UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA DA COMPUTAÇÃO

UMA ARQUITETURA DE NOMEAÇÃO PARA A INTERNET UTILIZANDO REDES VIRTUAIS

Joelle Quaini Sousa

SÃO CARLOS
2007

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

S725an

Sousa, Joelle Quaini.
Uma arquitetura de nomeação para a internet utilizando redes virtuais / Joelle Quaini Sousa. -- São Carlos : UFSCar, 2008.
127 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2007.

1. Redes de computação – protocolos. 2. DNS. 3. Internet (Redes de computação). 4. Banco de dados. I. Título.

CDD: 004.62 (20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"Uma Arquitetura de Nomeação para Internet Utilizando Redes Virtuais"

JOELLE QUAINI SOUSA

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Membros da Banca:

Hélio Crestana Guardia

Prof. Dr. Hélio Crestana Guardia
(Orientador - DC/UFSCar)

Tereza Cristina Melo de Brito Carvalho

Profa. Dra. Tereza Cristina M. de Brito Carvalho
(Co-Orientadora – PCS/EPUSP)

Prof. Dr. Luis Carlos Trevelin
(DC/UFSCar)

Mauricio Ferreira Magalhães

Prof. Dr. Mauricio Ferreira Magalhães
(DCA/FEEC/UNICAMP)

São Carlos
Novembro/2007

À minha mãe, Emilia, que deu apoio
incondicional e constante a todas as
quimeras de minha vida, incluindo essa.

AGRADECIMENTOS

Agradeço a Deus pelas pessoas notáveis em minha vida, considero a presença delas uma benção divina.

À minha mãe, por sempre me permitir acreditar em todos os sonhos tangíveis ou surreais que insistiram em se concretizar.

Ao Prof. Dr. Hélio Crestana Guardia, meu orientador, pelos constantes questionamentos que permitiram dar forma final ao mestrado. À Profa. Dra. Tereza Cristina M. B. Carvalho, minha co-orientadora, cuja atuação protegeu e fomentou este trabalho durante todos os estágios de sua concepção e desenvolvimento. Ao Prof. Dr. Wilson Vicente Ruggiero pelo instigante curso de Análise de Desempenho.

À Ericsson do Brasil e Suécia pelo suporte e patrocínio ao desenvolvimento de pesquisas avançadas na área de Redes Virtuais, das quais este mestrado é fruto.

Ao Marcio Augusto, que com seu aguçado intelecto auxiliou sobremaneira a definição de várias questões críticas. Muito da originalidade desta dissertação deve ser atribuída a ele. Ao Reinaldo Matushima pelo apoio fundamental, principalmente quanto ao protótipo.

Ao meu pai e irmãs, Priscila, Jessica e Leticia pelo suporte e encorajamento durante toda minha vida. Aos meus avôs Giacomo e Nelsan, pelas boas lembranças que tenho guardadas.

Ao Rafael pelo amor e dedicação sem igual, me transformando em uma pessoa melhor. À Pricoli, pela sua preciosa amizade, sabedoria e resignação em me fazer entender que Deus sempre quer o nosso bem. A todos meus amigos, em especial ao Eugeni, Kell, Ciba, Fabi, Toledo, Fernanda, Piqueira, Tork e Patrícia, pela compreensão durante todo o tempo que fomos privados de nossa convivência.

Ao Wolf, pelo carinho que despertou dentro de mim deixando uma saudade incomensurável. À Bianca por todo seu carinho e proteção durante toda sua vida.

RESUMO

Face a vários novos requisitos de comunicação demandados por equipamentos em desenvolvimento constante, tais como computadores móveis portadores de múltiplas interfaces comunicantes, devido à inserção de diversos *middleboxes* [1], o modelo arquitetural TCP/IP necessita ser aprimorado para suportar novas tecnologias e protocolos. Originalmente, quando a Internet foi projetada, no final dos anos 70 nem mobilidade nem *multihoming* (i.e. equipamento com diversas conectividades físicas simultâneas) foram considerados.

Pela proposição de uma nova arquitetura de nomeação para a Internet, que seja capaz de identificar univocamente qualquer entidade comunicante, bem como proporcionar suporte às tecnologias já extensamente utilizadas, este trabalho objetivou promover a mobilidade e o suporte a diversos *middleboxes* para a Internet, principalmente no que diz respeito à identificação e à autenticação de nós e objetos (i.e. serviços, dados e usuários).

Almejando atingir tal objetivo utilizou-se a tecnologia de Redes Virtuais, que permite uma adesão incremental de suas funcionalidades, protocolos e aplicações. Esta abordagem não representa, portanto, um modelo cujas mudanças à arquitetura da Internet causam-lhe transformações estruturais, diferentemente de outras propostas que abordaram este problema desta forma sem sucesso [2; 3].

Para tanto, uma taxonomia de Redes Virtuais foi proposta e avaliada por um estudo de caso que compreendeu a sua aplicação prática. Além disso, realizou-se uma análise de bancada de redes de diversos protocolos e o estudo das propostas da literatura associada. Tais realizações culminaram na proposta de uma Arquitetura de Nomeação para a Internet utilizando Redes Virtuais *Overlay*.

Palavras-chaves: Arquitetura TCP/IP, DNS, *Overlays*, Redes Virtuais e Tabelas *Hash* Distribuídas.

ABSTRACT

Regarding new computational and networking requisites such as wireless networks, mutihoming interfaces, load-balancing mechanisms and several other middleboxes [1] present today, these facts, allied to the static and conservative nature of the Internet and its sheer size turn the capability to correct these problems an almost impossible attempt, as it demands structural changes. In the Internet inception, in the late 70th, neither mobility nor multihoming where foreseen in its original intents.

In this sense, the proposition of a novel naming architecture for the Internet to identify univocally services and data, irrespective to its node characteristics, would have an acute changing effect and will allow its elements to be precisely represented and authenticated.

In order to achieve these purposes, the use of Virtual Networks was considered as it allows the incremental introduction of new technologies, protocols and applications being itself a more viable alternative when compared to several failed attempts to introduce new structural changes to the Internet [2; 3].

A proposal for a taxonomy for Virtual Networks was described here as a result of a site survey that was conducted to function as subject to this architecture proposition. Besides, a literature investigation of related projects followed by a network testbed of several protocols originated the proposition of a Layered Naming Architecture for the Internet using Virtual Networks.

Keywords: Distributed Hash Table, DNS, Overlays, TCP/IP Architecture and Virtual Networks.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ADSL	*Asymmetric Digital Subscriber Line*
AES	*Advanced Encryption Standard*
ASNe	*Application Specific Network*
CAN	*Content Addressable Network*
BGP	*Border Gateway Protocol*
DDoS	*Distributed Denial of Service*
DHCP	*Dynamic Host Configuration Protocol*
DHT	*Distributed Hash Table*
DNS	*Domain Name/Naming Server/System*
DoS	*Denial of Service*
GPRS	*General Packet Radio Service*
HIT	*Host Identity Tag*
HTTP	*Hyper Text Transfer Protocol*
IP	*Internet Protocol*
IPsec	*Internet Protocol Security*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
IPNL	*IP Next Layer*
NAT	*Network Address Translator*
OSI	*Open System Interconnection*
OSPF	*Open Shortest Path First*
P2P	*Peer-to-Peer*
PPP	*Point-to-Point Protocol*
PSNe	*Purpose Specific Networks*
PSTN	*Public Switched Telephone Network*
RIP	*Routing Information Protocol*
RRS	*Resolution Reference Service*

RSVP	*Resource ReserVation Protocol*
SFR	*Semantic Free Referencing*
SSL	*Secure Sockets Layer*
TCP	*Transmission Control Protocol*
TRIAD	*Translating Relaying Internet Architecture integrating Active Directories*
UDP	*User Datagram Protocol*
URI	*Uniform Resource Locator*
WBI	*Web Based Intermediary*
WLAN	*Wireless Local Area Network*
WRAP	*Wide-Area Relaying Protocol*

SUMÁRIO

CAPÍTULO 1 INTRODUÇÃO

1.1 Apresentação

A importância da comunicação pode ser notada pela constante busca histórica por meios cada vez mais eficientes para se obter a troca de informações. Diversos modelos de comunicação foram concebidos, desde os meios mais primitivos baseados em sinais de fumaça até o modelo atual de comunicação eletrônica, entendido pela transmissão de sinais entre agentes codificadores e decodificadores, que podem ser simultaneamente emissores e receptores. A viabilização de meios capazes de transportar informações na forma de sinais eletromagnéticos sejam elas das mais variadas origens (dados, voz ou vídeo) permitiu que as transações modernas acontecessem quase que instantaneamente e em escala mundial.

A complexidade de uma rede como a Internet, cujas proporções geográficas abrangem uma escala planetária de computadores e igualmente de usuários, é notória. Somado a isto, temos o aumento vertiginoso do uso de equipamentos móveis modernos com múltiplas interfaces de rede conectadas simultaneamente (*multihoming*), tais como GPRS (*General Packet Radio Service*) e WLAN (*Wireless Local Area Network*) que requerem mecanismos de controle de acesso e escolha da melhor interface de conectividade em tempo real.

Entretanto, nem mobilidade ou *multihoming* foram considerados na concepção da arquitetura TCP/IP, cenário perfeitamente aceitável, considerando-se que naquele período cada equipamento pertencia a um único domínio e raramente (ou nunca) se movia. Com a introdução do mapeamento dinâmico PPP (*Point to Point Protocol*) [4] e DHCP (*Dynamic Host Configuration Protocol*) [5], a pré-concepção de que o endereço IP (*Internet Protocol*) identificaria um nó (entidade comunicante) unicamente foi invalidada e a situação se agravou com a introdução de endereços IP privados e o surgimento de diversos *middleboxes* [1] (tais como Túneis IP, adaptadores *SOCKS*, Classificadores de Pacotes, Balanceadores de Carga, *Firewalls* de rede, *Firewalls* de Aplicação, entre outros).

No estado presente da arquitetura TCP/IP, o protocolo IP identifica simultaneamente o nome e o endereço das entidades. Esta sobrecarga de informações (identificação e endereçamento) em uma só nomenclatura apresenta diversos problemas, principalmente quanto à segurança (identificação e autenticação de nós) e mobilidade. Desta

forma, dado o estabelecimento de uma conexão baseada no endereço IP e na porta da aplicação (elemento referente à quarta camada do modelo OSI) de nós pares, a troca da interface de acesso à Internet pelos mais variados motivos (a exemplo da procura de uma melhor conectividade ou advinda da mobilidade) causaria a quebra da semântica das conexões (atualmente baseada no par **<IP, porta>** de uma interface programática ponto-a-ponto) outrora estabelecidas.

Diante de novos requisitos de comunicação (e.g. mobilidade, *multihoming,* autenticação), o modelo arquitetural TCP/IP apresentou-se incapaz de suportá-los de maneira natural, ou seja, sem lançar mão de diversos artifícios causadores de rupturas estruturais e semânticas (como, por exemplo, através da inserção de *middleboxes*). Nesse sentido, este trabalho almeja a proposição de uma nova estrutura de nomeação para a Internet, que seja capaz de identificar univocamente qualquer nó computacional comunicante (evitando, por exemplo, o não-repúdio e garantindo a sua autenticação) sem deixar de proporcionar suporte às tecnologias já extensamente utilizadas.

A tecnologia de Redes Virtuais*,* explicada no capítulo três, que permite uma adesão incremental de suas funcionalidades com mínimas modificações à infra-estrutura atual da Internet, representa uma idéia atrativa para a concepção da nova arquitetura quando comparada à proposição de arquiteturas completamente novas (e.g. IPv6 [6]). A adição de novas camadas de protocolos para a identificação unívoca de objetos (aplicações e interfaces) descrita neste trabalho tem como conseqüência, também, a resolução de problemas de segurança, principalmente os relacionados ao não-repúdio, privacidade e o suporte à mobilidade e *multihoming*.

1.2 Objetivos

O objetivo central deste trabalho consiste na proposição de uma nova arquitetura de nomeação para a Internet que possibilite a identificação unívoca de objetos, sejam eles serviços ou interfaces. Para tal, tem-se os seguintes objetivos específicos:

a. **Análise de Projetos Relacionados** para definir quais componentes são almejados pela comunidade da Internet e quais já estão funcionais de acordo com a literatura de redes virtuais;

b. **Proposição de uma taxonomia** de classificação para redes virtuais que possibilite a formalização de termos e, consequentemente, permita estabelecer parâmetros de comparação entre projetos;
c. **Testes de Desempenho** que utilizem alguns dos projetos mais relevantes do item a. (aqueles que possuam implementação disponível) para esclarecer a viabilidade de adoção parcial ou total de algum modelo pré-existente como produto final deste projeto.
d. **Especificação Funcional** de um sistema de nomeação com múltiplas camadas, sem hierarquia, para a Internet baseado em Redes Virtuais, prestando-se a esclarecer quais suas vantagens em relação ao sistema tradicional e quais as principais funções desempenhadas pelo mesmo.
e. **Especificação de Requisitos** que explicitarão quais componentes são contemplados pela nova infra-estrutura e, portanto, quais características se espera habilitar com sua utilização.
f. **Proposição formal** que define concretamente quais artifícios, camadas e tecnologias serão suportadas por esta arquitetura baseadas nos resultados obtidos em a., b. e c. e a classifica. Uma prova de conceito deve ser disponibilizada.

