suporte do ProtoGENI para federação é baseado no SFA (Seção 4.6.1), através da uma *clearinghouse* simples[2], a qual deve ser compartilhada com todos os membros da federação. A *clearinghouse* é responsável por buscar informações sobre gerenciadores de componentes, usuários, slices, além de prover as informações de segurança para a autenticação e o controle de acesso [pro13a]. A *clearinghouse* permite a confiança entre as instituições federadas se tornando um ponto central para a troca de certificados raiz e de listas de revogação de certificados, conforme mostrado na Figura 4.10.

No modelo de federação do ProtoGENI, as autenticações são tratadas localmente por cada uma das redes de teste. Assim, se um usuário precisa de um certificado para acessar a rede, ele deve requerer esse certificado à instalação local do Emulab. O modelo de confiança é construído de tal forma que todos os usuários precisam de um par de chaves X.509, no qual a chave pública é usada como identificador GENI (GID). Como consequência, os usuários comuns não trocam mensagens com a *clearinghouse*, mas sim com as autoridades de *slice* e os gerenciadores de componentes. As autoridades, então, requisitam suas credenciais à *clearinghouse* para poderem acessá-la.

Embora várias *clearinghouses* possam ser necessárias para controlar redes de teste

[1] http://www.emulab.net/

[2] A *clearinghouse* do ProtoGENI não apresenta todas as funcionalidades recomendadas pela GENI para

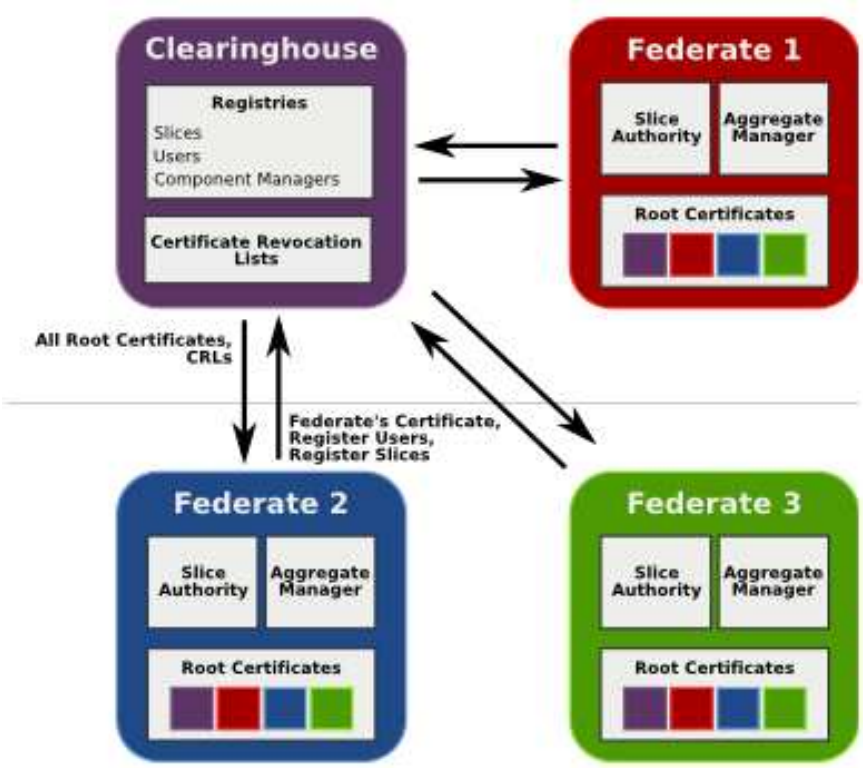


Figura 4.10. Visão esquemática do funcionamento da ProtoGENI *clearinghouse*. Fonte: [pro13b].

federadas, atualmente, não existe suporte para esse tipo de operação no ProtoGENI. A localização da *clearinghouse* é feita de forma fixa pelo certificado, o qual é obtido por um script de federação diretamente da Universidade de Utah.

Uma informação importante sobre gestão de identidade nas redes de *hosts* programáveis é que, com exceção da *Million Node GENI*, todas as demais aceitam as credenciais da GENI, o que facilita a gestão de identidade ao se federar essas redes de teste dentro do contexto da GENI.

Redes programáveis

Esse conjunto de redes de teste é caracterizado pelo uso de nós de rede programáveis. Assim, as redes de teste da GENI classificadas nesse item são formadas por comutadores OpenFlow ou por nós equipados com *General-Purpose Processing Engines* (GPEs) e *Network Processor Blades* (NPB). A ideia aqui é realmente prover um ambiente semelhante ao núcleo de uma rede e que seja programável.

Destacam-se nessa categoria as redes OpenFlow universitárias integradas à GENI, o que inclui as redes de Stanford, Indiana, Washington, Wisconsin, entre outras. Dentre essas, apenas um subconjunto é capaz de aceitar as credenciais da GENI.

esse módulo.

Redes experimentais sem fio

Nessa categoria se destacam duas redes de teste, sendo elas a ORBIT e a DOME. A rede DOME consiste de 35 ônibus equipados com computadores e equipamentos sem fio WiFi e 3G, além de GPSs. A rede ORBIT possui uma escala maior, contendo 400 nós totalmente programáveis dispostos em grade. Esses nós podem executar diferentes imagens, o que inclui o uso de OpenFlow para experimentos de SDN sem fio. Isso possibilita testes como, por exemplo, novos protocolos de roteamento sem fio, protocolos DTN (*Disruption-Tolerant Networking*), Radio Grid etc. Além disso, o ORBIT também apresenta uma pequena rede OpenFlow cabeada constituída por comutadores OpenFlow comerciais e comutadores OpenFlow baseados em NetFPGA[3].

O ORBIT propõe um framework conhecido na literatura como OMF (cOntrol and Management Framework) [SCC12,ROJS10]. O framework genérico de controle, medidas e gerência de redes de teste OMF é uma ferramenta importante para a integração dos ambientes de experimentação. A partir dele, é possível definir um modelo de comunicação padronizado entre *testbeds* distribuídos. Originalmente criado para a gerência do ambiente de experimentação apenas na rede sem fio do projeto ORBIT, ganhou novas funcionalidades para a expansão também a outros tipos de redes de teste. De fato, o OMF apresenta ferramentas genéricas, que podem ser utilizadas para controlar qualquer tipo de equipamento de teste.

A Figura 4.11 exibe a arquitetura do OMF, onde é possível visualizar suas funcionalidades e como a gestão de identidade pode ser analisada. Os passos, seguindo a partir da pesquisadora Alice ilustrada na figura, são compostos pelo envio de uma descrição de experimento (*Experiment Description* - ED), que carrega as informações referentes aos requisitos do experimento solicitado, para o controlador de experimentos (*Experiment Controller* - EC). O controlador de experimentos tem como função intermediar as requisições do usuário junto ao gerenciador de agregados (*Aggregate Manager* - AM). Sendo assim, o controlador de experimentos interpreta a descrição do experimento e solicita ao gerenciador de agregados a alocação dos recursos necessários ao experimento solicitado pelo pesquisador. Em seguida, o gerenciador de agregados faz o contato com os controladores de recurso e com os módulos de monitoração, configurando o experimento.

A monitoração nas redes de teste baseadas no OMF é feita com a *OMF Measurement Library* (OML). A OML é uma biblioteca utilizada para instrumentar o programa executado no experimento. Isso significa que o experimentador deve gerar um programa que executa em cada um dos nós e esse programa deve ter pontos de medição gerados com as funções da OML. Com isso, as medidas de interesse podem ser coletadas periodicamente e automaticamente pelos mecanismos providos na OML. A biblioteca coleta os dados, enviando-os ao servidor de coleta de medidas (*Measurement Collection Server* - MCS), o qual é parte do gerenciador de agregados, podendo realizar um pré-processamento antes do envio [SCC12].

Os usuários do ORBIT são alunos e pesquisadores de todo mundo. Neste ambiente, a liberação de acesso aos nós sem fio é realizada por meio de um administrador, que

[3]http://www.orbit-lab.org/wiki/Documentation/OpenFlow/bSwitchControl

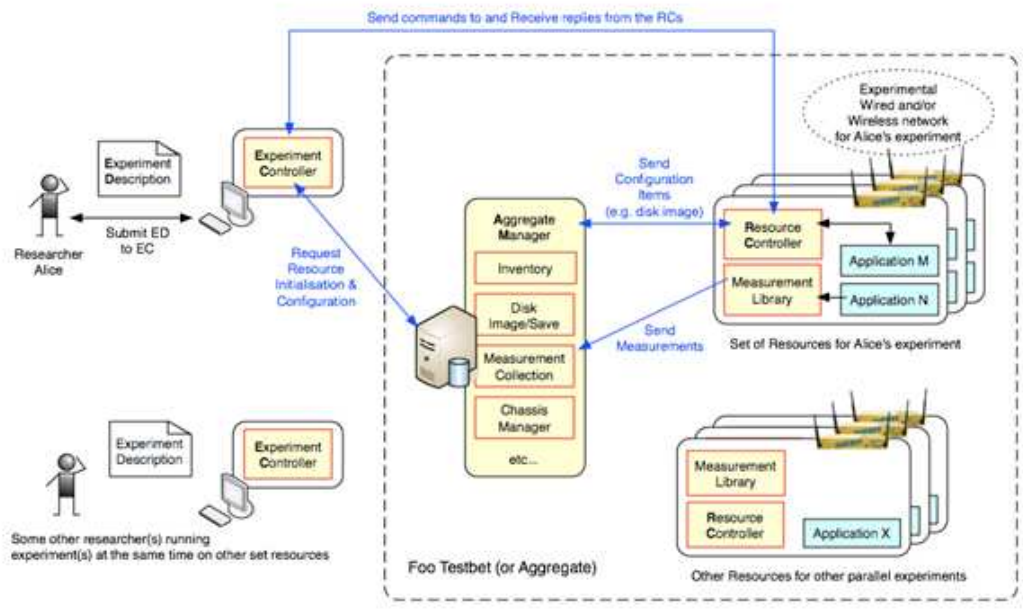


Figura 4.11. Visão esquemática dos módulos que constituem o OMF. Fonte [ROJS10].

cadastra as chaves públicas dos usuários nos nós sem fio para acesso SSH. Sua base de dados é, portanto, local.

A integração de redes de teste OMF já é possível através do uso do *Extensible Messaging and Presence Protocol* (XMPP). A integração do OMF com redes de teste que utilizem outros arcabouços ainda está em fase de desenvolvimento, utilizando-se o SFA, descrito na Seção 4.6.1.

O uso do XMPP está disponível no OMF desde a versão 5.2 do arcabouço de controle e gerenciamento. O XMPP é utilizado para estabelecer a comunicação entre o controlador de experimentos e os nós da rede, através de uma arquitetura *publish/subscribe*. Nesse caso, todas as entidades das redes de teste federadas devem se registrar no servidor XMPP e se inscrever em um conjunto de nós PubSub. Esses nós PubSub têm como objetivo receber inscrições, e, em seguida, sempre que receber uma mensagem, encaminhá-la para todos os nós inscritos. As mensagens trocadas devem seguir o padrão XML (*Extensible Markup Language*). Além de criar os nós PubSub, o XMPP também é usado para funções de segurança, como autenticação e criptografia [SCC12, xmp13].

A importância dos nós PubSub é que o OMF conta com o uso de mensagens unicast e de grupo. Um grupo pode ser definido como um conjunto de nós definidos no script do experimento ou como um `meta` grupo, o qual inclui todos os nós em um experimento específico, sessão ou domínio. Nesse esquema, o domínio identifica a própria rede de testes, que é subdividida em sistema e sessão. A parte de sistema abrange todo o hardware da rede de testes, enquanto que a parte de sessão se refere elementos lógicos, como as sessões de usuários e seus experimentos. Essas entidades podem ser compreendidas melhor se vistas de forma hierárquica, como mostrado na Figura 4.12. No esquema do

XMPP, cada entidade é indicada como um nó PubSub, ou seja, cada domínio, sessão, experimento, nó e grupo são representados como nós PubSub. Nós em um grupo se inscrevem no nó desse grupo, enquanto que todos os nós de um experimento se inscrevem no nó do experimento. Analogamente, cada nó físico se inscreve no seu nó PubSub correspondente. Portanto, todas as trocas de mensagens na rede federada podem ser feitas com intermédio dos nós PubSub.

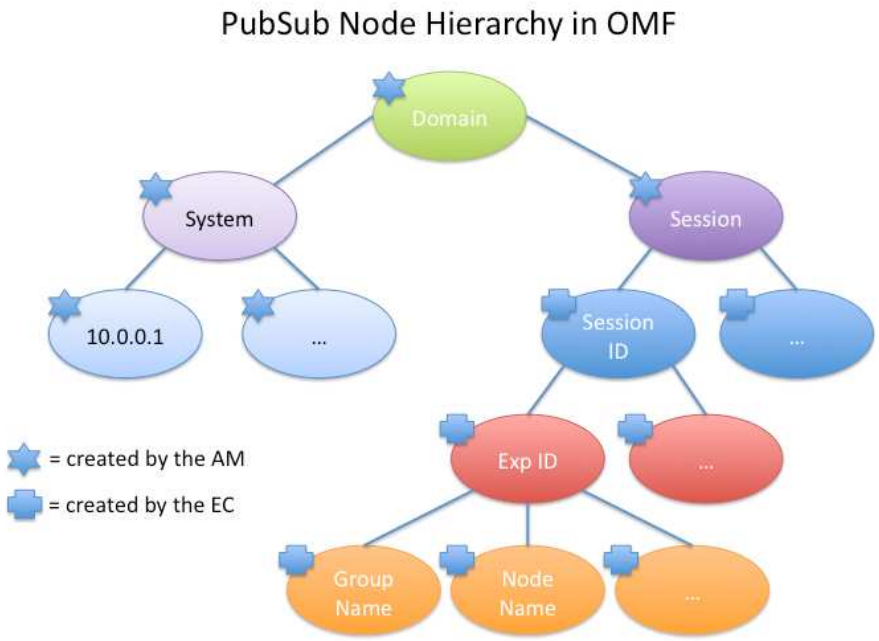


Figura 4.12. Visão hierárquica dos nós PubSub propostos no esquema de federação baseado em XMPP para o OMF. Fonte [xmp13].

A gestão de identidade é feita através de usuários permanentes e usuários temporários. O primeiro usuário, criado durante a instalação, é o `admin`. Além disso, o gerenciador de agregados também cria um usuário `aggmgr` que tem acesso para criar os nós PubSub de domínio e sistema, além de todos os outros sub-nós. Tanto o `admin` quanto o `aggmgr` são usuários permanentes, enquanto que todos os outros são temporários. Os clientes XMPP, ou seja, cada uma das redes de teste representadas pelos seus gerenciadores de agregados, podem registrar e apagar usuários para as suas redes de teste. Assim, após criar uma identificação para o experimento, o controlador de experimentos, com o intermédio do gerenciador de agregados, cria um usuário para o experimento, e, em seguida cria todos os nós PubSub. Ao final do experimento, todos os nós PubSub são removidos. Os controladores de recursos registram o nome do usuário na instanciação do experimento e, quando são desligados, apagam todos os registros de usuários experimentadores.

A Figura 4.13 mostra um diagrama com a troca de mensagens usando o protocolo XMPP para o registro dos recursos e a inicialização do experimento, incluindo o servidor XMPP, o controlador de experimentos e o controlador de recursos.

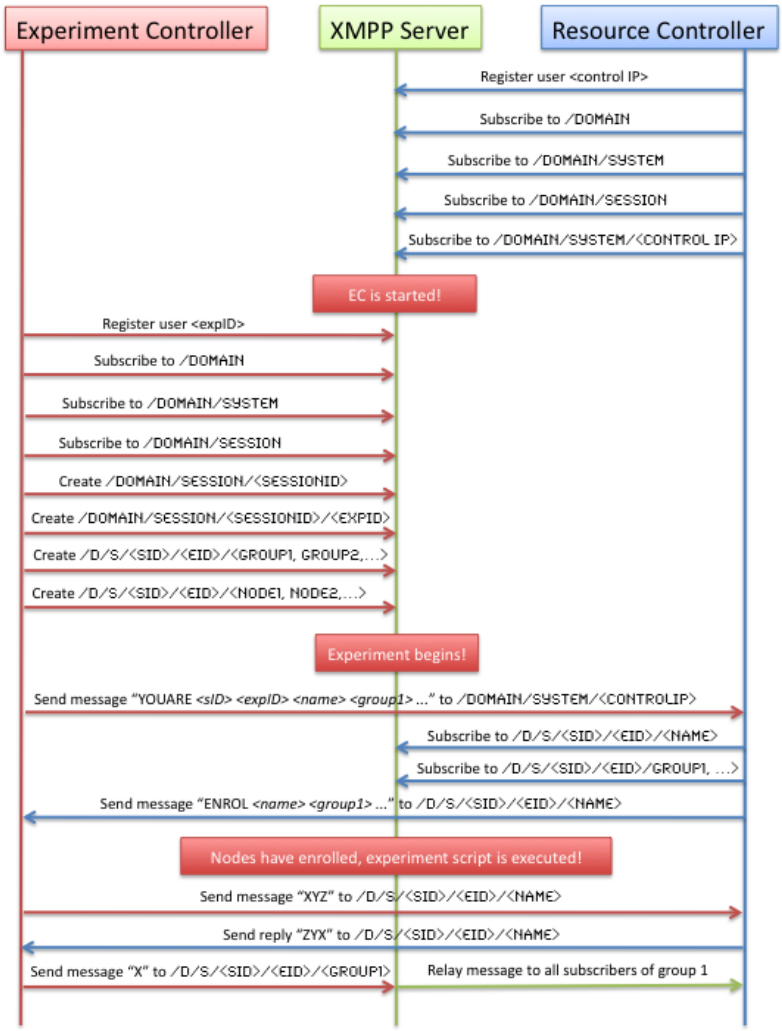


Figura 4.13. Troca de mensagens no protocolo XMPP entre o controlador de experimentos, o servidor XMPP e o controlador de recursos. Fonte [xmp13].

Redes de propósitos específicos

Esse conjunto de redes de teste que fazem parte da GENI é composto pelas redes Deter, Kansei e ViSE. Essas são redes criadas para experimentação de questões específicas da arquitetura da rede. A Deter é uma rede para experimentos com segurança, consistindo de um conjunto de 200 computadores que podem ser interligados usando topologias arbitrárias. Uma vez que esses computadores podem ser configurados de acordo com a necessidade do experimentador, podendo ser, inclusive, caracterizados como roteadores, essa é uma rede considerada como programável. As redes Kansei e ViSE são formadas por sensores, de forma a viabilizar experimentos com esse tipo de equipamento.

É importante destacar que nenhuma das redes nessa categoria, até o momento atual, possui suporte às credenciais da GENI, o que dificulta a federação das mesmas com as demais redes da GENI.

4.5.3.2. FIRE

O FIRE (*Future Internet Research and Experimentation Initative*) [fir13] é para a Europa o que a GENI é para os Estados Unidos, ou seja, é um projeto de Internet do Futuro voltado à criação e experimentação para a nova Internet. O FIRE, quando proposto, já visava o conceito de federação e, assim, existem algumas iniciativas para federação em andamento, como o OneLab [one13], o OpenLab [FG11] e o FEDERICA [SFC+09]. Outro ponto interessante do FIRE é o projeto OFELIA [KW11], o qual se propõe a desenvolver uma rede de teste exclusivamente OpenFlow. Além dessa, se destaca a iniciativa FIBRE [SAM+12], que se propõe a construir uma rede de teste baseada em SDN interligando o Brasil e a Europa. A seguir, é feita uma breve descrição desses projetos.

OFELIA

O OFELIA [KW11] é um projeto direcionado à criação de uma rede de teste OpenFlow. Para tanto, o projeto desenvolve uma rede de teste federada que integra diversas redes de teste, também chamadas de ilhas. Atualmente, o projeto conta com 10 ilhas interligadas.

O controle e a gerência dos experimentos realizados na rede do projeto OFELIA é feito com o arcabouço *OFELIA Control Framework* (OCF) [KW11]. O OCF é responsável por gerenciar os recursos e realizar a comunicação entre as ilhas da rede de teste federada. É importante destacar que o OFELIA trata a federação subdividindo o problema em intra-federação e inter-federação. A interligação de ilhas baseadas em OCF é considerada como intra-federação, enquanto que a integração com outros tipos de arcabouço é feita pela inter-federação [New12]. No caso de inter-federação, o OFELIA utiliza o SFA ou a API da GENI, embora outras interfaces possam ser implementadas e utilizadas em conjunto com o OFELIA.

A interconexão entre as ilhas OFELIA é feita com o uso de túneis VPN entre as ilhas e Ghent. As ilhas que participam da rede GEANT podem usar o enlace de 1 Gbps para criar a conectividade na camada 2. Nas ilhas que não pertencem ao GEANT, a conexão é estabelecida por meio de VPN na Internet [New12].

Um módulo central do OCF é o *Expedient*, o qual é uma interface web que realiza a comunicação entre os usuários e a rede de teste. É no *Expedient* que as funções de autenticação dos usuários e autorização para uso de recursos são realizadas, por meio das funcionalidades dispostas pelo OCF. De fato, o *Expedient* foi construído com base no modelo dos sites de reservas de viagem, que permitem ao viajante reservar voos, hotéis e carros em uma única interface [ofe13a].

A Figura 4.14 mostra uma visão esquemática do funcionamento do *Expedient*. Inicialmente, os usuários se conectam via web ou por sua API de origem (API GENI) para alocar recursos e executar o seu experimento. O *Expedient* recebe as requisições do usuário e realiza a alocação utilizando, caso seja necessário, recursos de diferentes ilhas. Para reservar esses recursos em cada ilha, o *Expedient* apresenta um *plugin* para cada tipo de tecnologia. Ou seja, para o controle de um ambiente OpenFlow padrão, o *Expedient* utiliza o *plugin* OpenFlow, e para a comunicação com o PlanetLab, o *plugin* PlanetLab. Uma vez estabelecida a comunicação com o *testbed*, a alocação de recursos

é intermediada com o gerenciador de agregados da ilha. No caso de uma ilha OpenFlow padrão, há o gerenciador de agregados chamado de *Optin Manager*.