Cada um destes objetivos específicos deve gerar um subproduto, sendo que a junção dos mesmos deve integrar um único documento que possa servir de guia para um possível protótipo para a Internet.

1.3 Motivação e Relevância

O amadurecimento dos equipamentos e tecnologias de acesso à Internet vivenciado nas últimas décadas por usuários residenciais e comerciais, que inicialmente utilizavam linhas telefônicas com acesso ADSL (*Asymmetric Digital Subscriber Line*) através de seus computadores pessoais (*desktops*) e, então, puderam contar com novos equipamentos móveis (*laptops*) com acesso sem fio e mais recentemente com telefones inteligentes (*smart phones*) com acesso GPRS, CDMA, EDGE estabeleceu diversos novos desafios.

Do ponto de vista dos usuários, os desafios estão diretamente relacionados com a complexidade de configuração de parâmetros dos sistemas operacionais que atuam neste ambiente heterogêneo de redes e equipamentos. Não menos desafiador, do ponto de vista dos

provedores de serviço e de infra-estrutura de comunicação, o problema principal se encontra na demanda constante por mobilidade, por um ambiente seguro, pela capacidade de expandir-se continuamente dado o crescimento vertiginoso da Internet e pela disposição em adotar novas tecnologias que são desenvolvidas diariamente.

Em virtude de tantos requisitos computacionais e da heterogeneidade de ambientes de rede, faz-se necessária uma arquitetura que seja capaz de atender naturalmente tais desafios (e.g. mobilidade, *multihoming*, segurança e etc.), capacidade ausente na Internet dos dias atuais baseada na arquitetura TCP/IP.

Contudo, a proposição de um novo modelo arquitetural em substituição ao TCP/IP, a ser adotado em escala mundial e capaz de atender a todos os requisitos citados apresenta-se como uma idéia conveniente, porém não factível. Isto se deve à impossibilidade de troca de todos os inúmeros equipamentos, protocolos e infra-estrutura já implantados, além do fato de que isso causaria momentos consideráveis de indisponibilidade, um alto custo financeiro, sem considerar a resistência natural esperada por parte dos provedores e usuários na adoção de novas tecnologias e protocolos.

Dessa forma, a proposição de tecnologias complementares a serem implantadas de forma incremental ao modelo atual apresenta-se como uma solução mais adequada. Neste sentido, a utilização de redes virtuais junto à arquitetura TCP/IP com o objetivo de habilitá-la a suportar diversas características não naturalmente projetadas representa uma alternativa atrativa e factível. A numerosa quantidade de pesquisas já apresentadas na comunidade de redes comprova sua eficiência [7; 8; 9].

Este trabalho contempla, portanto, o uso de redes virtuais na proposição de um novo sistema de nomeação unívoco e não hierárquico (em contrapartida ao modelo hierárquico atual do DNS – *Domain Name System*) para a Internet que permita desacoplar a identificação e localização geográfica dos equipamentos. O desacoplamento obtido por esse novo sistema permite que a mobilidade, entre outras características, ocorra de maneira natural. Um detalhamento meticuloso de cada parâmetro a ser adotado será discorrido ao longo deste texto.

1.4 Estrutura do Trabalho

Esta dissertação consiste em sete capítulos, que seguem explanados. O primeiro capítulo, intitulado “Introdução”, descreve o problema em questão, o contexto

motivador, enumerando os principais objetivos esperados e a relevância do projeto. No capítulo dois, é apresentada uma fundamentação teórica necessária para o entendimento dos demais capítulos.

Utilizando conceitos, termos e técnicas explanados no capítulo dois, o capítulo seguinte apresenta uma proposta de taxonomia para Redes Virtuais, formalizando seus termos e características. Em seguida, uma aplicação prática baseada em projetos presentes na Internet, apresentada na forma de um estudo de caso desta taxonomia, é realizada no capítulo quatro.

O quinto capítulo apresenta o trabalho realizado, explicitando todas as atividades que foram executadas para a obtenção da arquitetura de nomeação com múltiplas camadas desenvolvida. Em seguida, o sexto capítulo sintetiza os principais resultados, parciais e finais, obtidos neste trabalho tecendo elucidações sobre os mesmos.

Finalmente, o último capítulo conclui este trabalho esclarecendo quais as principais características que beneficiarão a Internet com sua implementação, bem como quais as principais dificuldades a serem enfrentadas em possíveis trabalhos futuros.

CAPÍTULO 2 FUNDAMENTAÇÃO TEÓRICA

Apesar de seu consagrado sucesso, a Internet apresenta-se longe do “ideal”. Com o advento da ubiqüidade, aliado ao crescimento vertiginoso do número de usuários e computadores conectados, as deficiências que acometem a Internet mostraram-se cada vez mais evidentes.

A necessidade de uma mudança arquitetural nunca foi tão importante como agora, como pode ser observado pelo número crescente de propostas que emergiram na comunidade de pesquisa [2; 3; 8; 10; 11] para solucionar suas falhas. Ironicamente, o crescimento da Internet que motivou estas propostas agora tornou seu sucesso improvável: o tamanho do cenário da Internet já instalado representa um entrave para propostas que requerem mudanças significativas na infra-estrutura em funcionamento envolvendo inúmeros roteadores. Os esforços de anos para tornar padrão o uso do IPv6 representam uma prova plena da dificuldade de se substituir infra-estruturas [12] em produção.

Em virtude da dificuldade de se propor modelos que requeiram alterações estruturais, uma possível solução que enfoca uma infra-estrutura mais flexível de nomeação implementada acima da camada de rede (i.e. que não altere a estrutura funcional dos roteadores) mostra-se como uma idéia atrativa. Porém, fica claro que questões diretamente relacionadas com o roteamento, tais como: proteção completa de ataques DoS [13], controle apurado sobre o roteamento de pacotes, qualidade de serviço, entre outras, não são tratadas por esta proposta.

2.1 Infra-estruturas de Nomes

A importância da nomeação nos sistemas computacionais representa uma peça fundamental para a implementação de novas funções, na medida em que a atribuição de nomes permite referenciar seus componentes associados. Após estabelecer uma infra-estrutura de nomes, todas as outras funções são viabilizadas, como por exemplo, o endereçamento, roteamento, segurança, mobilidade entre outras. Desta forma, a atribuição de nomes deve ser uma tarefa cautelosa e criteriosa na medida em que afetará as demais funcionalidades arquiteturais.

Três principais classes de espaços de nomes são possíveis de serem implementadas: espaços de nomes com hierarquia (i.e. análogos ao modelo do DNS atual); espaços de nomes sem hierarquia, tal como é implementado pelo projeto SFR (*Semantic-Free Referencing)* [14] e, finalmente, um modelo que mescla os dois anteriores.

A infra-estrutura de nomes da Internet atual, baseada no DNS, tem desempenhado seu papel com excelência a considerar que em sua concepção diversas funcionalidades não foram previstas. Uma ilustração do mecanismo de funcionamento do DNS normal e reverso, que converte nomes em nível de usuários para endereços IP e vice-versa, pode ser observada na Figura 1. Existem somente dois espaços de nomes globais, o do DNS e de endereçamento IP, ambos fortemente acoplados com a estrutura pré-existente da Internet (domínios e topologias/segmentos de rede, respectivamente).

A rigidez e a dificuldade de expansão deste espaço de nome hierárquico e do seu endereçamento são responsáveis por uma variedade de problemas arquiteturais. Além disso, esta classe de espaço de nomes impõe sérios desafios à mobilidade, como pode ser observado na arquitetura TCP/IP, na medida em que as referências adotadas embutem não só características classificatórias, como seria desejável, mas também, particulares à localização ou administração do nó que as hospeda, claramente dificultando sua eventual mobilidade.

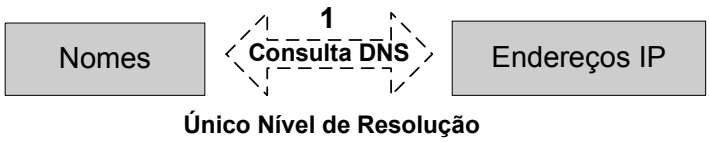

Figura 1. Resolução de Nomes DNS na Internet

Outros desafios são encontrados. Pode-se citar o fato de a estrutura de nomeação da Internet ser fortemente orientada à identificação de nós, o que dificulta a identificação granular de serviços e dados. Somado a isso, temos a constante adoção de *middleboxes*, na tentativa de acomodar características não naturalmente suportadas (e.g. endereços privados, balanceamento de carga, filtro de pacotes), mecanismos que violam o modelo original da Internet em diversos aspectos. Não é preservada a conectividade fim-a-fim com a inserção de *middleboxes*; além disso, decisões de encaminhamento que deveriam ser integralmente tomadas pelo roteamento IP, podem agora ser influenciadas pela presença dos *middleboxes* para nós estratégicos (como acontece com o uso de balanceadores de carga).

A busca incessante por um sistema de nomes mais adequados para a Internet tem se tornado comum e acarretado, por outro lado, em novos desafios para o DNS aumentando a complexidade de sua estrutura, conforme Walfisha (2004) [14]. Além disso,

conflitos políticos e sociais relacionados ao uso de nomes com semântica já representam um grande problema, haja vista as disputas legais por nomes "conhecidos" e a planificação, ou melhor dizendo, a quebra do balanceamento dos domínios raízes do DNS com o aumento vertiginoso da procura por nomes em certos níveis (atualmente 80% dos nomes encontram-se concentrados em apenas dois domínios [15] da Internet).

Para solucionar esses, entre outros problemas, níveis adicionais de nomeação se fazem necessários. Em conformidade com Stoica *et al* [16], quatro camadas de nomeação são necessárias: uma para acomodar nomes em nível de usuário como o DNS faz atualmente; outra para identificar serviços (SIDs – *Service Identifiers*); uma terceira para identificar nós computacionais (EID – *End Point Identifiers)*; e finalmente aquela que irá realizar a mesma função do endereçamento IP, identificando as diretivas de encaminhamento de pacotes. Na Figura 2 são ilustrados os três níveis de resolução necessários para acomodar adequadamente estes novos espaços de nomes. Traduz-se os nomes convencionais para Identificadores de Serviço (passo 1), em seguida estes nos Identificadores de Nós (passo 2) e finalmente encontra-se o seu respectivo endereço (passo 3).

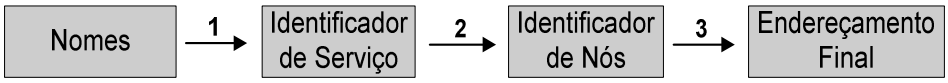

Figura 2. Novo Sistema de Resolução de Nomes [16]

Esta proposição de inserção de duas novas camadas de nomeação desvinculadas não representa uma novidade, sua concepção é advinda da taxonomia de Saltzer *et al* [17] concebida em 1993 sobre elementos constituintes de uma comunicação em rede. Nela, são definidos quatro elementos básicos de comunicação:

a. **Serviços e Usuários**. Serviços são as funções utilizadas pelos usuários e, por conseqüência, estes são clientes daqueles. Um exemplo de serviço é uma aplicação que fornece horário, e um exemplo de cliente é a aplicação que requisitou o serviço.
b. **Nós**: São computadores que podem executar serviços ou aplicações de usuários. Alguns são clientes ou prestadores de serviços de uma rede, enquanto outros podem simplesmente executar funções de encaminhamento de pacotes (nenhuma distinção é feita entres estes dois tipos).

c. **Pontos de Acesso à Rede**: Consistem em portas de acesso a uma rede ou, simplesmente, correspondem aos elementos que conectam os nós às redes. Como um exemplo simples pode ser citado uma interface de rede, que conecta um nó à sua rede de acesso e que possui um identificador, atualmente denominado sob o termo "endereço".
d. **Caminho**. Constitui a ligação entre dois nós finais de acesso à rede; percorrendo nós de encaminhamento intermediários e enlaces de comunicação.

Esta nomenclatura serve de base para este trabalho, e seus termos serão deliberadamente emprestados na identificação de subelementos das seções seguintes.

2.2 Infra-estrutura de Armazenamento: Tabelas Hash Distribuídas

Com a popularização dos sistemas de compartilhamento de arquivos baseados em P2P, um exemplo de rede virtual, emergiu o desafio central de torná-los escaláveis. Os primeiros sistemas, tais como o Napster [18] e Gnutella [19], apresentavam algumas limitações que dificultavam sua distribuição em larga escala: o Napster utilizava um serviço de diretório centralizado e o Gnutella realizava suas consultas através de um mecanismo de inundação. Inspirado por estes desafios, diversos sistemas de Tabelas *Hash* Distribuídas (DHT, ou Tabelas de Dispersão Distribuídas) foram propostos [20; 21; 22; 23; 24].