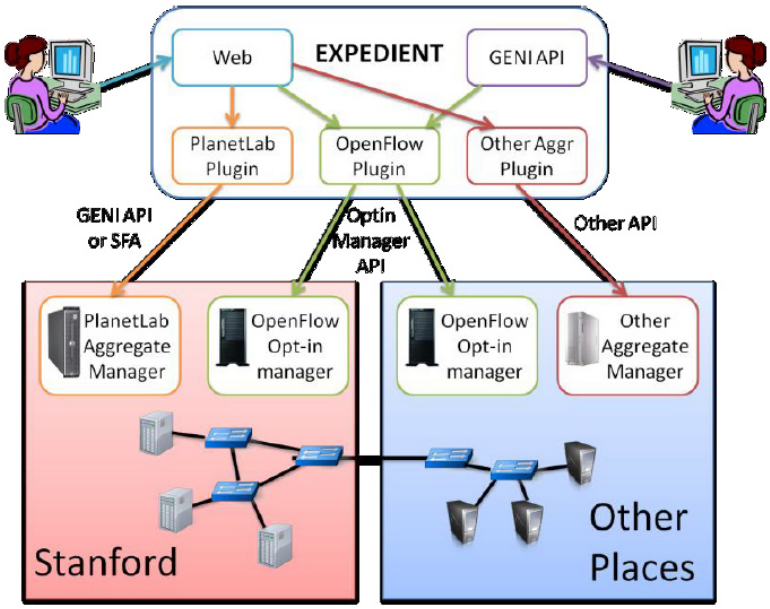


Figura 4.14. Relação entre o *Expedient* e as ilhas da rede de teste federada OFELIA. Fonte: [Sun].

O *Optin Manager* é responsável por receber as mensagens oriundas do *Expedient* por meio de HTTPS e XML-RPC e repassá-las a um controle local. O *Optin Manager* realiza a gestão de identidade, verificando as credenciais dos usuários e as políticas locais da ilha. Uma vez aprovada a requisição do usuário, o pedido do usuário é encaminhado ao FlowVisor para que sejam, por exemplo, criadas as fatias de experimentação para aquele pesquisador. Portanto, o *Optin Manager* permite que os usuários especifiquem os recursos do OpenFlow a serem alocados no seu *slice*.

Outros módulos chave do OFELIA estão representados na Figura 4.15, onde é possível visualizar, além dos componentes já citados, a presença de uma base de diretórios LDAP para a gerência dos usuários e um gerenciador de agregados diretamente responsável pela virtualização de máquinas, realizada com o Xen [LKK+10]. Esse gerenciador de agregados é responsável por gerenciar o uso de máquinas virtuais na rede de teste. Assim, ele cria e descarta máquinas virtuais, além de permitir que os usuários realizem operações administrativas, como pausar e reiniciar, por exemplo.

OneLab e OpenLab

Os projetos OneLab e OpenLab são uma iniciativa para a construção de uma rede de testes de larga escala por meio do uso de federação. Assim, esses projetos disponibilizam meios para interligar diversos projetos, através de um conjunto de ferramentas para experimentação e a federação das redes de teste europeias.

O OneLab e o OpenLab oferecem acesso a diversas redes de teste, sendo as quatro principais: a PlanetLab Europe, que já inclui mais de 200 instituições na Europa; a

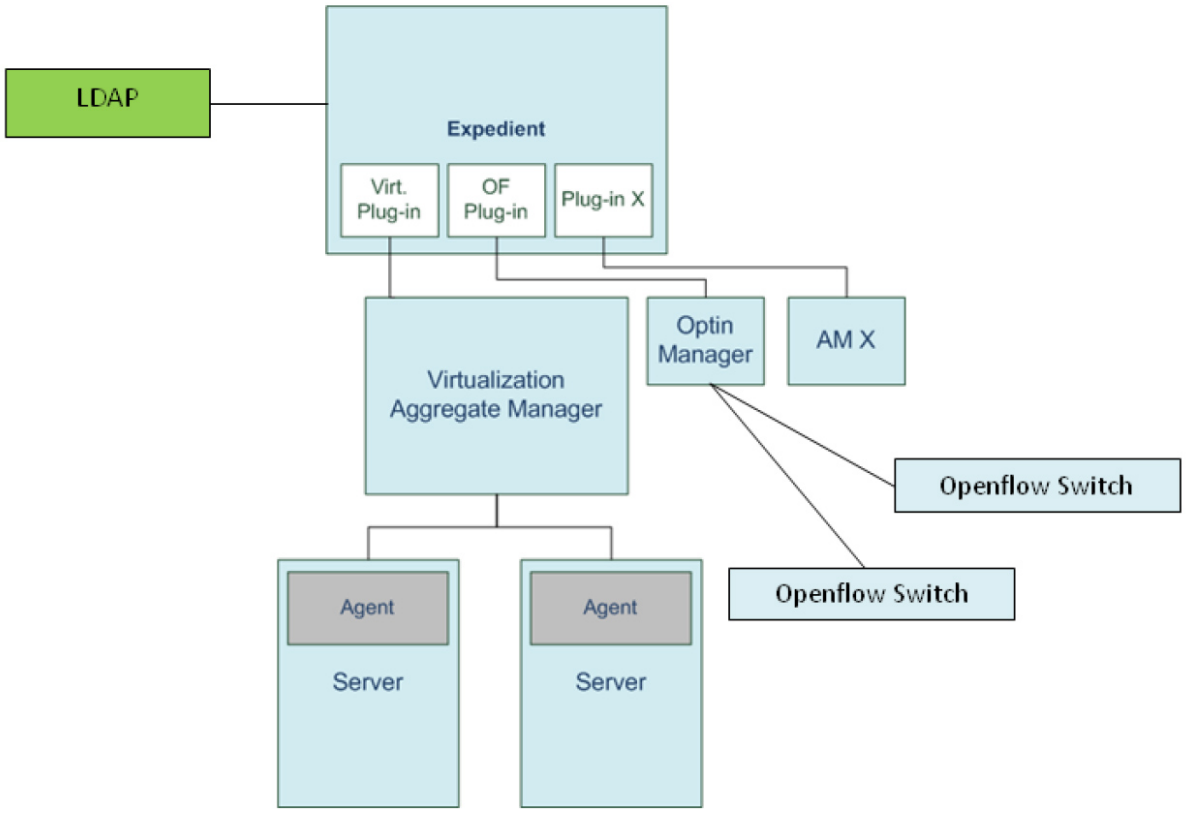


Figura 4.15. Visão geral da arquitetura do OFELIA. Fonte: [Sun].

NITOS, que é uma rede de teste sem fio que utiliza o arcabouço de controle OMF; a ETOMIC, que é uma rede para medidas de alta precisão na rede; e a DIME, que visa medições sobre a topologia e a estrutura da Internet [one13, FG11].

O OneLab tem um foco especial na federação de redes de teste. Atualmente, as técnicas de federação utilizadas entre as redes conectadas ao OneLab são baseadas nas federações do PlanetLab e do OMF. Assim, a plataforma PlanetLab é idêntica à utilizada na GENI, o que facilita, entre outros, a federação entre o PlanetLab Europe (PLE) e o PlanetLab Central (PLC). Já o acesso ao NITOS é realizado por SSH ao nó principal da rede e a partir deste nó é possível acessar os demais, também por SSH.

FEDERICA

O FEDERICA [SFC+09] é baseado na rede Gigabit Ethernet de pesquisa europeia, a GÉANT. Seu intuito é ser uma das principais plataformas para testes experimentais da Europa, e tem como personagens a figura do pesquisador e do administrador do sistema, responsável pelo NOC (*Network Operations Center*). Suas principais características são a criação de máquinas virtuais e VLANs em roteadores lógicos (Juniper).

No que diz respeito à gestão de identidade desse ambiente, o administrador do NOC acessa um portal de administração, que utiliza uma base de usuários própria. Já na

administração/alocação dos *slices*, a figura do administrador gerencia via SSH a liberação dos usuários. Esses usuários, por sua vez, também acessarão as máquinas virtuais disponibilizadas a eles nos nós por SSH ou VNC. Sua base é local e não federada por padrão.

FIBRE

O FIBRE (*Future Internet experimentation between BRazil and Europe*) é uma rede de teste para Internet do Futuro em desenvolvimento que promove uma cooperação entre o Brasil e a Europa. A ideia chave desse projeto é desenvolver ilhas distribuídas geograficamente, conforme mostrado na Figura 4.16 e federá-las. As ilhas que estão em desenvolvimento podem ser divididas em ilhas cabeadas e ilhas sem fio. As ilhas cabeadas utilizam o arcabouço de controle definido pelo projeto OFELIA, ou seja, o OCF. Já as ilhas sem fio utilizam o arcabouço OMF, utilizados nas ilhas NITOS e ORBIT. Além disso, algumas ilhas também são baseadas no arcabouço ProtoGENI.

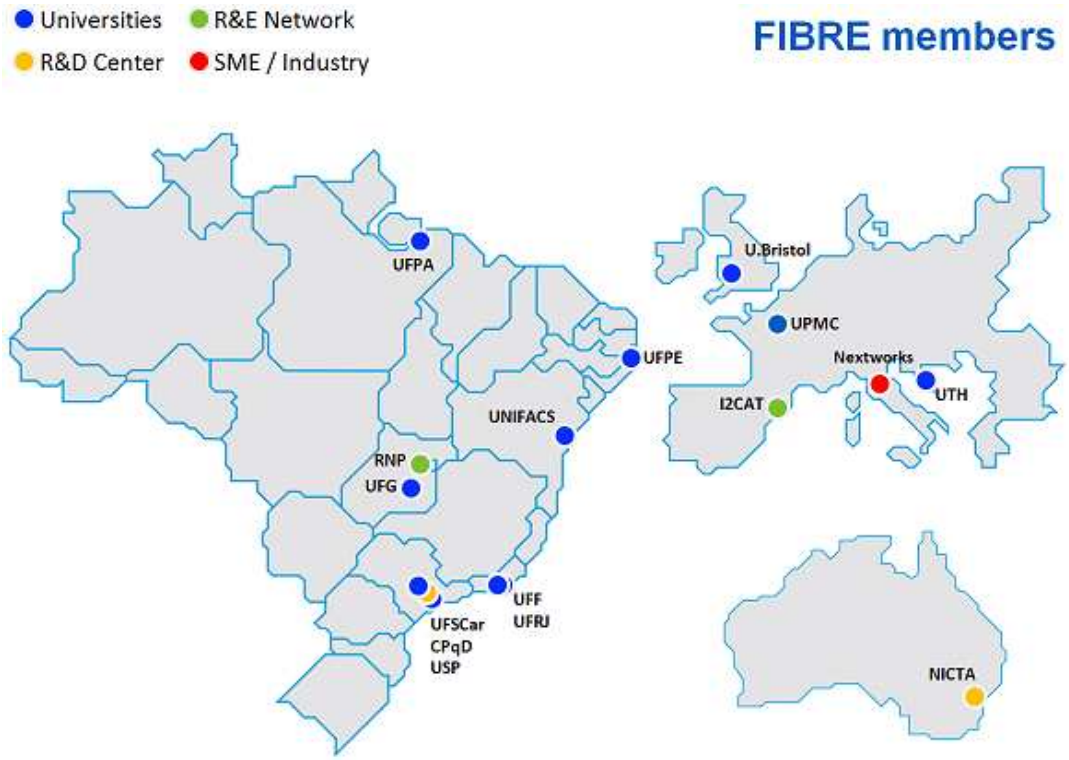


Figura 4.16. Localidade das ilhas federadas pelo FIBRE. Fonte: [fib13].

A escolha desses arcabouços de controle para a construção das novas ilhas se deu devido à necessidade de federação com as redes europeias OFELIA e NITOS, cujas equipes também participam do FIBRE.

Atualmente, a intra-federação das ilhas do FIBRE, ou seja, a interconexão de ilhas com o mesmo arcabouço de controle, já está em fase de implementação. Contudo, a inter-federação ainda está em fase de desenvolvimento e talvez seja concretizada por meio das ferramentas providas pelo SFA.