Almejando o armazenamento eficiente de grandes quantidades de dados concomitantemente com a necessidade de busca e recuperação de informações em tempo reduzido (requisitos comumente exigidos na Internet), surgiu a necessidade das DHT. Sua composição original é advinda das estruturas de dados de Tabelas *Hash* simples que mapeiam chaves e seus respectivos valores de forma singular (i.e. não distribuída) sendo capazes de transformar uma chave em um *hash* através de uma função transformadora. Uma função de cálculo de *hash* consiste em uma transformação matemática, que consome uma mensagem de comprimento arbitrário e a converte em um dado com número fixo de bits, denominado valor de *hash* [25]. Isto é, dada uma entrada de comprimento qualquer, a partir da seqüência destes *bits* de entrada, a função de *hash* calcula e retorna um número fixo, conhecido como valor de *hash.*

Funções *hash* não são reversíveis e geralmente mapeiam de maneira eficiente chaves para um conjunto de milhões de posições (por exemplo, inteiros). O mapeamento de

múltiplas chaves para a mesma posição acarreta em colisões que podem ser tratadas de maneira ativa, ou seja, notificando-se o evento ocorrido para que seja tratado, ou de maneira passiva, em que se sobrescreve o conteúdo. Funções de mapeamento perfeitas são livres de colisão, entretanto possuem um alto consumo de processamento no gerenciamento de suas chaves.

Tabelas *Hash* Distribuídas [26] constituem, fundamentalmente, dicionários capazes de mapear chaves em posições específicas de um conjunto de vetores. Elas representam uma classe descentralizada de sistemas que partilham o armazenamento de chaves entres os diversos nós participantes da infra-estrutura de rede (DHT) e são capazes de realizar a localização de um dado a partir da busca de certa chave conhecida utilizando tabelas de roteamento para encaminhar as requisições para o próximo nó.

Cada nó participante da DHT divide a responsabilidade pelo armazenamento de uma parte do espaço de chaves, sendo que nenhuma inferência pode (nem deve) ser feita sobre o conteúdo armazenado. Desta forma, não é possível (e nem desejável) garantir que um nó seja responsável pelas suas chaves em particular. O objetivo é ratear o custo de armazenamento e distribuir os dados entre todos os nós aumentando a disponibilidade dos mesmos e provendo a independência de consultas, ou seja, a resposta é obtida de forma independente de qual(is) nó(s) será(ao) consultado(s). Além disso, DHTs permitem que nós ingressem e deixem o sistema a qualquer momento sem que haja o comprometimento das informações armazenadas. Isto é possível graças à replicação de todos os dados armazenados [27] entre os diversos nós.

Para manter estas informações em um ambiente onde cada nó pode entrar ou sair em qualquer momento, são utilizados métodos de replicação da informação, ou *erasure codes* [28], onde um bloco de informação de tamanho **s** é divido em um número **n** de fragmentos de tamanho **k** (sendo k<s), e quaisquer **m** fragmentos distintos (sendo m<n) são suficientes para a reconstrução do bloco. O balanceamento entre as proporções de variáveis n, k e n é variável e dependente da implementação adotada, conforme será exemplificado adiante.

Uma ilustração dos mecanismos básicos para o funcionamento de uma DHT de propósito geral pode ser observada na Figura 3. Para utilizar os serviços de uma DHT, aplicações distribuídas podem valer-se de um conjunto de chamadas bem definidas para a inserção, procura e remoção de chaves e dados, conforme ilustrado na Figura 3. Quando uma aplicação ou usuário deseja encontrar uma chave **k** conhecida, esta pode utilizar a chamada

Procura (k) e em caso de sucesso, obter-se-á o conteúdo associado a aquela chave, ou seja, os seus **Dados**.

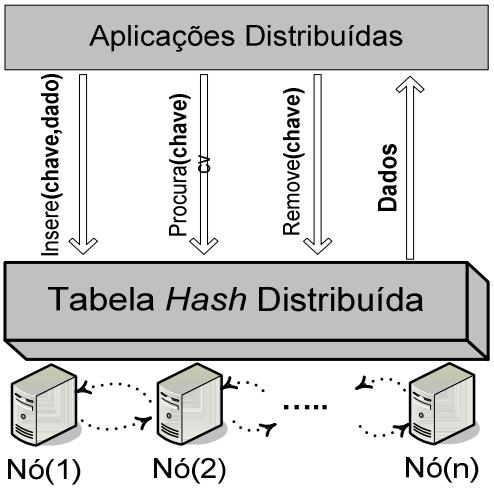

Figura 3. Funcionamento de uma DHT Genérica

As principais características obtidas com a utilização de DHTs são as seguintes:

- Localização determinística;
- Balanceamento de carga;
- Onisciência não obrigatória por parte dos nós;
- Ausência de um ponto de falha centralizado;
- Grande capacidade de armazenamento e expansão;
- Distribuição uniforme de recursos;
- Utilização de chaves opacas (não passíveis de serem memorizadas).

2.3 Classes Gerais de Interfaces DHTs

DHTs foram apresentadas para a comunidade acadêmica por volta de 2001, quase que simultaneamente, com quatro diferentes arquiteturas: CAN (*Content-Addressable Network*) [21], Chord[22], Pastry[29] e Tapestry [23]. Uma breve descrição sobre essas arquiteturas e seus algoritmos de roteamento será aqui detalhada seguida pela descrição mais detalhada do projeto OpenDHT [30], aqui selecionado como candidato deste projeto.

Três classes gerais de interfaces DHT podem ser encontradas na literatura, ocupando cada uma delas diferentes papéis no espectro da generalidade/simplicidade (uma taxonomia detalhada pode ser encontrada em [31]). A partir de uma dada chave, as seguintes funcionalidades são proporcionadas em cada uma das classes abaixo:

- **Roteamento:** proporciona acesso ao nó DHT responsável pela chave buscada e a todos os nós envolvidos no encaminhamento.
- **Busca:** proporciona acesso ao nó DHT responsável pela chave buscada.
- **Armazenamento:** provê operações de inserção e recuperação de dados através do roteamento aos nós responsáveis pelo armazenamento.

O modelo de roteamento consiste no mais genérico de todos, permitindo aos clientes realizar chamadas de código em cada nó envolvido no encaminhamento, o que facilita implementações de sistemas DHT de *multicast* [32] e *anycast* [33]. O modelo de busca, menos genérico, permite somente que códigos sejam embutidos nos nós finais. Tais modificações têm sido utilizadas historicamente para acelerar o processamento de consultas [34], sistemas de arquivo [35; 36] e o encaminhamento de pacotes [8]. Os reais benefícios deste sistema de roteamento e busca compreendem, portanto, as funções específicas acrescentadas à infra-estrutura DHT base (como poderá ser visto, por exemplo, na multiplicidade de funcionalidades desempenhadas pela rede virtual *overlay* i3).

Por fim, as DHTs de armazenamento, as mais restritivas do ponto de vista da flexibilidade, proporcionam somente primitivas de inserção, recuperação e remoção de dados sem permitir que códigos específicos alterem o curso ordinário de suas operações. Esta falta de flexibilidade, que por um lado limita o número de aplicações suportadas, por outro é compensada por um desempenho superior. Na medida em que a infra-estrutura não precisa se preocupar com códigos alheios, esta pode ser altamente otimizada para executar as funções de armazenamento, como é o caso, por exemplo, da OpenDHT. Além disso, seus usuários, com o propósito específico de armazenamento, encontram grande facilidade de uso dada a especificidade e simplicidade de suas operações.

A busca de informações em redes virtuais depende dos algoritmos de roteamento empregados por sua infra-estrutura. Ao se empregar DHTs como infra-estrutura, cada informação é relacionada a uma chave, e apenas através desta chave ela pode ser recuperada. Cada nó do sistema possui um identificador unívoco. Sempre que uma informação é solicitada, a partir da chave pertinente àquela informação e da função de *hash*, um conjunto de nós ou o nó específico que guarda aquela informação é encontrado através do

roteamento, permitindo o acesso à informação. A função de *hash* empregada varia de uma arquitetura para outra.

Cada nó mantém uma tabela de roteamento que consiste em um pequeno subconjunto de nós do sistema. Quando um nó recebe uma consulta de uma chave pela qual ele não é responsável, este encaminha a consulta para algum nó vizinho para prosseguir a consulta. Esta segue até que se obtenha sucesso em sua localização ou que não seja possível localizá-la, significando que a chave buscada não está presente no sistema.

A seguir serão abordados cinco principais exemplos de DHTs.

2.3.1 CAN

A CAN (*Content Addressable Network*) [21] consiste em uma das quatro DHTs de propósito geral originalmente propostas em 2001 e serve de infra-estrutura de armazenamento e busca de dados para as aplicações P2P. Foi projetada baseada em um espaço cartesiano multidimensional de **d** dimensões. Este espaço de coordenadas é completamente lógico e não possui relação direta com as coordenadas físicas do sistema cartesiano. Seu plano virtual é dinamicamente dividido entre todos os nós de maneira que cada um possua sua própria zona.

Um exemplo de alocação de nós é mostrado na Figura 4 com um espaço de coordenada bidimensional assumindo cada eixo o intervalo Real [0, 1]. Divide-se este plano em 5 nós (A, B, C, D e E) com cada subárea proporcional ao número de chaves armazenadas. Na zona do hipotético nó D, podemos ver que este é responsável por um conjunto limitado e bem definido de chaves no espaço virtual bidimensional, tendo estas, variação de coordenadas no plano no eixo das abscissas de 0,5 a 0,75, e no eixo das ordenadas de 0,5 e 1.

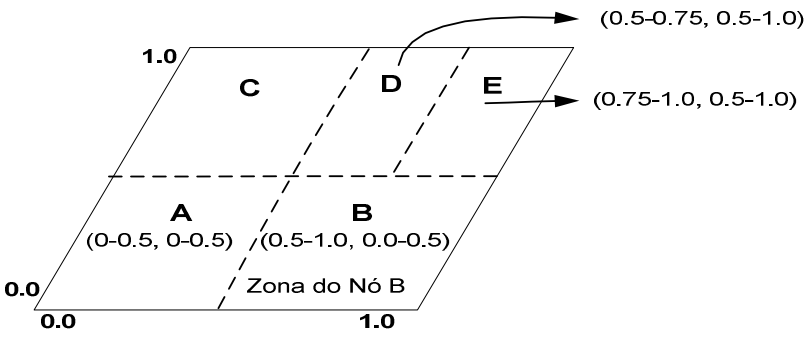

Figura 4. Exemplo de um espaço CAN 2D com 5 nós [21]

Todos os pares (chave, valor) entre os nós participantes da rede são armazenados neste espaço virtual. O mapeamento de uma chave k é deterministicamente localizado em um ponto P do espaço cartesiano, utilizando-se uma função *hash* uniforme. O par (chave, valor) é então armazenado no nó responsável pela zona onde o ponto P se encontra. Desta forma, a busca pelo valor correspondente a uma chave k deve ser feita aplicando-se a mesma função no valor da chave para a descoberta do ponto para o qual a consulta deverá ser direcionada. Se o ponto P não é de responsabilidade do nó, ou de seus vizinhos, a busca deverá ser roteada pela rede CAN até encontrar o nó que possua a zona onde o ponto P se encontra.

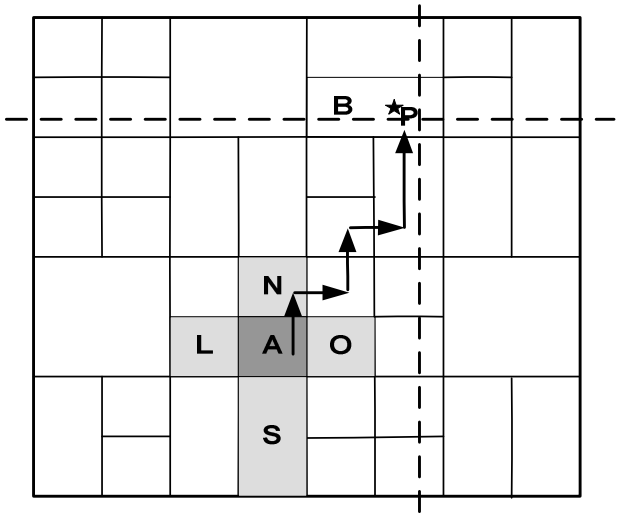

Figura 5. Roteamento CAN no Espaço de Coordenadas 2D [21]

Como a unidade de mensagem CAN possui as coordenadas do destinatário e cada nó mantém uma tabela de roteamento contendo o endereço IP e as coordenadas da zona de cada um dos seus vizinhos imediatos, o roteamento é feito encaminhando-se a mensagem para o vizinho que mais se aproxime da informação buscada, conforme mostra a Figura 5, em que a chave representada pelo ponto P, que está situada na zona no B foi procurada a partir da localidade do nó A. Detalhes do algoritmo de busca desta DHT podem ser encontrados em [21]. Para um espaço com **d** dimensões dividido em **n** zonas iguais, em média são necessários (**d/4**)(**$n^{1/d}$**) saltos e cada nó mantém **2d** vizinhos conhecidos.