4.6. Propostas de Gestão de Identidade e Federação em ambientes de experimentação em Internet do Futuro

Nesta seção serão apresentas as propostas de integração entre os diversos *testbeds* e projetos de redes experimentais para a Internet do Futuro existentes. Veremos que a sua maior contribuição diz respeito à troca de mensagens e gerenciamento dos recursos a serem alocados de forma distribuída. Será possível também entender como a gestão de identidade é realizada nestes ambientes e como tais propostas podem colaborar para a sua integração.

Primeiramente, serão abordados os conceitos envolvidos no SFA (*Slice-based Federation Archicteture*) [PRFC10b], descrevendo suas definições padronizadas de um conjunto mínimo de interfaces e tipos de dados, tal que seja possível a interoperação entre *testbeds* distintos. Após a introdução ao SFA será realizada a apresentação de uma proposta que visa facilitar a utilização da federação, a PanLab [GBW+06], onde veremos suas funcionalidades e seus esforços para a experimentação no modelo federado de *testbeds*. Após a apresentação das duas principais propostas de exportação de interfaces e interoperabilidade entre *testbeds*, será apresentado o NOVI (*Networking innovations Over Virtualized Infrastructures*) [LGP+12], discutindo sua proposta ambiciosa de integração de *testbeds* federados e facilidade de experimentação nesse ambiente.

4.6.1. *Slice-based Federation Architecture* (SFA)

O SFA 2.0 (*Slice-based Federation Architecture*) [PRFC10b] é a evolução do SFA 1.0 (*Slice-based Facility Architecture*) [PRFC08], cuja proposta é criar uma arquitetura para a federação entre *testbeds*, inicialmente voltada para ser empregada no PlanetLab, Emulab, VINI [BFH+06b] e GENI mas podendo ser expandida a outros *testbeds*. A evolução da propostas se deu inicialmente por meio conceitual e depois pela experiência na implementação da solução. Sua ideia principal é focada na possibilidade de usuários registrados em um ambiente de experimentação participante da federação poderem usufruir de outros ambientes parceiros, sem a necessidade de um recadastramento. A Tabela 4.1 lista os *drafts* até a versão 2.0 de 2010, assim como a Figura 4.17 mostra sua linha do tempo na evolução das versões.

SFA – Slice-based *Facility/Federation* Architecture	
Facility	
2007, Novembro	SFA Ver. 0.8
2008, Agosto	SFA Ver. 1.01
2009, Abril	SFA Ver. 1.04
2009, Junho	Implementação pelo PlanetLab do SFA Ver. 0.05
2009, Setembro	Implementação pelo PlanetLab do SFA Ver. 0.07
Federation	
2010, Julho	SFA Ver. 2.0

Tabela 4.1. Desenvolvimento do SFA

Os objetivos a que o SFA se propõe a atender são:

- Permitir que os proprietários declarem políticas próprias para alocação e uso de

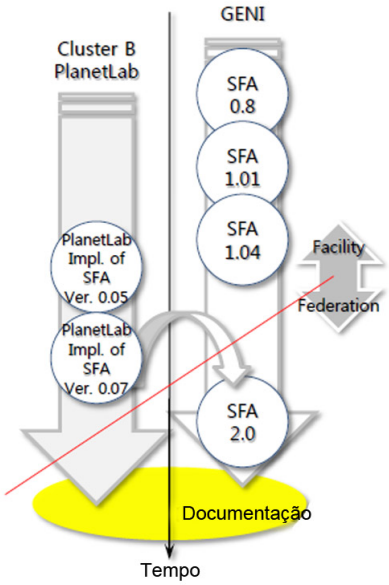


Figura 4.17. Ilustração da evolução das versões do SFA conforme sua documentação.

recursos de infraestruturas sob seu controle, e também que forneçam mecanismos para que essas políticas sejam cumpridas. A rede de testes interconectada será composta por múltiplos proprietários, mas por uma única federação de infraestrutura.

- Permitir ações básicas aos operadores, como instalar e remover *hardware*, instalação e atualização de *software* e monitorar a rede em termos de desempenho, funcionalidade e segurança. O gerenciamento provavelmente é descentralizado, o que faz com que haja mais de uma organização administrando coleções disjuntas de infraestruturas.

- Permitir aos pesquisadores criar, "popular", reservar recursos e executar experimentos em suas fatias (*slices*).

- Permitir que PIs possam identificar o conjunto de pesquisadores de sua organização que possuem permissão para utilizar a infraestrutura.

Como exemplo da utilização do SFA, temos a integração do PLC (PlanetLab Central – EUA) e o PLE (PlanetLab Europe), como pode ser visto na Figura 4.18. Para um entendimento deste exemplo é necessário relembrar a arquitetura do PlanetLab, onde há o *Slice Manager* (SM), *Aggregate Manager* (AM), *Registry* (R) e *Component Manager* (CM). Como visto anteriormente, o SM é responsável pelo recebimento de pedidos de reserva dos recursos (*slices*) e o AM pela alocação real desses recursos nas máquinas. Na Figura 4.18, é possível observar como é feita a integração entre os ambientes distribuídos. A partir da comunicação entre os SM e AM do PLC e do PLE, a alocação de recursos é possível utilizando-se apenas a base de usuários (R) local. Portanto, um dos principais objetivos do SFA é padronizar a comunicação entre os ambientes distribuídos e independentes por meio de trocas de mensagens e comunicação direta entre os diferentes

SM e AM, verificando os recursos disponíveis e os alocando conforme a política local pertinente a cada *testbed*.

O SFA prevê *testbeds* no mesmo nível hierárquico, ou seja, de forma que sejam tratados como em uma rede P2P (*peer-to-peer*), onde a comunicação entre os ambientes pode ser negociada diretamente sem o papel de um ente central para controle de todo o ambiente. Porém, em [PRFC08] e [PRFC10b], temos a supervisão de uma autoridade de maior nível hierárquico sobre outros *testbeds*, onde esse funciona como um servidor *top-level* nas questões relativas a reserva de recursos (*slices*) e gerência de autoridade. Isso quer dizer que, apesar da proposta P2P de comunicação entre os integrantes da federação, algumas funcionalidades são dependentes do *top-level* de cada consórcio, ou de forma mais geral, para o PlanetLab, dependentes do PLC.

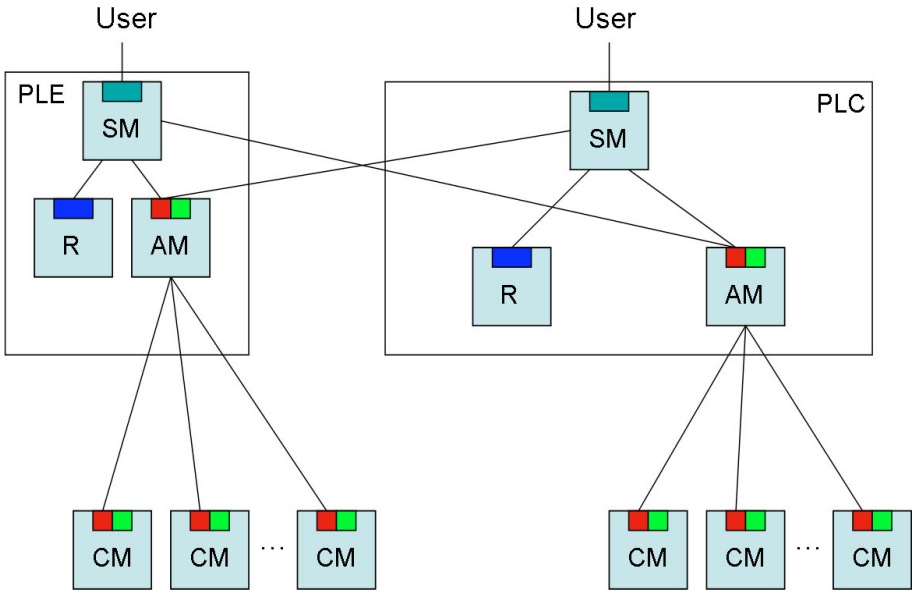


Figura 4.18. Arquitetura de exemplo de utilização do SFA entre o PLC e o PLE. Fonte: [PRFC10b]

O uso do *testbed* federado utilizando SFA se inicia com uma requisição para alocação de um *slice*. Então, o usuário envia sua requisição ao *Slice Manager* e este verifica e solicita os recursos disponíveis nos *Aggregate Managers*. Esses passos são feitos por meio de mensagens do tipo RSpec (*Resource Specification*), um formato de mensagem baseado na linguagem XML. Este formato de mensagem facilita a legibilidade do usuário e carrega consigo a descrição do que o usuário deseja requisitar, como por exemplo, quantos computadores, qual a largura de banda desejada, a localização dos *testbeds* onde recursos devem ser alocados etc. Um exemplo de um arquivo RSpec do SFA pode ser visualizado na Listagem 4.1. Nessa listagem, recursos são requisitados em dois *testbeds* distintos, chamados de “ppk” (linha 3) e “plc” (linha 12), onde, respectivamente, no primeiro *testbed* é alocado apenas um nó e no segundo, são alocados dois nós. É possível observar também que nos dois últimos nós requisitados, especificações de restrição de banda são enviadas, como vemos nas linhas 17 e 22.

Listagem 4.1. Exemplo de um arquivo RSpec.

```
<?xml version='1.0' encoding='ASCII'?>
  <RSpec type="SFA">
    <network name="ppk">
      <site id="s3">
        <name>kaist</name>
        <node id="n1">
          <hostname>kaistnode1.planet-lab.kr</hostname>
        </node>
      </site>
    </network>

    <network name="plc" slice="inria_omftest">
      <sliver_defaults/>
        <site id="s13">
          <name>Princeton</name>
          <node id="n10025">
            <bw_limit units="kbps">10000</bw_limit>
          <sliver/>
          </node>
          <node id="n10328">
            <hostname>planetlab-9.cs.princeton.edu</
               hostname>
              <bw_limit units="kbps">10000</bw_limit>
          </node>
          <node id="n10845">
            <hostname>delta.cs.princeton.edu</hostname>
          <sliver/>
          </node>
        </site>
    </network>
  </RSpec>
```

Já para a efetivação da alocação dos recursos, e envio de comandos diretamente às interfaces exportadas dos componentes, são utilizadas mensagens do tipo XML-RPC [Win99]. Esse tipo de chamada de procedimento é escrito em formato XML e tem como base o protocolo HTTP. Para a autenticação da chamada de procedimento, é utilizado o SSL (*Secure Sockets Layer*) em conjunto ao HTTP (HTTPS - *Hypertext Transfer Protocol Secure*), garantindo o transporte seguro das mensagens.

Quando um recurso é solicitado, é utilizada uma política de alocação de recursos. No SFA, essa política é simples, baseando-se no melhor esforço. Ou seja, como cada *test-bed* tem a possibilidade de livre oferta de seus recursos, não há uma política homogênea de recursos que garanta que tudo o que foi solicitado será concedido. Sendo assim, são selecionadas apenas fatias que se enquadrem nas especificações solicitadas pelo usuário e garantam que atendem à sua demanda.

4.6.2. Identificador Global

Para a gestão de recursos, o SFA utiliza um identificador global (GID - *Global Identifier*) para nomear seus componentes (nós, *slices* etc.), sendo este o identificador único e global referenciado sempre que for necessária a comunicação nos *testbeds* da federação. O GID deve ser, portanto, seguro e passível de confirmação de sua veracidade. Podemos dizer que o GID é um certificado composto por três elementos.

```
GID = (PublicKey, UUID, Lifetime)
```

Onde:

- `PublicKey`: chave pública correspondente à chave privada do objeto em questão.
- `UUID` (*Universally Unique Identifier*): é o identificador único universal utilizado como identificação numérica.
- `Lifetime`: é o tempo de vida do GID.