2.3.2 Chord

O projeto Chord [22] é baseado em uma DHT robusta de roteamento unidimensional para suportar aplicações P2P. É capaz de suportar **n** usuários participantes, cada um dos quais com um subconjunto de registros armazenados e estar preparado para armazenar bits e dados do índice para a utilização por outros usuários. O endereço IP de cada nó usuário pode ser mapeado para um número de **m** bits através de uma função *hash* consistente, como SHA-1. Desta forma, é possível converter qualquer endereço IP em um número de 160 bits (i.e. o identificador do nó).

Todos os identificadores estão organizados em um espaço de chave circular e neste caso, temos 2^{160} possibilidades de armazenamento consistindo em um espaço amostral gigantesco quando comparado aos atuais 32 bits do endereçamento IP. Devido a sua descentralização e simetria, dados podem ser localizados utilizando somente O(log N) mensagens em média [37], mesmo em face de falhas ou regressos de nós. Uma ilustração deste cenário pode ser observada na Figura 6 com um círculo de identificadores de nós para m = 5, ou seja, 2^5 identificadores. Observe que os nós 1, 4, 7, 12, 15, 20 e 27 correspondem a nós reais e estão sombreados representando o conjunto *overlay* de nós que fazem parte da DHT Chord.

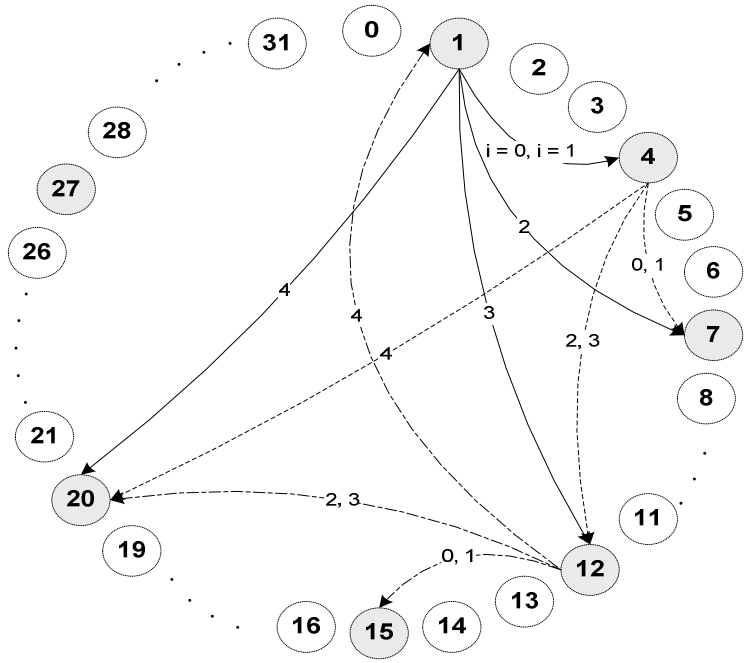

Figura 6. Roteamento em anel DHT com 32 Identificadores [22]

Os identificadores são representados como em um círculo de números (de 0 a $2^m - 1$), então o sucessor(k) é definido como sendo o identificador de nó do primeiro nó real seguinte a k, no sentido horário (veja a Figura 6). Por exemplo, o sucessor(5) = 7, o sucessor(9) = 12 e sucessor(20) = 27. Os índices, ou chaves (nomes de registro), também são mapeados para números através da função *hash* SHA-1 para gerar um número de 160 bits.

Para localizar uma chave utilizando o algoritmo do Chord, o nó solicitante envia um pacote a seu sucessor contendo seu endereço IP e a chave procurada. Este se propaga pelo anel até localizar o sucessor (aquele que possui a chave solicitada) para o identificador que está sendo procurado. Ao ser localizado, o sucessor correspondente devolve as informações diretamente ao solicitante através do seu endereço IP. Em virtude de cada nó dispor obrigatoriamente do endereço IP do seu sucessor e de seu predecessor, as consultas podem ser enviadas no sentido horário ou anti-horário, realizando o menor percurso. Entretanto, este artifício não apresenta grande impacto na redução de buscas, pois a pesquisa continua sendo linear para todos os nós, desempenho este insuficiente em um sistema P2P de grande porte (dado que a ordem de nós pesquisados por consulta seria O(N/2)).

Tabela 1. Definição de Variáveis para um Nó n no Anel DHT

Notação	**Definição**
Finger[k].início	$(n + 2^{k-1}) \bmod 2m,\ 1 \leq k \leq m$

Finger[k].intervalo	[finger[k).início, finger[k+1].início, se $1 \leq k \leq m$ [finger[k].início, n, se k = m
Finger[k].nó	Primeiro nó cujo identificador é igual a ou sucessor de n.finger[k].nó
Sucessor	Sucessor imediato do nó n no anel de identificadores; Sucessor = finger[1].nó
predecessor	Predecessor imediato do nó n no anel de identificadores

Para tornar a busca mais eficiente, o Chord mantém informações adicionais de outros nós, armazenado-as em uma tabela de **m** entradas, denominada de tabela *finger*. Esta mantém os estados das variáveis do sistema que realmente precisam ser armazenados no processo de roteamento e resolução de funções de *hash* para tornar os tempos de busca mais eficientes. Mais precisamente, em uma DHT com N nós participantes, com o auxilio desta tabela, cada nó possuirá informações apenas sobre O(log N) outros nós e será capaz de resolver buscas com O(log N) mensagens de roteamento aos seus vizinhos do anel.

A construção desta tabela é realizada no momento de associação de um novo nó ao anel (e este deve possuir conhecimento de seus vizinhos imediatos) e a cada modificação de sua estrutura, seja por novas associações, modificações ou partidas de nós vizinhos, esta deve ser refeita. No caso da DHT Chord, novas associações ao anel geram $O(\log^2 N)$ mensagens de atualização [22]. Veja as diretrizes para a construção da tabela *finger* na Tabela 1.

$$\text{Início} = n + 2^i \text{ (módulo de } 2^m) \qquad \textbf{(1)}$$
$$\text{Endereço IP do Sucessor (início [i])}$$

Na prática, cada uma das entradas desta tabela nos nós DHT possui dois campos: início e o endereço IP de seu sucessor, como são mostrados na equação eq.(1) (que são obtidos através da Tabela 1). Os valores dos campos correspondentes à entrada **i** no nó **n** são regidos pelas fórmulas encontradas na eq.(1). Cada consulta **i** de uma chave destinada a um nó **n** será efetuada conforme o intervalo em que a chave se encontra, e este é definido pela notação [*finger*[k].intervalo) da Tabela 1 originando as consultas observadas pelas linhas **i** ($0 \leq \mathbf{i} < \mathbf{m}$, ou seja, neste exemplo temos $0 \leq \mathbf{i} < 4$) indicadas na Figura 6.

No exemplo da Figura 6, a tabela *finger* do nó n = 1 armazena os identificadores iniciais finger(k).início, com $1 \leq k < m$, $(1+2^0) \bmod 2^5 = 2$, $(1+2^1) \bmod 2^5 = 3$,

$(1+2^2)$ mod 2^5 = 5, $(1+2^3)$ mod 2^5 = 9 e $(1+2^4)$ mod 2^5 = 17 , respectivamente (observe a Tabela 2(a)). O sucessor do identificador 2 é o nó 4 dado que este é o primeiro nó subseqüente ao 2, o sucessor do identificador 3 também é o nó 4. Consultas realizadas nos intervalos [2,3) e [3,5) realizadas no nó 1 serão remetidas ao sucessor nó 4 conforme pode ser observado em i=0 e i=1 na Figura 6.

Tabela 2. Tabela *Finger Chord* da Figura 6 [22]

i	Início	Intervalo	Sucessor
0	2	[2,3)	4
1	3	[3,5)	4
2	5	[5,9)	7
3	9	[9,17)	12
4	17	[17,1)	20

a) Tabela *Finger* detalhada do nó 1

Início	IP do Sucessor
5	7
6	7
8	12
12	12
20	20

b) Tabela *Finger* do nó 4

Início	IP do Sucessor
13	15
14	15
16	20
20	20
28	1

c) Tabela *Finger* do nó 12

Duas características relevantes podem ser observadas. A primeira delas é que cada nó armazena informações de um número relativamente pequeno de sucessores (log N), sendo que a maior parte desses nós tem identificadores numericamente bastante próximos. Segundo, a tabela *finger* de um nó não contém informações suficientes para determinar diretamente o sucessor de uma chave **k** arbitrária. Por exemplo, o nó 1 na Figura 6 não pode determinar o sucessor da chave 14, uma vez que o sucessor (nó 15) não aparece em sua tabela *finger*.

Utilizando a tabela *finger*, a pesquisa arbitrária de uma chave no nó **n** prossegue da seguinte maneira: Se a chave estiver entre **n** e o sucessor(n), então o nó que contém a informação sobre a chave é o sucessor(n), e a pesquisa se encerra. Caso contrário, a tabela *finger* é consultada para determinar a entrada cujo campo início é o predecessor mais próximo da chave. A seguir, uma solicitação é enviada diretamente ao endereço IP contido nessa entrada da tabela *finger*, solicitando que ele continue a pesquisa. Tendo em vista que cada pesquisa reduz à metade a distância restante até o destino, é possível mostrar que o número médio de pesquisa é $\log_2 N$. Exemplificando, considere outra pesquisa de chave k = 14 iniciada no nó 1. Devido ao fato do 14 não estar entre 1 e 4, a tabela *finger* é consultada na procura do predecessor mais próximo a 14, que é 9, e assim a solicitação é encaminhada ao endereço IP da entrada de 9, isto é, ao nó 12. O nó 12 verifica que o k = 14 está entre ele e o

seu sucessor (ver Tabela 2 (c)), no caso 15, e assim retorna o endereço IP do nó 15 ao solicitante.

Maiores detalhes sobre os procedimentos tanto de busca, como quanto a correção da tabela *finger*, que precisa ser constantemente atualizada pela entrada e saída dos nós podem ser encontrados em [22].

2.3.3 Pastry

No projeto Pastry, uma DHT é implementada com funções especiais de roteamento em um espaço unidimensional de armazenamento. Cada nó da rede possui um identificador (*nodeId*) de 128 bits que pode ser gerado a partir de uma função hash aplicada ao seu endereço IP ou a sua chave pública. O *nodeId* identifica a posição de cada nó em um espaço circular de chaves (variando de 0 a 2128–1, conforme Druschel (2001) [29]) e estas são mapeadas para o nó cujo *nodeId* está numericamente mais próximo da sua identificação. Considerando-se um espaço amostral de **n** nós, Pastry é capaz de encaminhar consultas com ordem de resolução $O(\log_2^b N)$ nós (onde **b** consiste em um parâmetro de configuração pré-definido) [29].

As chaves e o *NodeId* são projetados de forma a otimizar o roteamento de mensagens, de forma que estas representem uma seqüência de dígitos com base 2^b. A cada etapa do roteamento, um nó normalmente encaminha as mensagens para outro nó cujo *nodeId* compartilhe com a chave pelo menos um digito (ou **b bits**) a mais do que é compartilhado com o nó atual. Se nenhum nó é conhecido, a mensagem é encaminhada para o nó cujo *nodeId* compartilha um prefixo com a chave e está numericamente mais próximo da chave do que o presente nó. Para suportar este procedimento de roteamento, cada nó mantém uma tabela de roteamento, um conjunto de vizinhanças e um conjunto de folhas.

As tabelas de roteamento são formada por $|\log_2^b N|$ linhas, cada uma com $2^b - 1$ entradas. Para cada entrada na tabela de roteamento é associado o endereço IP de potenciais nós cujo *nodeId* tem um prefixo apropriado. O conjunto de vizinhança **M** contém os *nodeIds* de |M| nós que estão mais próximos (de acordo com uma métrica de proximidade) do nó local. O conjunto de vizinhança é utilizado no roteamento das mensagens e na atualização das tabelas de variáveis do sistema Pastry.

O conjunto de folhas, **L**, é formado pelos $|L|/2$ nós sucessores e $|L|/2$ nós predecessores mais próximos de um dado nó. Um estado hipotético de um nó Pastry com *nodeId* igual a 10233102 (base 4), em um sistema de 16 bits para identificação e um valor de b = 2 é apresentado na Tabela 3. A tabela interna “Conjunto de Folhas” contêm os nós que

estão mais próximos do nó local e, igualmente, a "Conjunto de Vizinhanças" contém os nós que estão mais próximos do nó local de acordo com a métrica de proximidade Pastry.