A chave pública (*PublicKey*) corresponde ao objeto identificado pelo GID, formando assim a base para a autenticação. Outra variável presente na tupla é um identificador universalmente único (UUID - *Universally Unique IDentifier*) [ISO04] que deve ser imutável e absoluto. O objetivo de usar o UUID é garantir que, mesmo que a chave pública mude, o objeto possa ser identificado dentro da federação. A última variável é o tempo de vida do GID (*Lifetime*), que descreve sua validade (destacando que os GIDs dos objetos podem ser "renovados", atualizando seu tempo de vida). Uma autoridade também é identificada por seu próprio GID, e, portanto, passível de verificação por uma estrutura de chaves públicas hierárquica. Ou seja, objetos subordinados na hierarquia de chaves públicas de outros objetos podem ser validados de forma intermediária, sem consulta à Autoridade Certificadora raiz.

Todo GID é associado a um registro, que é a identificação daquele objeto em formato HRN (*Human-Readable Name*). Esta informação ainda pode incluir dados complementares, como a URI (*Uniform Resource Identifier*) do gerenciador de objetos, o endereço IP do nó ao qual o objeto pertence, entre outras informações. Como ilustração deste mapeamento hierárquico, podemos criar uma analogia entre o HRN e o nome em um serviço de DNS (*Domain Name System*) para identificar a quem e em qual hierarquia o objeto em questão pertence. A hierarquia em questão deve assumir uma autoridade de nível mais alto que seja confiável para todas as entidades, resultando em nomes padronizados pelo seguinte formato:

```
top-level_authority.sub_authority.sub_authority.name
```

Sendo assim, podemos exemplificar a utilização de um HRN para um objeto através da entrada `plc.princeton.ndt`, onde temos a ferramenta NDT (*Network Diagnostic Tool*) sendo executada em um nó hierarquicamente alocado em Princeton, que por sua vez pertence à autoridade de maior nível hierárquico (*top-level*) PLC.

4.6.2.1. Registro

Um registro (*registry record*) guarda informações sobre objetos em um sistema (por exemplo, componentes e fatias de experimentação) e sobre entidades, (por exemplo,

usuários, *Management Authorities*, *Slice Authorities*). Os *registry records* são definidos de acordo com o seguinte formato:

```
Record = (HRN, GID, Type, Info).
```

Onde *Type* pode ser *Management Authorities*, *Slice Authorities*, *Component*, *Slice* ou *User*. De acordo com o valor de *Type*, Info pode receber diferentes conteúdos. Destacamos como exemplo o User:

Para `Type = User`,

Temos `Info = (PostalAddr, Phone, Email, AuthTokens[]).`

Além dos campos intuitivos para *PostalAddr, Phone, Email*, temos o campo *AuthTokens* que armazena os tokens de autenticação do usuário em questão para acesso a seus recursos (*slivers*) associados. Diferentes tipos de componentes têm diferentes tipos métodos de acesso, por exemplo, para um *host*, o acesso utilizado é o *ssh*, e sendo assim, neste campo estaria armazenada sua chave para acesso.

É importante destacar que o registro armazena essa informação relativa ao *AuthTokens*, mas não é ele o responsável por distribuir, atualizar e remover os tokens. Em geral, as informações disponíveis em um registro são relativamente estáticas. Para informações mais detalhadas e dinâmicas sobre um objeto é necessário contatar quem o controla ou responde por ele.

4.6.2.2. Ticket

Um ticket é criado para vincular recursos a um ator (por exemplo, usuário, pesquisador). Essa liberação é feita com o componente assinando uma RSpec para produzir esse Ticket. Sendo assim, tickets emitidos por um componente, posteriormente, podem ser resgatados para a aquisição de seus recursos.

O formato de um ticket é:

```
Ticket = (RSpec, GID, SeqNum)
```

Onde, o número de sequência (SeqNum) garante que o ticket seja único.

Para instanciar o slice, é necessário obter um ticket para cada um dos recursos a serem alocados, o que é feito com a seguinte operação:

```
Ticket = GetTicket(Credential, RSpec)
```

A obtenção ou não do ticket depende, entre outros fatores, da disponibilidade do recurso e da política de alocação. Após obter o ticket, é possível alocar os slivers usando o respectivo ticket, o que é feito com a operação:

```
RedeemTicket(Ticket)
```

4.6.3. Credenciais

Uma credencial carrega os direitos emitidos para uma entidade em particular. Por exemplo, um usuário pode receber uma credencial que o permita instanciar uma fatia de experimentação em um conjunto de componentes por um determinado período de tempo.

Uma credencial é descrita pela seguinte tupla:

`Credential = (CallerGID, ObjectGID, ObjectHRN, Expires, Privileges, Delegate)`

Onde `CallerGID` identifica a entidade para a qual a credencial foi emitida; `ObjectGID` e `ObjectHRN` identificam o objeto para o qual a credencial se aplica; `Expires` informa por quanto tempo a credencial é válida, `Privileges` identifica a classe de operações que o detentor da credencial tem permissão para invocar; e `Delegate` informa se o detentor da credencial pode delegar a credencial para outra entidade.

O SFA padrão utiliza as credenciais para realizar o controle de acesso dos recursos dos *testbeds*. Contudo, o SFA também pode ser usado com o *Attribute-Based Access Control* (ABAC), descrito na Seção 4.2. O modelo de controle de acesso do ABAC define que as autorizações devem ser providas com base nos atributos de um usuário. No SFA, as *Management Authorities* e as *Slice Authorities* podem definir atributos para uso de seus recursos que correspondem aos privilégios definidos no SFA, descritos na Seção 4.6.3.1 [PRFC10a].

4.6.3.1. Controle de acesso e Autorização (Privilégio)

Cada privilégio (`Privilege`) implica o direito de invocar uma certa quantidade de operações em uma ou mais interfaces da SFA. A Tabela 4.2 resume a relação entre privilégios, interfaces e operações.

Privilégio	Interface	Operações
`authority`	Registry	todas
`refresh`	Registry	Remove, Update
`resolve`	Registry	Resolve, List, GetCredential
`instantiate`	Slice	GetTicket, CreateSlice, DeleteSlice, UpdateSlice
`bind`	Slice	GetTicket, LoanResources
`control`	Slice	UpdateSlice, StopSlice, StartSlice, DeleteSlice
`info`	Slice	ListSlices, ListComponents, GetSliceResources, GetSliceBySignature
`operator`	Management	todas

Tabela 4.2. Permissões (privilégios) conforme perfil do ator.

As interfaces citadas na Tabela 4.2 são definidas apenas em alto nível pela especificação da SFA e devem ser exportadas pelas entidades para possibilitar comunicação externa.

De acordo com as especificações do SFA, direitos relacionados aos *slices* irão se originar das *Slice Authorities* (SAs). Serão elas quem aprovarão ou serão responsáveis por fatias associadas a usuários. Sendo assim, uma SA tem o privilégio *authority* para associar usuários e fatias. Já os direitos relacionados a recursos de componentes são subordinados ao *Management Authority* (MA), que define suas políticas próprias de alocação de recursos e associa usuários a esses componentes. Sendo assim, um MA tem o privilégio *authority* para associar usuários e componentes.

Já sobre o privilégio *refresh*, usuários, componentes e autoridades possuem autoridade sobre suas respectivas atribuições. Por exemplo, usuários possuem o privilégio *refresh* para as fatias de experimentação a que estejam afiliados.

Cada componente implementa uma política de alocação de recursos que determina quantos recursos são atribuídos a cada fatia de experimentação. Um usuário, que recebe os privilégios *instantiate* ou *bind* para uma determinada fatia, é visto como tendo o direito de solicitar recursos a um componente. No entanto, depende de cada componente a decisão de hospedar ou não uma fatia e de quanto recurso alocar a ela.

4.6.3.2. Ferramentas de auxílio ao usuário

Para a utilização do SFA entre *testbeds*, estes devem permitir a utilização de comandos contidos em um script Python, chamado "sfi.py", que é responsável pela maioria dos controles citados até o momento. São comandos aceitos pelo "sfi.py": *list, show, add, remove, slices, resources, create, update, delete* [sfa13b]. Ou seja, a partir deste *script*, é possível realizar a integração dos *testbeds* e executar comandos nos diversos ambientes distribuídos.

Para facilitar a gerência e visualização dos conteúdos mencionados, há uma ferramenta ilustrada na Figura 4.19, onde o administrador e usuário podem acompanhar as ações referentes aos recursos. Essa ferramenta se chama Sface e foi criada no âmbito do projeto OneLab [one13]. Sface ajuda o usuário a visualizar o RSpec de um dado *slice*, editá-lo e submetê-lo novamente ao sistema (através do botão *Update Slice Data*).

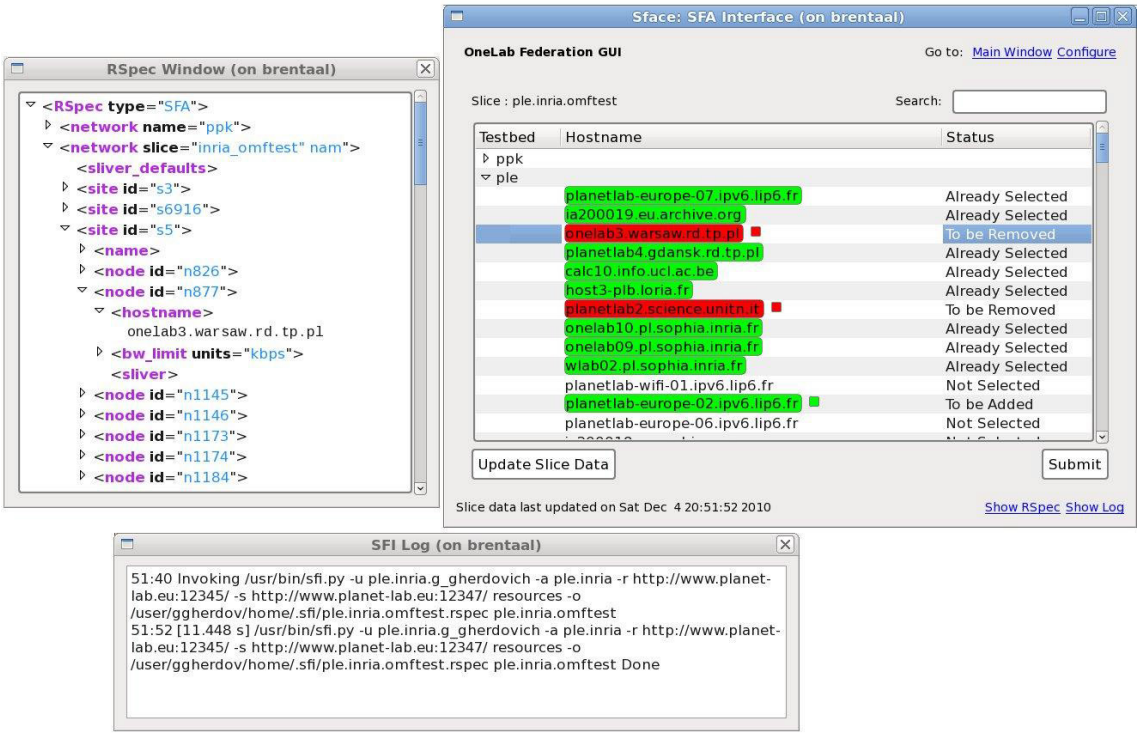


Figura 4.19. Interface da ferramenta Sface. Fonte: [one13]

4.6.4. *Panlab Federation* (Teagle)

O *Pan-European Laboratory for Next Generation Networks and Services* (Panlab) [GBW+06] e seu sucessor, o *Panlab Infrastructure Implementation* (PII), visam a

expansão da facilidade de experimentação em ambientes distribuídos, provendo uma infraestrutura federada. Panlab é um projeto europeu, que começou em 2006 o desenvolvimento de indicativos referentes a mecanismos e procedimentos voltados à regras de abstração de alto nível que possibilitassem a federação de *testbeds* de Internet do Futuro, sendo o PII responsável pelo framework que implementa essa funcionalidade.