Tabela 3. Tabela Pastry de Roteamento do nó 10233102 [29]

Node ID 10233102 (m=16; b = 2)			
Conj. Folhas	**<Menores**	**Maiores>**	
10233033	10233021	10233120	10233122
10233001	10233000	10233230	10233232
Tabela de Roteamento (m/b linhas)			
02212102	1	**2**2301203	**3**1203203
0	1**1**301233	1**2**230203	1**3**021022
10**0**31203	10**1**32102	2	10**3**23302
102**0**0230	102**1**1302	102**2**2302	3
1023**0**322	1023**1**000	1023**2**121	3
10233**0**01	1	10233**2**32	
0		1023312**0**	
		2	
Conj. de Vizinhança		**2^b – 1 entradas por linha**	
13021022	10200230	11301233	31301233
02212102	22301203	31203203	33213321

Entradas na m^{th} coluna possui m próximos dígitos

Entradas na n^{th} linha compartilham os primeiros n dígitos c/ nó atual [*prefix comum* + ***próx dígito*** + *resto*]

Dada uma mensagem qualquer, para realizar o seu roteamento o nó primeiramente checa se a chave está dentro da faixa de endereços contemplados pelo conjunto de folhas e, em caso afirmativo, a mensagem é encaminhada diretamente para o destinatário. Isto quer dizer que existe um nó no conjunto de folhas que está mais próximo da chave pesquisada (possivelmente no presente nó). Em caso negativo, a chave não é encontrada no conjunto de folhas, então a tabela de roteamento é usada e a mensagem é encaminhada para o nó que compartilha um prefixo comum com a chave (pelo menos um dígito). Em certos casos, é possível que o nó associado não esteja alcançável. Neste caso, a mensagem é encaminhada para um nó que compartilhe um prefixo com a chave e esteja numericamente mais próximo da chave do que o nó atual.

Quanto à manutenção das tabelas de roteamento em virtude da entrada e saída de vizinhos, uma função heurística é aplicada para garantir que as entradas na tabela de

roteamento são escolhidas de maneira adequada (i.e. elas fornecem uma boa localização dos nós). Maiores detalhes podem ser obtidos na documentação original do projeto disponível em [29].

2.3.4 Tapestry

Objetivando prover uma rede virtual *overlay* de roteamento *multicast*, Tapestry [23] apresenta diversas similaridades ao Pastry em relação à sua infra-estrutura DHT. Igualmente aqui, é utilizado o conceito de prefixo/sufixo no roteamento, bem como algoritmo de entrada e saída dos nós e, também, é levado em consideração o custo de armazenamento de dados.

Diferentemente do Pastry, o Tapestry não utiliza conjuntos de nós folhas e de vizinhos. Quando a tabela de roteamento de um nó não possui uma entrada para um nó que compartilhe um sufixo comum com a chave (pelo menos um ou mais dígitos), a mensagem é encaminhada para um nó que está numericamente mais próximo da chave do que o nó

atual. O número de saltos esperados é $O(\log_{16}N)$ [33].

Participantes da rede DHT podem publicar seus objetos periodicamente através do roteamento e envio de mensagens para o nó raiz. Neste processo, cada nó no caminho armazena um ponteiro de mapeamento do objeto. Múltiplos servidores podem publicar ponteiros para o mesmo objeto e na ocorrência de *links* redundantes estes serão priorizados pela latência e/ou localidade. Finalmente, objetos são então localizados através do roteamento de mensagens em direção à sua raiz. Para isto, cada nó intermediário faz a verificação do mapeamento alterando-o conforme necessário em um processo convergente até sua localização.

2.3.5 O PlanetLab e sua Tabela Hash, o OpenDHT

O ambiente de PlanetLab [38] oferece aos seus usuários diversos serviços inovadores, cada qual sendo executado em um ambiente próprio. Estes serviços visam, tanto oferecer suporte aos experimentos em execução em outros ambientes, como oferecer serviços distribuídos em escala global para usuários internos e externos do PlanetLab. Um exemplo destes serviços consiste no OpenDHT [30], um serviço de Tabela *Hash* Distribuída disponibilizado publicamente em mais de 300 nós do PlanetLab (dados de Agosto de 2007).

O PlanetLab consiste de uma Rede Virtual *Overlay* que provê uma infra-estrutura distribuída de bancada de teste para a execução de experimentos de campo na

Internet. Através do espalhamento geográfico de múltiplos computadores em escala global, é dada aos pesquisadores a oportunidade de alocar recursos (de comunicação e processamento) em vários sistemas, permitindo que os mesmos executem os seus experimentos sobre a topologia real da Internet, com a flexibilidade provida pela disponibilidade de computadores (em verdade, máquinas virtuais) em diversos pontos desta topologia.

Deste modo, o PlanetLab permite a contemplação de anseios de duas comunidades bem distintas de "clientes": os pesquisadores que almejam um ambiente para implementação de novas aplicações e serviços (flexibilidade em detrimento da disponibilidade) e usuários finais, tencionando o uso de serviços inovadores. Tal sistema permite aos pesquisadores um ciclo de maturação bastante natural para suas aplicações, protocolos e tecnologias até a sua disponibilização ao público em geral (i.e. usuários da Internet). Permite que o experimento transcorra em uma ambiente real, uma rede global sujeita a falhas, congestionamentos e picos de utilização, com uma carga de trabalho também real, provida por uma comunidade crescente de usuários.

Com efeito, pode-se considerar o PlanetLab hoje não apenas como a bancada de testes distribuída que lhe motivou a criação, mas também como a infra-estrutura que abriga diversos serviços inovadores, como aqui ilustrado com o OpenDHT, confirmando de maneira irrefutável o acerto do abarcamento de pesquisadores e usuários finais em um mesmo projeto.

O eixo condutor da estratégia de criação do PlanetLab assenta-se sobre três pilares conceituais, definidos descritivamente como três "dimensões" em [12]: a dimensão física (i.e. um *overlay* da ordem de centenas de nós), a dimensão componencial (i.e. a presença de um monitor de máquinas virtuais em cada nó físico e serviço de gerenciamento global) e a dimensão operacional (provimento incremental de funcionalidades fazendo uso da Internet corrente como substrato de comunicação).

Cada uma destas dimensões, ou conceitos-mestre, define um aspecto que se mostra crucial na obtenção das funcionalidades almejadas. A significância, a capacidade de prover um ambiente experimental com resultados "convincentes", está diretamente relacionada à escala (neste caso, da ordem de centenas de nós). Já o uso de técnicas de virtualização, aliadas a um monitoramento distribuído e gerenciamento centralizado, permite um alto grau de compartilhamento e simultaneidade que aumentam a diversidade do "ecossistema" de usuários e aplicações sob estudo.

Finalmente, as duas características anteriores, quando aliadas a um modelo de introdução gradual de novas funcionalidades, com a explícita permissão de aumento gradual da base de usuários, tornam o PlanetLab uma atraente plataforma de implementação, bem

como de fornecimento de serviços. Diante dos benefícios supracitados e da capacidade de tornar real, do ponto de vista de escala, utilização e susceptibilidade a falhas qualquer experimento de larga escala, optou-se pela utilização do OpenDHT neste projeto.

2.3.5.1 A Arquitetura do OpenDHT

O OpenDHT consiste em serviço largamente disponibilizado através da infra-estrutura do PlanetLab que implementa a DHT Bamboo [39]. Primando pela simplicidade e desempenho, esta DHT desobriga seus usuários da penosa tarefa de instalação de uma instância local da infra-estrutura para sua utilização. Como uma DHT, primariamente de armazenamento, seu acesso pode ser feito através de uma API pública baseado em chamadas de inserção (*put*), recuperação (*get*) e remoção (*rm*) que processam as operações e armazenam os dados nos diversos nós ativos. Cada nó armazena uma porção do total de dados guardados na OpenDHT e é capaz de encaminhar requisições aos seus pares DHT até que a chave de busca seja localizada. Procurando suportar um mecanismo de busca similar ao fornecido nas DHTs daquela categoria, uma biblioteca denominada ReDiR (*Recursive Distributed Rendezvous*) foi disponibilizada para seus usuários, mas este não será o foco deste trabalho.

Operando essencialmente como uma DHT de armazenamento, para manter seus dados em sistema, seus clientes devem realizar inserções com freqüência inferior ao TTL (*Time-To-Live*) definido (que consiste em uma função diretamente relacionada ao espaço disponível). Atualmente (junho, 2007), o armazenamento se mantém ativo por até 604.800 segundos (ou uma semana). Além disso, nenhuma inferência sobre operações passadas deve influenciar futuras inserções, proporcionando potencial equivalente para entrantes e usuários ativos sendo estes limitados somente por uma possível ausência de espaço.

Finalmente, para prevenir estados de *starvation* (situação em que alguns clientes nunca conseguem armazenar seus dados) a OpenDHT deve garantir uma taxa mínima constante de inserções para evitar situações de rajadas de armazenamento seguidas por períodos sem nenhuma inserção.

Em relação à sua interface, está é igualmente acessível por chamadas Sun RPC (*Remote Procedure Calls*) sobre TCP ou XML RPC sobre http sendo, portanto, de fácil acesso por qualquer linguagem de programação, mesmo dentro de redes privativas com mecanismos de NAT e/ou *Firewalls*. Uma lista de servidores ativos pode ser encontrada em [40]. Além disso, nenhuma arbitragem é feita sobre a escolha das chaves, que podem atingir até 20 bytes e serem associada a valores de até 1024 bytes (entretanto, valores maiores podem ser armazenados na forma de vários blocos menores conectados pela mesma chave).

Finalmente, a OpenDHT proporciona um método de autenticação opcional com um segredo (de 40 bytes máximos) que garante a integridade dos valores armazenados impedindo sobrescritas acidentais e prevenindo ataques desta natureza. Para a remoção dos valores armazenados, então, um acesso seguro baseado no algoritmo SHA-1 foi disponibilizado. A Tabela 4 abaixo sumariza as três primitivas básicas para a interface Put/Get utilizada neste projeto, descritas nas colunas Primitivas e Funcionalidades.

Tabela 4. Interface Put/Get com H(s) Representando o SHA-1 de 's'.

Primitivas	Funcionalidades
put(k, v, H(s), t)	Insere a tupla (k, v) com um TTL t permitindo a futura remoção com o segredo s.
get(k) retorna {(v, H(s), t)}	Leitura dos valores (v) armazenados sob a chave (k) sem segredo de autenticação associado.
remove(k, H(s), s, t)	Remove a tupla (k, v) com o segredo s e tempo restante t.

Como pôde ser observado, as chaves são aqui referenciadas pela consoante **k** e segredo obtido pela função SHA-1 é denotado pela consoante **H**. Os valores associados às chaves foram representados pela letra v e o TTL pela consoante t.

Este modelo de serviço/interface simplifica significativamente o desenvolvimento das aplicações clientes que podem utilizar os benefícios de um serviço de armazenamento e nomeação extremamente confiável, distribuído e com alta disponibilidade.

2.3.5.2 O Algoritmo de Alocação do OpenDHT

Garantir que o armazenamento de nomes será sempre possível compreende uma tarefa imprescindível a ser desempenhada adequadamente pela DHT candidata, haja vista que em um ambiente como a Internet, a disponibilidade da infra-estrutura de resolução de nomes representa um requisito imperativo.

O algoritmo aqui utilizado, o FST (*Fair Space-Time)* [30] deve garantir que períodos de rajadas de inserções não sejam seguidos por períodos com negação de serviço aos seus clientes. Entende-se por clientes aqui, usuários associados aos seus endereços IP. Ainda que esta seja uma consideração primariamente simplista, alguns mecanismos de verificação da origem (como o *TCP's three-way handshake [41]*) são utilizados. Entretanto, algumas conseqüências são notórias: usuários utilizando NAT ou *firewalls* comuns podem vir a competir entre si por armazenamento; usuários móveis podem ser beneficiados com maior

capacidade e clientes com endereços classe A poderiam “virtualmente” obter armazenamento quase que ilimitado.

Na prevenção de situações de *starvation*, taxas mínimas de chamadas put são necessárias para garantir que inserções não serão rejeitadas. Sem esta abordagem, todo o espaço disponível poderia ser consumido no intervalo de um TTL seguido pelo mesmo intervalo sem nenhuma inserção, o que não é aceitável para um ambiente como a Internet.

Desta forma, tomamos um intervalo T em segundos a que todos devem se submeter e um montante B em bytes que representa todos os *puts* (i.e. as inserções) feitos neste intervalo. A taxa mínima que todos os nós da DHT devem aceitar inserções é, então, r_{min} = C/T, onde C representa a capacidade do disco (veja o gráfico da Figura 7).

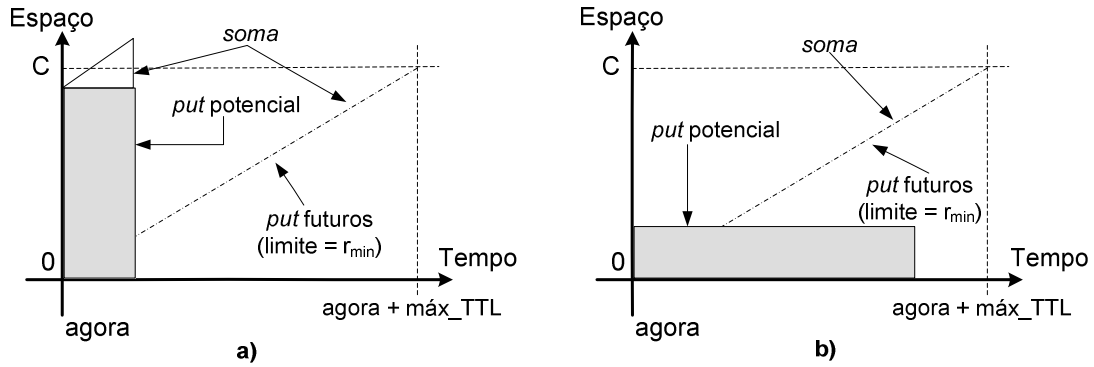

Figura 7. Prevenindo Starvation no OpenDHT

Ao considerar novas inserções, o FST deve determinar se tal aceitação irá impactar na capacidade futura de novas inserções observando os valores r_{min} e B. Uma exemplificação pode ser observada na Figura 7 que apresenta a taxa de aceitação versus o espaço em disco. O espaço mínimo r_{min} reservado para inserções futuras pode ser observado na linha tracejada (cujo limite superior é r_{min}). Supondo dois comportamentos de inserções, um mais enérgico (em termos de número de bytes) com um TTL curto, conforme a Figura 7(a), e outro mais ameno com um TTL maior, conforme a Figura 7(b), o comportamento OpenDHT esperado é ilustrado.

Requerer que os *puts* em andamento não ameacem a taxa mínima de reserva futura (r_{min}) é graficamente equivalente a observar se a soma da linha $y = r_{min}x$ não excederá a capacidade C de armazenamento em quaisquer pontos futuros. Pode-se observar que no caso Figura 7(a) esta condição é violada, enquanto que no caso Figura 7(b) o sistema se comporta como esperado.

Baseado nesta analise gráfica intuitiva, uma função de admissão mais complexa foi derivada. Sendo B(t) o número de bytes armazenados no sistema em um tempo t

e sendo D(t1, t2) o número de bytes no intervalo [t1, t2) (limitado pelo TTL do sistema), para qualquer ponto no tempo (chamado t_{agora}) é possível estimar o número total de bytes, $f(\tau)$, armazenados no sistema em um tempo $t_{agora} + \tau$ assumindo que novas inserções continuarão a ocorrer com uma taxa mínima r_{min} através da equação eq.(2).

$$f(\tau) = B(t_{agora}) - D(t_{agora,}\, t_{agora} + \tau) + r_{min} * \tau \qquad \textbf{(2)}$$

Os dois primeiros termos representam a capacidade acordada de armazenamento que será disponibilizada no disco no tempo $t_{agora} + \tau$. O terceiro termo representa a mínima taxa de armazenamento que o algoritmo FST irá garantir no intervalo $(t_{agora,}\, t_{agora} + \tau)$.

Considerando a hipótese de uma nova inserção de tamanho x e TTL l chegar no tempo t_{agora}, esta só será aceita se, e somente se, as condições da equação eq.(3) abaixo forem satisfeitas em todo o intervalo $0 \leq \tau \leq l$:

$$f(\tau) = x \leq C \qquad \textbf{(3)}$$

Se a inserção for aceita, a função $f(\tau)$ será atualizada. Como a inspeção detalhada dos parâmetros matemáticos envolvidos na análise de *starvation* não representa o objetivo principal deste trabalho, nos deteremos aqui no tempo gasto para cada atualização, que apresenta tempo logarítmico em função do número de inserções aceitas (quando da observação dos limites da função $f(\tau)$ utilizando uma árvore balanceada conforme mostrado por [30]). Tempos de resolução da ordem O(log n) para um armazenamento de O(log n) não são satisfatórios para um sistema de resolução de nomes, de forma que, esta arquitetura de nomes terá que utilizar alguns artifícios para contornar tal problema, conforme será discutido nos Capítulos 5 e 7.

2.4 Proxies

Proxies, ou procuradores, consistem em *middleboxes* que interconectam computadores em uma rede "restrita" através do compartilhamento de sua conexão com as demais máquinas. Assim, toda solicitação de conexão de uma máquina da rede local (i.e. da rede restrita) para um nó da rede externa (i.e. da Internet ou de outra rede) é direcionada ao *proxy*, este, por sua vez, realiza a conexão com o nó desejado, repassando a resposta à solicitação para a máquina da rede local.

Um *proxy* é, portanto, um software capaz de armazenar dados, alterando-os, ou não, para serem disponibilizados para os demais computadores em redes por questões de desempenho, segurança, acesso entre outras. Diversos tipos de *proxies* podem ser encontrados hoje [1], como por exemplo, os *Web Proxies* (que inspecionam os pacotes HTTPs modificando-os), HTTP *Caching Proxy* (um caso particular dos *Web Proxies* que alteram o fluxo IP normal substituindo-o adequadamente), Performance *Enhancing Proxy* (que visam melhorar o desempenho das consultas), SIP *proxies* (que atua como um tradutor SIP de tráfegos multimedia), *Transparent Proxies* (que interceptam os tráfegos de forma transparente) e etc.

O *Web Proxy*, classe particular de *proxy* empregado neste projeto, provê um cachê de páginas da Internet que são disponibilizados para clientes HTTP da rede local. Ao requisitar um documento na *World Wide Web*, o cliente conta com o *proxy* para localizá-lo, ou seja, fazer a consulta local (em *cache*) ou remota (procura pelo seu IP em servidores DNS e então obtêm os dados) para disponibilizar o documento procurado e salvar uma cópia local. Neste trabalho, utilizou-se o *Web Proxy Scone*. Maiores detalhes serão apresentados adiante.

2.4.1 Framework Scone

O *Framework Scone* [42] consiste em uma ferramenta baseada na linguagem Java disponibilizada sob a licença GNU-GPL [43] com o objetivo de permitir um rápido desenvolvimento de novas aplicações *Web*. Sua arquitetura modular oferece diversos componentes que podem ser utilizados para modificar ou melhorar aplicações *Web* através da inserção de *plugins* (i.e. módulo de software adicionado a um programa principal para desempenhar funções específicas). Neste trabalho, utilizou-se o *Web Proxy Scone* e o WBI *plugin* adicionando-se códigos específicos a ele.

Scone apresenta uma arquitetura modular com quatro bases principais e diversos componentes que podem ser combinados para o desenvolvimento de novas ferramentas *Web*. São eles:

- **Proxy Programável:** Prove interfaces de acesso e manipulação de documentos. É baseado no IBM WBI (*Web Based Intermediary*) e utiliza o conceito de *tokenstream* e orientação a objetos para simplificar e acelerar o processamento de dados. O WBI oferece uma plataforma *proxy* analítica de desempenho com diversas ferramentas de depuração de código que pode ser usada livremente em ambientes de pesquisa.

- **NetObjects**: Apresenta uma base de armazenamento Web para dados como os URI (*Uniform Resource Locator*), documentos HTML, *links*, servidores, usuários e suas atividades. A criação e acesso a estes objetos pode ser utilizada na criação de eventos para as classes observadoras através do mapeamento realizado pelo *object-to-RDB* e *caching*.
- **Robô:** Esta ferramenta auxilia a indexação de dados *Web* sendo capaz de percorrer os diversos conteúdos (i.e. páginas, arquivos, diretórios e etc) para a formação de uma base de dados. Utiliza um filtro de classificação (*classifier-filter-concept*) que permite categorizar *links* de documentos baseado em seus atributos.
- **Rastreamento de Acesso:** Este módulo Scone admite o rastreamento das ações dos usuários na utilização de seus navegadores *Web*, permitindo que diferentes tipos de aplicações gerem eventos customizados baseados no comportamento de navegação.

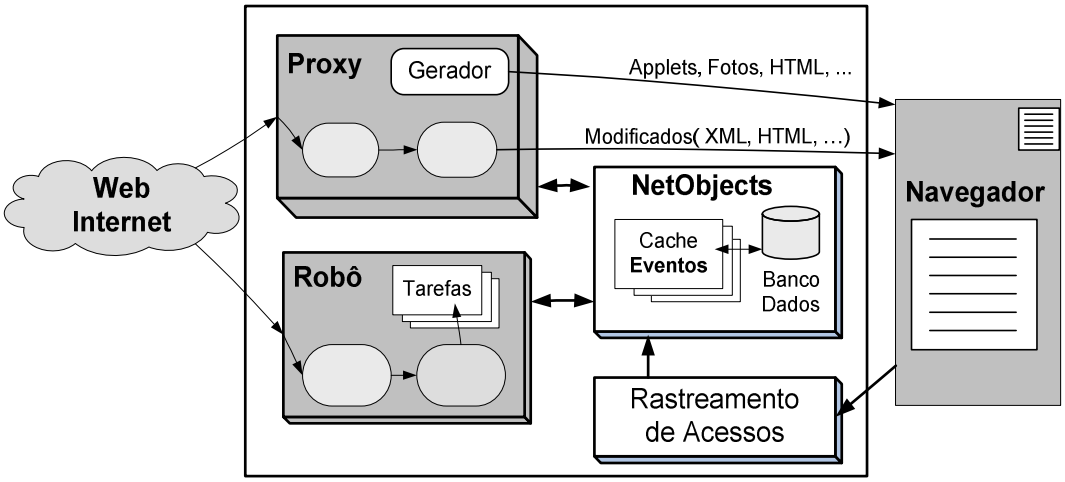

Figura 8. Arquitetura do Proxy Scone

Uma ilustração dos componentes base e seus relacionamentos pode ser vista na Figura 8 abaixo. Dada uma rede local com seus clientes utilizando os serviços de *proxy Scone* via seus Navegadores, este possibilita acesso a diversos serviços ordinários (e.g. acesso *Web*) e customizados (e.g. páginas HTML/XML customizadas, um novo espaço de nomes e etc..) e é capaz de armazenar informações fundamentais para uma boa análise comportamental dos clientes através das ferramentas de rastreamento.

A adoção do *Proxy* Scone neste trabalho se deu basicamente pela utilização pelo mesmo de uma linguagem altamente portável e universal, o Java, em sua concepção e pelas grandes vantagens proporcionadas pelo mesmo ao apresentar um *framework* altamente modularizado. O módulo *Proxy* Programável foi modificado com a adição de um *plugin* para

suportar dois espaços de nomes, um hierárquico representado pelos nomes DNS e outro sem hierarquia, representado pelo novo espaço de nomes proposto neste projeto.

CAPÍTULO 3 PROPOSTA DE UMA TAXONOMIA DE REDES VIRTUAIS

Nos seus primórdios, a Internet compreendia uma rede de pesquisa altamente flexível e aberta para realização de experimentos que poderiam causar-lhe mudanças arquiteturais [44]. No cenário atual, esta se apresenta altamente engessada, principalmente no que tange ao seu foco comercial, em que a Internet representa um ambiente em expansão constante e público (largamente difundido pelo termo de "Internet 1"). Mesmo diante da presença da "Internet de pesquisa", ou Internet 2, este cenário não se apresenta menos rígido dada a multiplicidade de pesquisas utilizando a rede como infra-estrutura ao invés de como sujeito da pesquisa [44].

Em busca da solução para a diversidade de problemas arquiteturais que afligem a Internet desde sua concepção, duas principais abordagens são encontradas: a purista e a pluralista [44], sendo a primeira associada com mudanças fundamentais e a segunda a mudanças incrementais. Ainda que mudanças fundamentais produzam melhores resultados do ponto de vista arquitetural devido à maior flexibilidade, duração além da possível intensidade das alterações, esta solução no estado de arte atual não representa uma alternativa factível, ainda que do ponto de vista tecnológico seja possível [12]. Como ilustração deste cenário, pode-se citar a vagarosa adoção do IPv6 [6] e a relativa baixa adoção de técnicas de roteamento *multicast* entre os provedores de serviço e usuários.

Como resultado das análises realizadas neste trabalho sobre diversos projetos de Redes Virtuais e da necessidade de formalizar termos que permitissem sua classificação e conseqüente comparação foi proposta uma Taxonomia de Redes Virtuais. Detalhes deste trabalho podem ser vistos em [45], contudo, a apresentação de seus termos fundamentais será realizada a seguir para o melhor entendimento do leitor, haja vista que estes serão utilizados com exaustão durante todo o texto.

3.1 Redes Virtuais

Em face da ossificação da infra-estrutura da Internet, a maneira remanescente de introduzir melhorias em sua arquitetura consiste na utilização de mudanças incrementais, como demonstraram diversas pesquisas pluralistas nesta direção [12; 46; 16; 44]. Considerando este tipo de abordagem, uma solução consagrada [12] consiste no conceito da

virtualização. Entende-se por virtualização, em sua forma mais ampla, uma abstração de alto nível capaz de ocultar dos usuários detalhes de implementação [44]. Incorporando este princípio na proposição de soluções arquiteturais para a Internet, tem-se o conceito de **Redes Virtuais** cujos propósitos, objetivo e funcionalidade são tão variados quanto complexos.