O Panlab é formado por três atores, o *Panlab Customer*, *Partner* e *Office*. Descritos a seguir:

- *Panlab Office*: é o coordenador, aquele que gerencia e provisiona os recursos disponíveis na infraestrutura global do Panlab.
- *Panlab Customer*: é a figura do usuário da infraestrutura, por exemplo, um pesquisador.
- *Panlab Partner*: é aquele que provê a infraestrutura de experimentação, ou seja, os recursos e equipamentos.

Como podemos observar na Figura 4.20, sua estrutura é hierárquica, onde o pesquisador deve sempre entrar em contato com o *Office* para requisitar seus recursos de experimentação. Todos os *testbeds* que se integrem a este ambiente devem estar abaixo do *Office*, de forma que seja possível a sua provisão e gerenciamento. Vemos na mesma figura a participação do *Partner*, no que diz respeito à gerência e manipulação dos recursos de cada *testbed*.

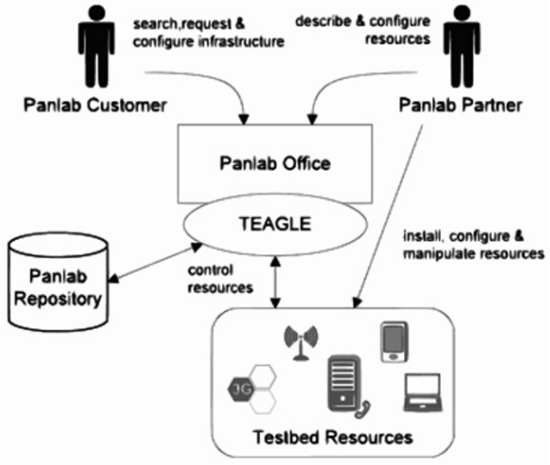


Figura 4.20. Hierarquia do PanLab. Fonte: [GBW+06]

Na Figura 4.20, também é possível ver, abaixo do *Office*, o Teagle. O Teagle é uma ferramenta de automatização para a gerência tanto dos recursos quanto dos experimentos em si. É o Teagle quem auxilia o pesquisador e o administrador dos *testbeds* na comunicação com os recursos em si.

Para ilustrar melhor o Teagle, tem-se a Figura 4.21, que representa seu diagrama com cada um de seus componentes. Partindo do experimentador (pesquisador), à esquerda

da figura, podemos verificar como são realizados os passos para o pedido, alocação e coleta de resultados no Teagle. Este experimentador envia por meio do Portal, que é um aplicativo Java Web Start capaz de realizar a comunicação com os demais componentes do sistema, por exemplo, um pedido de alocação de recursos. O Portal é também chamado, quando acessado pelo experimentador, de *VCT Tool*. O Teagle, ao receber a requisição, armazena todo o registro em um repositório e passa à *Policy Engine*, responsável por realizar a autenticação e autorização do pesquisador em questão. Após realizada a verificação, os recursos podem ser solicitados ao *Orchestration Engine*, que é o componente responsável pela verificação e alocação dos recursos. Toda a comunicação com os nós em si é realizada pelo *Teagle Gateway*. Esse componente se comunica diretamente com o PTM (*Panlab Testbed Manager*), que é o AM para o Panlab, e esse por sua vez envia as mensagens, padronizadas em XML-RPC ao nó (equipamento) através de seu RA (*Resource Adapter*), através do que é chamado de ponto de referência T_1 [WHC+10].

A interface T_1 é usada pelos gerenciador de domínio para exportar operações CRUD (*create, read, update, delete*), dando ao Teagle o que é chamado de ponto de referência para os recursos. É possível visualizar o XML referente utilizado na configuração dos dados em http://www.fire-teagle.org/T1.

O mapeamento dos comandos CRUD para os recursos reais é implementado pelo gerenciador de domínio. É utilizado o conceito de adaptadores de recursos para traduzir um comando T_1 genérico (e o XML de configuração associado) em algo que o recurso possa compreender (por exemplo, SNMP, SSH + scripts, etc.). A configuração é definida baseando-se no modelo de informação comum. Sendo assim, todos os recursos controlados pelo sistema necessitam estar descritos nos termos deste modelo. Embora esse modelo introduza uma sobrecarga inicial para a publicação e registro dos recursos, ele permite uma gestão de recursos mais refinada.

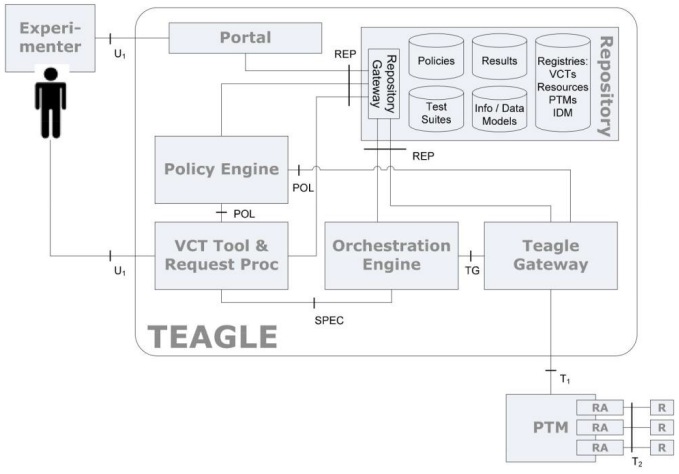


Figura 4.21. Arquitetura da ferramenta Teagle. Fonte: [GBW+06]

O Teagle é o responsável pela gestão de identidade interna aos *testbeds* que utilizam a infraestrutura do Panlab com PII. Os usuários são verificados por meio de papéis associados e delegadas as devidas permissões. Seus papéis podem variar entre:

- *Panlab Customer*: que é usuário (experimentador) do testbed.

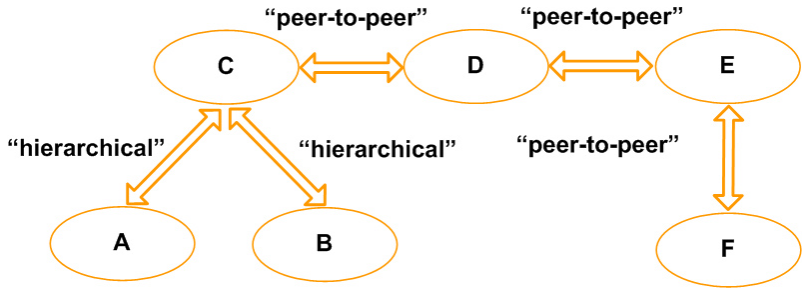


Figura 4.22. Generalização do NOVI para comunicação P2P e hierárquica. Fonte: [ABPO11]

- *Panlab Partner*: que tem o poder de administrar recursos específicos em seu testbed.
- *Teagle Administrator*: que é responsável por administrar os dados do Teagle em si e alterar diretivas.

4.6.5. *Networking innovations Over Virtualized Infrastructures* (NOVI)

O projeto NOVI (*Networking Innovations Over Virtualized Infrastructures*) [LGP+12] explora abordagens eficientes para compor infraestruturas para a Internet do Futuro. Os recursos pertencentes a vários níveis, ou seja, redes, armazenamento e processamento podem, em princípio, ser geridos por ambientes distintos, heterogêneos e independentes. O NOVI tem, portanto, a intenção de desenvolver e validar métodos, sistemas de informações e algoritmos que forneçam aos usuários, a partir de suas fatias isoladas, uma gama de recursos e serviços provenientes de infraestruturas federadas. NOVI faz parte do projeto FIRE para facilitar os modelos e métodos de federação de forma que os experimentos se tornem mais compreensivos e passíveis de reprodução. NOVI tem o intuito de criar uma federação entre PLE, FEDERICA e *GÈANT*, tendo como alvo a extensão da virtualização nesses ambientes a partir da combinação de máquinas virtuais e roteadores lógicos conectados pela camada 2 através de VLANs. O NOVI é apoiado pelo FP7, o Sétimo Programa-Quadro para a Pesquisa e Desenvolvimento Tecnológico, que é o principal instrumento da União Europeia para financiar a pesquisa na Europa e no mundo, em vigor de 2007 a 2013.

O NOVI é uma proposta de integração (generalização) para as diversas propostas de federações de *testbeds* até então discutidas. Sua ideia principal é suportar a integração entre os principais modelos de federação existentes. Como se pode observar na Figura 4.24, sua pretensão é suportar tanto a topologia hierárquica (SFA) quanto a P2P (Panlab/-Teagle). Para tanto, ele utiliza um conversor de formato das mensagens trocadas entre os *testbeds* heterogêneos. Os novos algoritmos, métodos e serviços do NOVI foram baseados, inicialmente, na proposta SFA, onde se tem a utilização do RSpec para descrição, solicitação de alocação de recursos, caracterização do hardware envolvido, restrições e dependendências. NOVI estende o SFA com a introdução de mecanismos sensíveis ao contexto da federação (alocação de recursos inteligente, monitoramento, gerenciamento de políticas e descoberta de recursos virtualizados) e visa a automatização de controle da fatia e das operações de gestão previstos a um ambiente de federação complexo.

4.6.5.1. Modelo de Informação

Um modelo de informação (IM) fornece semântica consistente e compartilhada, além da descrição de recursos e serviços disponíveis em um ambiente federado. No NOVI foi desenvolvido um modelo próprio de informação que deve suportar abstrações para atender a uma federação de infraestruturas e incluir os conceitos necessários para que possa ser usado para modelar outras infraestruturas de IF que venham fazer parte do NOVI.

A semântica web utilizada para definir os modelos de dados pelo NOVI é a *Web Onotology Language* (OWL) [DS04]. A escolha dessa linguagem ocorreu pelo suporte a informações de contexto e que por sua vez permitem NOVI criar serviços eficientes e complexos com os recursos disponíveis dentro da federação. São três módulos de ontologias apresentados pelo NOVI.

- Ontologia de recurso: provê os conceitos e métodos para descrever recursos presentes nas plataformas de IF, permitindo a requisição, por exemplo, de dados de um certo componente.
- Ontologia de monitoramento: provê conceitos e métodos pra operações de monitoramento que podem ser utilizadas para fornecer informações a ferramentas de monitoramento.
- Ontologia de policiamento: provê os conceitos e métodos de gerenciamento para a gerência e execução das políticas definidas para os membros da federação NOVI.

4.6.5.2. Arquitetura e Plano de Controle e Gerência

A Figura 4.23 representa o diagrama de componentes da arquitetura do NOVI. Podemos observar que o componente *Enterprise Service Bus* é considerado intermediário entre cada um dos componentes do *framework* de gerência do NOVI e os diversos *testbeds*. Destacamos a existência do componente *Normalized Message Router*, responsável por interpretar e traduzir as mensagens oriundas dos mais diversos *testbeds*. Percebe-se também o componente chamado *Policy Manager*, que é o responsável pela gestão de identidade neste ambiente. O *Policy Manager* será aquele que verificará, conforme o modelo de gestão de identidade de autorização, o papel ou atributo, por exemplo, do usuário experimentador em questão. Vale ainda citar que a alocação de recursos diretamente nos componentes NOVI é realizada por meio do *NSwitch*, funcionando da mesma forma como o RA do Teagle.

Cada componente no NOVI é tratado como um objeto, uma vez que esta proposta se baseia no conceito de SMC (*Self-Managed Cell*). SMC é um conceito vindo da tecnologia de redes ubíquas, e prevê que alguns serviços mínimos sejam suportados pelo ambiente em questão, e demais serviços possam ser adicionados. Sua política de tratamento de segurança para esses objetos é realizada de forma mais granular graças a um *framework* de segurança chamado Ponder2 [ABPO11]. O Ponder2 é a versão aprimorada do Ponder, e permite que cada objeto possa ter funcionalidades e políticas aplicadas

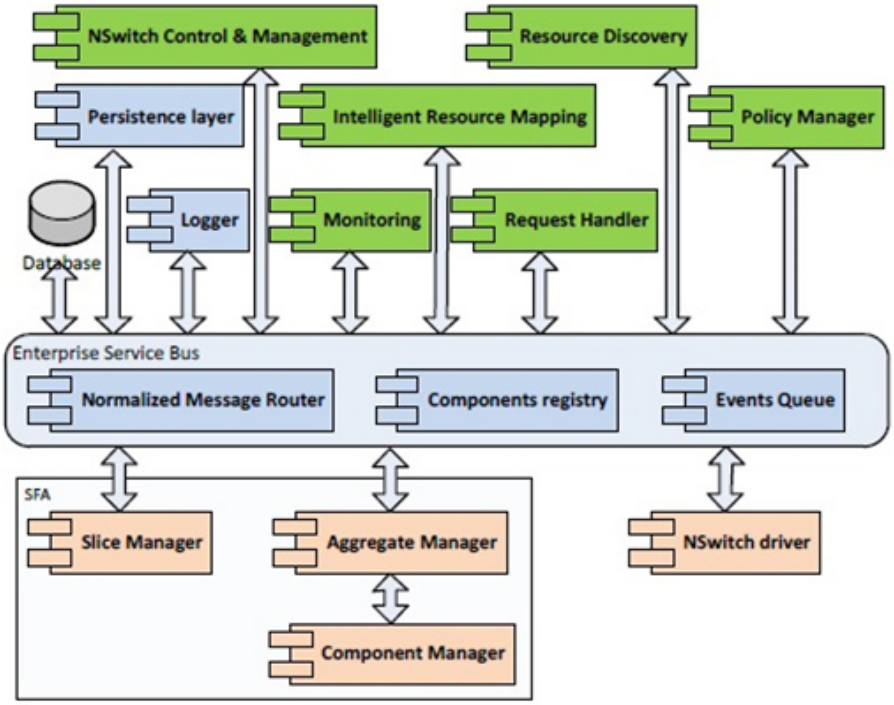


Figura 4.23. Arquitetura de componentes do NOVI. Fonte: [ABPO11]

de forma específica, o que é interessante em um ambiente distribuído e independente de experimentação em redes.

A Figura 4.24 mostra um nível mais geral da arquitetura e gerência da plataforma, sendo possível discriminar três níveis distintos, sendo o mais baixo as plataformas heterogêneas (domínios, infraestruturas) que contém os recursos virtuais alocados aos usuários. Esse nível pode ser chamado de plano de dados e apresenta a interconexão por meio do NSwitch (*NOVI's Distributed Virtual Switch*), que veremos com mais detalhes ainda nesta seção. O nível intermediário fica por conta dos recursos baseados na implementação do modelo SFA. E o nível mais alto apresenta os serviços aos usuários NOVI da federação (por exemplo, mapeamento de recursos inteligentes, policiamento, alocação de recursos, descoberta de recursos cientes de contexto, acompanhamento transparente de fatias do usuário e combinação de recursos através de domínios).

Vamos destacar cada um dos componentes das camadas do NOVI de forma sucinta, para que se compreenda melhor como NOVI pode colaborar para a integração dos ambientes de experimentação em Internet do Futuro.

API é o ponto de entrada para a interação entre o NOVI e seus componentes de gerência. Sua interface é baseada em um editor de ontologias. Basicamente suas atribuições são três:

1. Aceitar requisições de usuários autenticados requisitando recursos a um detrerminado componente;
2. Avaliar e encaminhar a solicitação do usuário ao componente;
3. Avisar ao usuário sobre a execução de seu experimento.

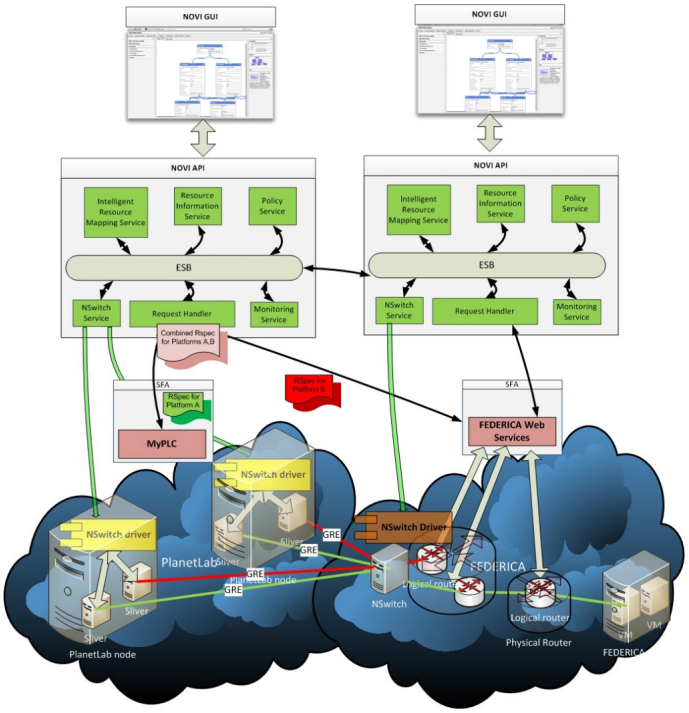


Figura 4.24. Planos de gerência do NOVI. Fonte: [LGP+12]

É possível visualizar sua interface pelo endereço http://novi-im.appspot.com/.

O **Resource Information Service (RIS)** atua como um ponto único de contato para os serviços do NOVI consultarem informações como o status de um certo recurso. Sendo *Request Handler* a interface utilizada neste momento para a comunicação com as camadas inferiores, possibilitando a consulta à reserva e anúncio de recursos. O RIS também utiliza consultas ao *Monitoring Service* para saber a disponibilidade e o status de certo recurso e o *Policy Service* para obter informações relativas aos direitos de acesso de um usuário.

As requisições de recursos e a decisão de alocação topológica nas redes virtuais envolvidas são feitas pelo **Intelligent Resource Mapping (IRM) Service**. Inicialmente formulado como um problema de domínio único para redes virtuais, o que já é considerado um problema combinatório *NP-Hard* [CB09] , foi então expandido para um problema multi-domínios pelo NOVI via *graph spitting* [HLBAZ11] e seleção inteligente por mapeamento intra-domínio.

Usuários podem submeter requisições para recursos individuais, topologias de recursos virtuais e especificar serviços de acordo com as topologias/recursos virtuais, e essas requisições são realizadas utilizando o formato Rspec.

Policy Service é o serviço responsável pelo gerenciamento do sistema baseado em sua política. Atualmente suportando políticas de controle de acesso que especificam quais direitos quais usuários devem ter para um dado tipo de recurso e uma política de ação baseada em evento, onde ações são indicadas conforme certo evento ocorre, por exemplo, uma falha em um componente.

O serviço **Request Handler Service** apresenta dois tipos de operações: o enca-

minhamento de requisições de alocação de recursos dentro da plataforma e o tratamento de chamadas externas realizadas por membros da federação.

Para o primeiro tipo é necessária a tradução para o modelo específico da plataforma. Onde, por exemplo, a comunicação entre NOVI IM e PlanetLab, mensagens devem ser traduzidas no formato RSpec do SFA. Já para segundo tipo temos o caminho oposto da requisição, onde chamadas remotas a outros membros da federação são realizadas. Sendo assim, a chamada externa pode ser realizada pelo RIS quando, por exemplo, necessita atualizar alguma informação. Ilustrando o exemplo a partir da comunicação entre o PlanetLab e o FEDERICA, a primeira etapa da comunicação com o SFA será realizada por RSpec, porém o segundo passo não poderá, já que o FEDERICA não utiliza esse modelo de mensagem, sendo assim, é necessário a a sua tradução, algo que o NOVI também provê.

O **NSwitch** (*NOVI Distributed Virtual Switch*) é um software distribuído que complementa a arquitetura de federação do NOVI porque provê a interação entre domínios heterogêneos através do plano de dados, permitindo a comunicação entre recursos através da camada 2 de forma isolada. Essa comunicação é possível através do *Open vSwitch* que provê uma camada de abstração para a comunicação dos protocolos e *hypervisors* distintos. Mais informações podem ser encontradas em [LGP+12].

4.7. Considerações Finais

Como objetivo de testar novas soluções de arquitetura de rede e protocolos para a Internet do Futuro, diversos grupos de pesquisa têm investido esforços na interconexão de suas redes experimentais. O ambiente heterogêneo resultante traz requisitos bastante desafiadores em termos de gestão de identidade, pois cada *testbed* pode ser administrado por uma instituição distinta, que utiliza políticas próprias para autorizar usuários a utilizar seus recursos.

O cenário de uso de diferentes *testbeds* pela comunidade acadêmica pode ser caracterizado como uma organização virtual (OV), com questões análogas a outros cenários como os que envolvem, por exemplo, comitês interinstitucionais formados para prover pareceres médicos ou acadêmicos.

Nessas organizações, a autorização de acesso depende de atributos definidos pela OV e não pela instituição de origem do usuário, como em outras federações de autenticação e autorização (A&A) tradicionais.

Um possível caminho a ser trilhado é a tentativa de harmonizar as redes de experimentação para Internet do futuro com a tecnologia de A&A federada, que viabiliza uma infraestrutura mais confortável para usuários e mais conveniente para administradores. Entretanto, algumas questões dificultam as propostas de uma solução harmônica para gestão de identidade nesses ambientes. Uma dificuldade é que os *testbeds* nem sempre utilizam infraestruturas federadas, ou seja, muitos possuem bases locais de usuários. Um outro ponto é que muitas redes experimentais ainda têm modelos de autorização baseados em credenciais explícitas (indicando explicitamente que determinado usuário tem autorização para determinada atividade) e não em modelos de autorização baseados em atributos, que são os utilizados em federações de A&A. Com isso, a integração de organizações virtuais em

infraestruturas federadas disponíveis hoje representa um importante desafio.

Este capítulo discutiu conceitos básicos importantes relacionados a gestão de identidade e apresentou diversas redes experimentais para a Internet do Futuro, que compõem a iniciativa americana GENI e a iniciativa europeia FIRE, incluindo o projeto FIBRE em desenvolvimento no Brasil. Foi também discutido como o problema de autenticação e autorização vem sendo tratado nesses ambientes.

O capítulo destacou algumas propostas para federação de ambientes de redes heterogêneos, que consideram a interligação de diferentes *testbeds*, comentando os projetos SFA (*Slice-based Federation Architecture*), *Panlab Federation (Teagle)* e NOVI (*Networking Innovations Over Virtualized Infrastructures*). Essas propostas ainda estão em fase de implementação e experimentação, portanto, ainda não existe uma solução única que resolva todos os desafios existentes.