O aumento do número de nós na Internet capazes de despenhar funções especiais (i.e. novas funcionalidades dantes não suportadas) habilita aos serviços, usuários e dados neles residentes a conectarem-se mutuamente através desta nova infra-estrutura modificada formando uma nova Rede Virtual de propósitos e características arbitrários. O mapeamento destes novos indivíduos comunicantes estabelece um modelo para a introdução de novas funcionalidades e características originais podem ser definidas [12; 44].

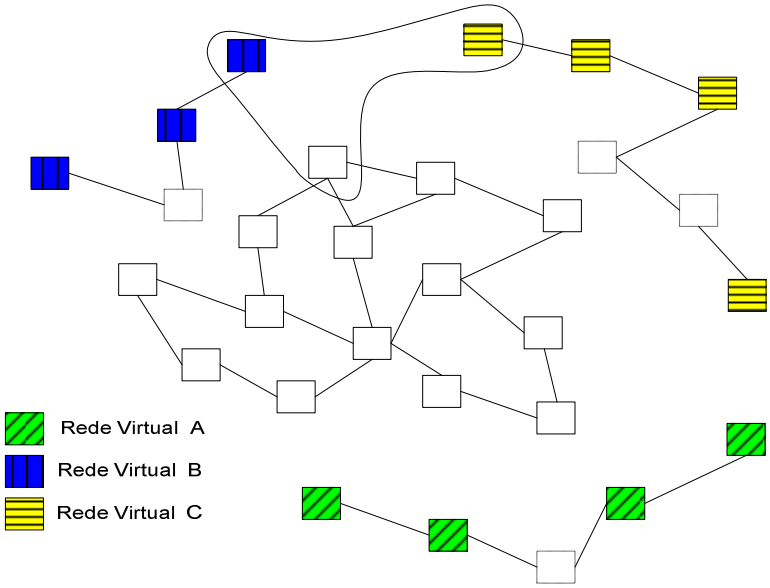

Figura 9. Múltiplas Redes Virtuais Operando Simultaneamente.

Neste cenário, é possível encontrar situações em que um nó pode pertencer simultaneamente a diversas Redes Virtuais, fato que possibilita o aumento da capacidade de introdução de novas funcionalidades. A utilização de um mesmo nó para diversos propósitos concorrentes (i.e. aquele que participa de diferentes Redes Virtuais) operando simultaneamente define um ambiente não só de competição de recursos (e.g. poder computacional, largura de banda, aplicações, etc), mas também, que claramente permite a agregação de funções complementares (e.g. os projetos i3[8] e HIP [11] que combinados originaram um terceiro projeto ainda mais robusto, o Hi3 [47]).

Na Figura 9, algumas possibilidades de configuração de Redes Virtuais, com nós pertencentes a múltiplas Redes Virtuais, podem ser observadas. Nela são representadas 3

redes virtuais, A, B e C e a rede substrato central. É possível observar, através da ilustração que uma terceira e nova rede virtual pode ser formada pela composição das Redes Virtuais B e C. Os benefícios determinísticos esperados com este tipo de composição são a pluralidade de funções advindas de cada Rede Virtual. Contudo, é preciso observar nesta composição os prováveis impactos sofridos no desempenho e possíveis inesperadas brechas de segurança.

Para um completo entendimento dos elementos que compreendem as Redes Virtuais, da relação entre eles (de contenção ou desvinculação) e papel desempenhado, uma breve descrição de cada qual será realizada a seguir almejando aumentar o entendimento do leitor.

3.1.1 O Transporte de Dados e *Metadados*

O estabelecimento de uma analogia entre os conceitos de dado e *metadado* e suas contrapartes aplicadas no ambiente de Redes Virtuais se faz necessário. Ainda que ambos sejam transportados pela mesma infra-estrutura, suas naturezas essenciais e propósitos são completamente díspares. O metadado descreve o conteúdo, a qualidade, a condição além de outras características do dado, aumentando o conhecimento do subsistema sobre o elemento manipulado. Geralmente, os *metadados* são utilizados para organizar as propriedades do dado, para fornecer informações a máquinas de busca (i.e. *search engines*), como catálogo, para declarar a propriedade de dados e para auxiliar a transferência dos mesmos [45].

Tal definição foi embasada no reconhecimento e aceitação dos paradigmas de redes de dados e de *metadados* (conceitos amplamente conhecidos, porém não apropriadamente considerados por arquitetos de protocolos de redes) [48; 49; 45]. Desta forma, pode-se interpretar o conceito de nomes e endereços (ressalvas devem ser feitas quanto a errônea interpretação destes dois como um mesmo conceito), bem como o de rotas (i.e. diretrizes de como alcançar os dados) como *metadados*, enquanto o conteúdo propriamente dito, transportado pelos pacotes, como o dado.

3.1.2 O Conceito de *Overlays* e *Underlays*

Um segundo conceito fundamental para a definição da presente taxonomia consiste no método de construção das redes virtuais. Cada qual em seu processo de formação requer a inserção de nova(s) funcionalidade(s), um novo protocolo ou função em alguma camada do modelo de referência OSI. Isto posto, dois tipos de nós podem ser distinguidos em sua formação.

Primeiramente, há aqueles que não sofrem modificações em sua pilha de protocolos de rede e que constituem os nós intermediários responsáveis por interconectar os nós "modificados". Há também outros nós que serão responsáveis pela introdução de mudanças, constituindo a Rede Virtual, e representando os elementos que irão interagir e utilizar as novas funcionalidades e aplicações advindas da virtualização. Entende-se que no cenário atual da Internet, os nós não modificados representam os roteadores que compõem a infra-estrutura de interconexão das redes virtuais, uma divisão que classifica tais redes pela sua localização em uma das camadas do modelo OSI em que a pilha de protocolo não é modificada.

Diante desta elucidação, algumas conclusões podem ser feitas. Quanto à introdução de modificações na pilha de protocolos abaixo da camada de rede da Internet (i.e. camada IP) esta implicará, forçosamente, na inserção de algoritmos de roteamento (i.e. artifícios de seleção de caminhos) na infra-estrutura de redes virtuais para a entrega de dados. Contudo, poucos roteadores (i.e. nós intermediários) na Internet são capazes de realizar algum processamento nas camadas abaixo de rede (do modelo TCP/IP) implicando, portanto, em limitações de expansão topológica e funcional. Em oposição, redes virtuais que introduzem mudanças na pilha de protocolos acima da camada de rede, sendo encapsulada por esta ao invés de encapsulá-la, estarão limitadas em seu desempenho e na capacidade de decisão de encaminhamento de pacotes desempenhada pelos roteadores da rede substrato. Pode-se concluir que este último cenário, apesar de não contemplar nenhuma inferência no modelo de roteamento consagrado há décadas pelo modelo TCP/IP não estará diretamente limitado em sua expansão topológica ainda que apresente restrições quanto às funcionalidades implementadas do ponto de vista do seu desempenho frente ao modelo atual.

As duas definições apresentadas foram cunhadas pelos termos de Redes Virtuais *Overlay*, aquelas que introduzem modificações acima da camada de rede, e Redes Virtuais *Underlay*, aquelas que modificam funções abaixo da camada de rede [45]. A coexistência destas duas classes é perfeitamente compreensível e esperada na Internet dada a multiplicidade de possíveis soluções. Na Figura 10 ilustra-se a presença de nós membros de uma ou mais Redes Virtuais; uma rede *Underlay* (Rede C) e uma rede *Overlay* (Rede B), bem como a camada modificada no modelo TCP/IP por cada solução.

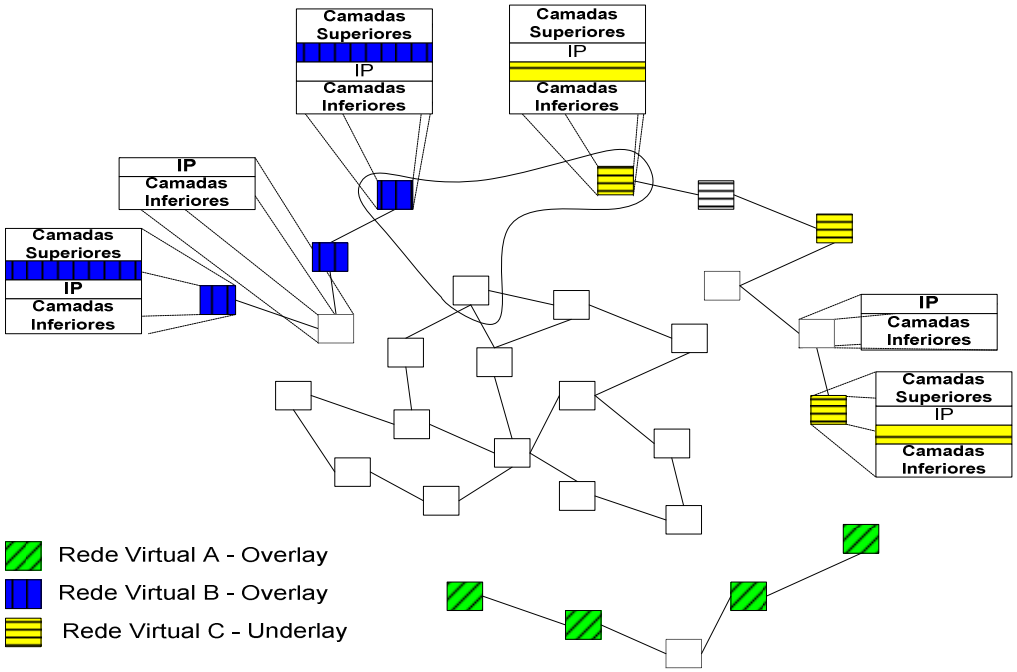

Figura 10. Redes Virtuais *Underlay* e *Overlay*

3.1.2.1 Redes Virtuais *Overlay*

Redes *Overlay,* um exemplo específico de redes virtuais, e candidatas alvo deste projeto de pesquisa, são implementadas acima da camada de rede do modelo TCP/IP (igualmente compostas pela agregação de diversos *hosts* previamente selecionados pertencentes a uma ou mais redes base de computadores). Estes computadores "específicos" (não no sentido de especializados) são habilitados a desempenhar novas funcionalidades aprimorando a rede base.

Um exemplo clássico de Redes *Overlay* trata da Internet nos seus primórdios: uma Redes *Overlay* de computadores selecionados (e.g. *hosts* equipados com *modems*) comunicando-se através da infra-estrutura pública de telefones PSTN (*Public Switched Telephone Network*). Através dela, redes locais construídas sobre *Ethernet* ou linhas telefônicas são conectadas e cabeçalhos são adicionados aos quadros de enlace formando uma nova rede. Um segundo exemplo, o das redes P2P (*Peer-to-Peer*), consiste em redes *Overlay* de aplicações específicas, por exemplo, de armazenamento descentralizado capazes de prover comunicação em grupo, distribuição de conteúdo e alta disponibilidade mesmo diante da falha de vários *peers overlay.*

Como exemplificado anteriormente, Redes Virtuais *Overlay* estaticamente configuradas não representam nenhuma novidade. Diversas redes *overlay* para diferentes

propósitos [50] já foram implementadas, para prover conectividade de enlace para redes OSI [51], facilitar o uso de *multicast* IP [52] e prover conectividade IPv6 [2].

Redes *Overlay* geralmente definem um endereçamento, nomeação, roteamento, QoS e um modelo de serviço para a comunicação entre as entidades pertences à rede virtual, designados para melhorar o desempenho, aumentar a confiabilidade ou a segurança da rede base. Entretanto, a inserção de novas características não dantes suportadas pode acarretar mudanças não previstas na rede base. Um exemplo a ser citado, consiste na implantação de redes *overlay* i3 sem o projeto *Secure*-i3 [53] que foi posteriormente projetado para corrigir falhas de segurança advindas de sua utilização.

De maneira geral, pode-se vislumbrar uma série de benefícios e desvantagens destas Redes Virtuais.

Possíveis benefícios são:

- Sua adoção não implica na necessidade de aquisição de novos equipamentos ou na modificação dos *softwares/hardwares/protocolos* já existentes, pois as mudanças propostas são incrementais.
- Elas geralmente não requerem que sua adoção seja feita na completude de equipamentos da rede base, podendo ser feita somente em *hosts* estratégicos (opcional).
- Suas novas características aperfeiçoam a rede substrato na medida em que inserem características "avançadas", tais como mobilidade, segurança ou *multicast*.

Possíveis desvantagens são:

- A adição de *overhead* (dado que a inserção de uma nova camada na pilha TCP/IP geralmente significa a adição de novos cabeçalhos, processamento adicional, gerenciamento, conversões e etc.).
- Adicionam complexidade ao modelo atual.
- Muitas vezes não são capazes de expandir-se de maneira adequada.
- Sua adoção pode apresentar problemas de segurança não previstos (e.g *Secure-i3*).