Referências

[ABPO11] Nazim Agoulmine, Claudio Bartolini, Tom Pfeifer, and Declan O'Sullivan, editors. *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management, IM 2011, Dublin, Ireland, 23-27 May 2011*. IEEE, 2011.

[BC01] Marjory S. Blumenthal and David D. Clark. Rethinking the design of the internet: the end-to-end arguments vs. the brave new world. *ACM Trans. Internet Technol.*, 1(1):70–109, August 2001.

[BFH+06a] Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, and Jennifer Rexford. In vini veritas: realistic and controlled network experimentation. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, pages 3–14, New York, NY, USA, 2006. ACM.

[BFH+06b] Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, and Jennifer Rexford. In VINI veritas: realistic and controlled network experimentation. *SIGCOMM Comput. Commun. Rev.*, 36(4):3–14, August 2006.

[CB09] Mosharaf Kabir Chowdhury and Raouf Boutaba. Network virtualization: state of the art and research challenges. *Comm. Mag.*, 47(7):20–26, 2009.

[CCR+03] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. PlanetLab: an overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12, July 2003.

[CIK10] D. Chadwick, G. Inman, and N. Klingenstein. A conceptual model for attribute aggregation. *Future Generation Computer Systems*, 26(7):1043 – 1052, 2010.

[DS04] Mike Dean and Guus Schreiber. OWL web ontology language reference. W3C recommendation, W3C, February 2004.

[edu13] *eduroam - EDUcation ROAMing*. http://www.eduroam.org, Acesso em Março de 2013.

[Fal11] Aaron Falk. Federation in geni - draft proposal - comments invited. In *GENI Engineering Conferences - GEC11*, pages 1–34. SBRT, July 2011.

[fed13] *GENI - Exploring Networks of the Future - Clearing House*. http://groups.geni.net/geni/wiki/GeniClearinghouse, Acesso em março de 2013.

[Fer12] H. Ferraiolo. A credential reliability and revocation model for federated identities, 2012. NIST Interafency Report NISTIR 7817.

[FG11] Timur Friedman and Anastasius Gavras. Fire openlab ip testbed and tool demo. In *Proceedings of the 4th European conference on Towards a service-based internet*, ServiceWave'11, pages 323–324, Berlin, Heidelberg, 2011. Springer-Verlag.

[FGR07] Nick Feamster, Lixin Gao, and Jennifer Rexford. How to lease the internet in your spare time. *SIGCOMM Comput. Commun. Rev.*, 37(1):61–64, January 2007.

[fib13] *FIBRE - Future Internet Testbeds Experimentation Between Brazil and Europe - Partners*. http://goo.gl/EGj9y, Acesso em março de 2013.

[fir13] *Future Internet Research & Experimentation*. http://cordis.europa.eu/fp7/ict/fire/, Acesso em março de 2013.

[FMM+11] N. C. Fernandes, M. D. D. Moreira, I. M. Moraes, L. H. G. Ferraz, R. S. Couto, H. E. T. Carvalho, M. E. M. Campista, L. H. M. K. Costa, and O. C. M. B. Duarte. Virtual networks: Isolation, performance, and trends. *Annals of Telecommunications*, 66(5–6), 2011.

[GBW+06] Anastasius Gavras, Heinz BrÃ¼ggemann, Dorota Witaszek, Kristiina Sunell, and JosÃ© Jimenez. Pan european laboratory for next generation networks and services. In *TRIDENTCOM*. IEEE, 2006.

[Gem06] J. Gemmill. *A Trust-Relationship Management Framework for Federated Virtual Organizations*. PhD thesis, University of Alabama, 2006.

[gen13a] *GENI - Exploring Networks of the Future*. http://groups.geni.net/geni/wiki/AvailableAggregates, Acesso em março de 2013.

[gen13b] *Global Environment for Network Innovations (GENI)*. http://www.geni.net, Acesso em março de 2013.

[GKP+08] Natasha Gude, Teemu Koponen, Justin Pettit, Ben Pfaff, Martín Casado, Nick McKeown, and Scott Shenker. Nox: towards an operating system for networks. *SIGCOMM Comput. Commun. Rev.*, 38(3):105–110, July 2008.

[GVV+12] D. Guedes, L. F. M. Vieira, M. M. Vieira, H. Rodrigues, and R. V. Nunes. Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores. In *Minicursos - Livro Texto do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 161–212. SBC, 2012.

[HLBAZ11] Ines Houidi, Wajdi Louati, Walid Ben Ameur, and Djamal Zeghlache. Virtual network provisioning across multiple substrate networks. *Comput. Netw.*, 55(4):1011–1023, March 2011.

[Int13] Internet2. Shibboleth – federated single sign-on software. `shibboleth.internet2.edu`, 2013.

[ISO04] ISO (International Organization for Standardization). *9834-8:2004 Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components*. ITU-T Recommendation X.667, September 2004.

[IT09] ITU-T. Ngn identity management framework, 2009. Recommendation Y.2720.

[JB10] Sangjin Jeong and Andy Bavier. Geni federation scenarios and requirements. Technical report, July 2010.

[JP05] A. Jösang and S. Pope. User centric identity management. In *AusCERT Asia Pacific Information Technology Security Conference*, 2005.

[KW11] Andreas Köpsel and Hagen Woesner. Ofelia: pan-european test facility for openflow experimentation. In *Proceedings of the 4th European conference on Towards a service-based internet*, ServiceWave'11, pages 311–312, Berlin, Heidelberg, 2011. Springer-Verlag.

[LGP+12] Leonidas Lymberopoulos, Mary Grammatikou, Martin Potts, Paola Grosso, Attila Fekete, Bartosz Belter, Mauro Campanella, and Vasilis Maglaris. The future internet. chapter NOVI tools and algorithms for federating virtualized infrastructures, pages 213–224. Springer-Verlag, Berlin, Heidelberg, 2012.

[LKK+10] Min Lee, A. S. Krishnakumar, P. Krishnan, Navjot Singh, and Shalini Yajnik. Supporting soft real-time tasks in the Xen hypervisor. In *Proceedings of the 6th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, VEE '10, pages 97–108, New York, NY, USA, 2010. ACM.

[MAB+08] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, March 2008.

[MFCD09] M. D. D. Moreira, N. C. Fernandes, L. H. M. K. Costa, and O. C. M. B.

Duarte. Internet do futuro: Um novo horizonte. In *Minicursos do Simpósio Brasileiro de Redes de Computadores*, SBRC'2009, pages 1–59. SBC, May 2009.

[New12] OFELIA Project Newsletter. Technical Report 9, December 2012.

[OAS05] Assertions and protocol for the security assertion markup language (SAML) v2.0. Organization for the Advancement of Structured Information Standards (OASIS), 2005.

[ofe13a] *i2CAT Distributed Applications and Networks Area - Expedient tool: a key component for OFELIA Control Framework*. http://dana.i2cat.net/expedient-tool-a-key-component-for-ofelia-control-framework/publications/, Acesso em março de 2013.

[OFE13b] *Ofelia - General Terminology*. https://alpha.fp7-ofelia.eu/doc/index.php/General_terminology, Acesso em março de 2013.

[one13] *OneLab - Future Internet Testbeds*. http://www.onelab.eu/, Acesso em março de 2013.

[Ope11] *OpenFlow Switch Specification - Version 1.1.0 Implemented (Wire protocol 0x02)*, February 2011.

[pla13] *PlanetLab - An Open platform for developing, deploying, and accessing planetary-scale services - Federation*. https://www.planet-lab.org/federation, Acesso em março de 2013.

[PRFC08] Larry Peterson, Robert Ricci, Aaron Falk, and Jeff Chase. Slice-based facility architecture. Technical report, July 2008.

[PRFC10a] Larry Peterson, Robert Ricci, Aaron Falk, and Jeff Chase. Slice-based facility architecture. Technical report, July 2010.

[PRFC10b] Larry Peterson, Robert Ricci, Aaron Falk, and Jeff Chase. Slice-based federation architecture. Technical report, July 2010.

[pro13a] *ProtoGeni ClearingHouse*. http://www.protogeni.net/ProtoGeni/wiki/ClearingHouseDesc, Acesso em março de 2013.

[pro13b] *ProtoGeni ClearingHouse*. http://www.protogeni.net/wiki/ClearingHouseDesc, Acesso em março de 2013.

[RD10] Jennifer Rexford and Constantine Dovrolis. Future internet architecture: clean-slate versus evolutionary research. *Commun. ACM*, 53(9):36–40, September 2010.

[ROJS10] Thierry Rakotoarivelo, Maximilian Ott, Guillaume Jourjon, and Ivan Seskar. OMF: a control and management framework for networking testbeds. *SIGOPS Oper. Syst. Rev.*, 43(4):54–59, January 2010.

[SAM+12] Sebastia Sallent, Antonio Abelém, Iara Machado, Leonardo Bergesio, Serge Fdida, José Rezende, Dimitra Simeonidou, Marcos Salvador, Leandro Ciuffo, Leandros Tassiulas, and Carlos Bermudo. FIBRE project: Brazil and Europe unite forces and testbeds for the Internet of the future. In *Proceedings of TridentCom 2012*, June 2012.

[SCC+10] Rob Sherwood, Michael Chan, Adam Covington, Glen Gibb, Mario Flajslik, Nikhil Handigol, Te-Yuan Huang, Peyman Kazemian, Masayoshi Kobayashi, Jad Naous, Srinivasan Seetharaman, David Underhill, Tatsuya Yabe, Kok-Kiong Yap, Yiannis Yiakoumis, Hongyi Zeng, Guido Appenzeller, Ramesh Johari, Nick McKeown, and Guru Parulkar. Carving research slices out of your production networks with openflow. *SIGCOMM Comput. Commun. Rev.*, 40(1):129–130, January 2010.

[SCC12] Bruno Oliveira Silvestre, Kleber Vieira Cardoso, and Sand Luz Corrêa. Controle e monitoramento de experimentos para a internet do futuro usando omf. In *Minicursos - Livro Texto do XXX Simpósio Brasileiro de Telecomunicações*, pages 1–34. SBRT, 2012.

[sfa13a] *i2CAT Distributed Applications and Networks Area - SFA for FEDERICA within the NOVI project*. http://dana.i2cat.net/872/uncategorized/, Acesso em março de 2013.

[sfa13b] *Scripts do SFA em Python*. http://svn.planet-lab.org/svn/sfa/trunk/, Acesso em marÃ§o de 2013.

[SFC+09] Peter Szegedi, Sergi Figuerola, Mauro Campanella, Vasilis Maglaris, and Cristina Cervelló-Pastor. With evolution for revolution: managing federica for future internet research. *Comm. Mag.*, 47(7):34–39, July 2009.

[Sun] Marc Sune. 1st version of the ofelia management software. Technical Report 5.1.

[WHC+10] Sebastian Wahle, Bogdan Harjoc, Konrad Campowsky, Thomas Magedanz, and Anastasius Gavras. Pan european testbed and experimental facility federation architecture refinement and implementation. *Int. J. Commun. Netw. Distrib. Syst.*, 5(1/2):67–87, July 2010.

[Win99] Dve Winer. XML-RPC Specification, June 1999.

[WMB+10] M. Wangham, E. Mello, D. Böger, M. Guérios, and J. Fraga. Gerenciamento de identidades federadas. In Luciano Porto, editor, *Livro de Minicursos do SBSEG*. SBC, 2010.

[xmp13] *XMPP in OMF 5.2*. https://mytestbed.net/projects/1/wiki/XMPP_in_OMF, Acesso em março de 2013.