3.1.2.2 Tipos Específicos de *Overlays* e *Underlays*

Diante da diversidade de características factíveis de serem habilitadas por redes virtuais O*verlay* e *Underlay* além da variedade de projetos encontrados na literatura destas redes [7; 54; 8; 10; 11; 47; 14], faz-se necessária uma análise cautelosa para identificar quais os princípios que são desejáveis de serem implementados na proposta deste trabalho. Uma

série de aplicações pode ser combinada para a composição da nova arquitetura. Alguns exemplos são:

- **Nomeação:** Possui a premissa básica de estabelecer mecanismos para reger a nomeação das entidades componentes do *overlay*. Para isso, precisa definir um novo espaço de nomes (seja ele hierárquico, misto ou sem hierarquia), os objetos alvo da classificação, o serviço de resolução de nomes entre outras características. Exemplo: Projeto SFR [14].
- **Roteamento**: Consiste em redes que não utilizam, ou utilizam parcialmente, os serviços já implementados pela camada de redes. Seu modelo pretende alterar completamente (e.g. *source route*) ou parcialmente (e.g. *loose source route*) as técnicas de roteamento, substituindo-as por outras presentes na rede *overlay*. Exemplo: Projeto i3 [8].
- ***Multicast/Anycast*:** Sua principal função é permitir que a rede substrato suporte a comunicação *multicast/anycast* [33] conseguindo, portanto, efetuar a entrega dos pacotes IP para múltiplos destinatários previamente selecionados. Exemplo: Projeto i3.
- **Mobilidade:** Objetiva capacitar os *hosts* a se moverem de maneira natural. Entende-se por isso, que dada uma conexão estabelecida entre entidades pares, a migração de uma delas para outro domínio (*single jump*) ou de ambas simultaneamente (*double jump*) não deve ocasionar a perda da comunicação estabelecida. Exemplo: Projeto i3 com uso de servidores *rendezvous* [8].
- **Segurança:** Trata-se de uma rede *overlay* de aplicação específica que procura proteger a rede base contra algum(ns) tipo(s) de ataque(s) específico(s) através do uso de técnicas preventivas (a exemplo do uso de autenticação e criptografia entre entidades comunicantes) ou mesmo corretivas (a exemplo da capacidade de remoção do identificador (i.e. *trigger)* de uma entidade maliciosa, por parte da vítima, em um ataque de DoS [13] em uma rede i3). Exemplo: *Secure*-i3 [53].
- **QoS*:*** Compreende uma Rede Virtual confeccionada para habilitar recursos de Qualidade de Serviço (e/ou de Conteúdo) em uma rede substrato (e.g. a Internet) permitindo que esta aumente seu conhecimento sobre as propriedades (e.g. latência de um link, carregamento de um servidor, reputação de um proprietário e etc) de cada um de seus elementos (e.g. dados, serviços e usuários). Exemplo: *OverQoS* [55].

Os exemplos acima ilustrados são frutos da análise de campo de diversos projetos relacionados no capítulo quatro deste trabalho e, portanto, estão baseados nos exemplos encontrados na literatura, não almejando, desta forma, representar uma análise absoluta diante da vasta possibilidade de Redes Virtuais factíveis de serem constituídas.

3.1.3 O Conceito de Transporte *Routing* e *Non-Routing*

Finalmente, combinando os conceitos de Redes Virtuais *Overlay* e *Underlay* aliados aos conceitos de dados e *metadados*, um terceiro fundamento pode ser estruturado. Uma característica comum a todas as redes virtuais consiste na necessidade de identificação de seus objetos, tipicamente em um espaço de nomes separado do espaço de nomes regular do IP e dos nomes de domínios/nós da Internet. Este requisito, juntamente à necessidade de métodos alternativos de roteamento e endereçamento (através do uso de endereços virtuais implementados sobre o IP), conduz à formação de um conjunto de identificadores para objetos da rede, geralmente constituídos pela tripla (nome, endereço e rota). Tal tripla, cujo principal objetivo consiste em disponibilizar informações sobre o dado, incluindo sua designação (nome), sua localização corrente na rede (endereço) e o meio de acessá-lo (rota) podem ser naturalmente associados ao *metadado*.

Diante da existência física dos *metadados*, a questão seguinte consiste nos meios e rotas utilizados para o seu transporte em uma rede virtual. Duas principais soluções podem ser delineadas: o transporte do dado e do *metadado* pode ocorrer simultaneamente e de forma análoga (ou seja, utilizando os mesmos mecanismos) ou separadamente (usando diferentes caminhos). Esta última solução consiste em um artifício especialmente interessante quando o objetivo da Rede Virtual *Overlay* compreende a implementação de mecanismos de nomeação e localização (e.g. SFR [14]) deixando que a transferência de dados ocorrer através da infra-estrutura de roteamento IP.

Outra solução, antagônica à anterior, trata das redes virtuais cujo principal objetivo consiste em modificar a infra-estrutura de roteamento tradicional (i.e. o roteameto IP), como por exemplo, através da implementação de uma arquitetura *multicast* em uma rede substrato (e.g. i3 [8]). Este nicho de redes virtuais será, forçosamente, responsável pelo transporte tanto de dados quanto de *metadados*. Este consiste em um conceito fundamental na classificação de redes virtuais que são diferenciadas neste ponto em "propósito" e "aplicações" diferentes em virtude de serem ou não responsáveis, também, pelo próprio transporte de dados. Podem-se observar na Figura 11 duas redes virtuais e suas respectivas

redes substrato (i.e. Rede IP), uma que realiza alguma forma de inferência sobre o roteamento (à direita), também denominada Rede Virtual *Routing* (ou, melhor dizendo, de Roteamento) e outra que não realiza (à esquerda), também denominada Rede Virtual *Non-Routing* [45].

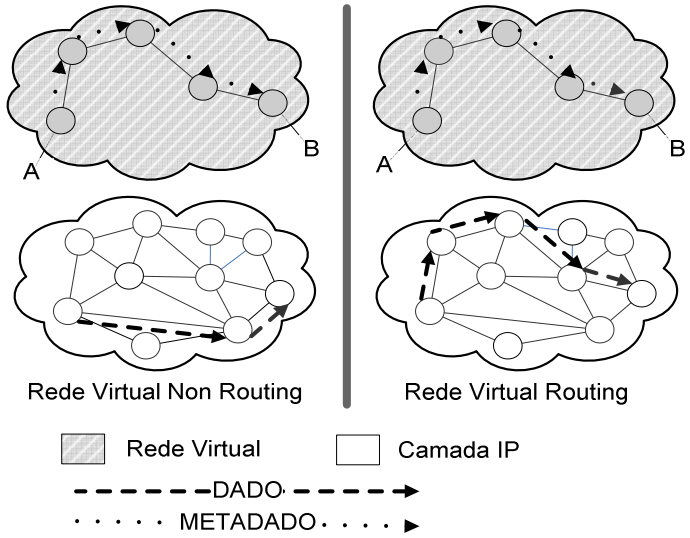

Figura 11. Redes Virtuais *Non-routing* versus *Routing*

Redes Virtuais *Non-Routing*, ou seja, aquelas que não interferem no curso natural do roteamento IP possuem como principal objetivo habilitar uma determinada funcionalidade (e.g. serviços de resolução de nomes, endereçamento, segurança, QoS, etc) através do estabelecimento de canais de controle de *metadados* (e.g. enviando o nome e endereço de um objeto e, às vezes, a rota para acessá-lo) desvinculados dos dados (e.g. o conteúdo referido pela tripla nome, endereço e rota) [45]. O objetivo na formação de uma Rede Virtual de Roteamento compreende o aumento do controle e de funcionalidades de uma rede na seleção de suas rotas, podendo habilitar novos caminhos para o transporte de dados.

Em virtude das mudanças no transporte/manipulação de dados e *metadados*, agora realizados pela Rede Virtual *Routing*, impactos no desempenho são esperados [45]. Acrescentando, por exemplo, um mecanismo seguro de identificação de nós (i.e. através da identificação criptográfica de objetos) para as camadas inferiores na pilha de protocolos (e.g. projeto *Secure*-i3) permite-se que diversas aplicações se beneficiem de tal funcionalidade. Este artifício, por exemplo, desobriga as aplicações de implementar seus próprios mecanismos de identificação de nós, que geralmente são adotados em uma fase inicial de forma não padronizada e não otimizada concentrando nos detalhes de implementação específicos de cada projeto.

3.2 Fundamentos da Taxonomia Proposta

Um dos aspectos fundamentais na formalização dos estudos de qualquer campo do conhecimento (não somente das ciências naturais) consiste na capacidade de categorizar seus objetos de maneira significativa (i.e. singular e ortogonal) e padronizada. Através da categorização das entidades de certa ciência, gera-se um processo completo de entendimento e, como conseqüência, tem-se a identificação de suas características relevantes e a simplificação da exposição formal de seus termos/objetos.

As características fundamentais de uma taxonomia, i.e. aquelas que permitem sua avaliação e julgamento de mérito, são: **inclusão, definibilidade, redutibilidade e aplicabilidade** [56]. Tais características garantem respectivamente que a taxonomia compreenda todos os termos de um campo de estudos; defina significativamente seus elementos; que as divisões e subdivisões as quais as entidades são submetidas compartilhem aspectos comuns; e que a taxonomia, como um todo, permitirá aos seus usuários compreender melhor e avaliar as capacidades dos objetos do campo da ciência estudado, bem como, tomar decisões/julgamentos sobre seus elementos.

Taxonomias apresentam diversas topologias possíveis, variando em complexidade e poder descritivo, desde as formas mais conhecidas como a plana, em faces ou anel até formações mais complexas como a hierárquica e a de rede [57]. A discussão sobre a aplicabilidade e o valor agregado de cada uma destas formas está além do escopo deste trabalho, entretanto, será aqui adotada a topologia hierárquica, conforme demonstrarão explicações seguintes.

Um objetivo precípuo da taxonomia proposta compreende o entendimento formal, categorização canônica e estudo das Redes Virtuais. Um aspecto proeminente no estudo destas consiste na diferenciação de seus propósitos: uma classe delas é formada pelo uso de aplicações únicas (e neste contexto, utiliza-se o seu significado estritamente computacional: um programa que disponibiliza um serviço) com o propósito de disponibilizar um serviço aos usuários. No exemplo de aplicações *peer-to-peer*, em que uma única aplicação disponibiliza dados (e.g. BitTorrent [58]), ou *streams* de vídeo (e.g. Joost [59]), ou ainda habilita interações de voz e vídeo (e.g. Skype [60]), têm-se redes virtuais cujo único propósito consiste em prover um serviço singular através do uso de uma aplicação. A outra classe compreende as aplicações cujo principal propósito consiste em prover uma determinada funcionalidade.

No presente contexto, a funcionalidade pode ser entendida como uma nova característica ou propriedade habilitada na rede substrato, ao invés de um único serviço. No exemplo de Redes Virtuais como o HIP ou i3 temos a provisão de novas características tais como segurança e mobilidade que podem ser usufruídas por qualquer aplicação/usuário residente no referido nó virtual. A classificação de uma rede virtual na primeira ou segunda classe permite a derivação de diversas informações/discussões, tais como a dificuldade de implementação, a possibilidade de adoção de padrão, o desempenho, os limites superiores de expansão (crescimento em escala) e etc. Diante disso, pode-se definir o primeiro táxon como o "propósito" da Rede Virtual que agrega duas classes: a "**Rede Virtual de Aplicação Específica**" (ou **ASNe** – *Application Specific Networks* [45]) e a "**Rede Virtual de Propósito Específico**" (**PSNe** – *Purpose Specific Networks* [45]).

As Redes Virtuais de Propósito Específico compreendem a classe alvo desta dissertação. São formadas por um conjunto de nós pertencentes a uma infra-estrutura de rede base (entende-se por rede base aquela que serve de substrato para a rede *PSNe*) e têm por objetivo prover novas funcionalidades/características estruturais habilitando a rede atual a desempenhar diferentes papéis. Consequentemente, elas adicionam uma camada extra de indireção ou virtualização, alterando uma ou mais propriedades da rede substrato ao incorporar a ela funções "avançadas" (i.e. funções não nativamente suportadas). Analogamente, as Redes Virtuais de Aplicação Específica, igualmente formadas por um conjunto de nós pertencentes a uma infra-estrutura de rede base, possuem o objetivo de habilitar novos serviços/aplicações específicas na rede substrato, podendo, inclusive utilizarem-se das características estruturais implementadas por uma PSNe (e.g. o software ASNe Coral [61] utilizando a PSNE i3 do PlanetLab).

PSNes ou **ASNes** podem ser classificadas particularmente como **Redes Virtuais U*nderlay***, podemos citar como exemplo o projeto PSNe *Network Pointers [62]* que substitui o enlace *Ethernet* por um mecanismo de encaminhamento inserido como uma função modular baseada no uso de ponteiros de redes (análogo aos conceitos de ponteiros de programação) que permitem aos *hosts* utilizarem qualquer forma de acesso à camada de enlace na descoberta das interfaces físicas destino. Além disso, podem também ser classificadas como **Redes Virtuais *Overlay***, e os projetos PSNE HIP e ASNe Joost podem ser citados como exemplo. No primeiro caso, as conexões estabelecidas entre as entidades HIP (a serem explicadas adiante no texto) são transportadas através da infra-estrutura "regular" de roteamento IP e fornecem infra-estrutura de transporte de *metadados* para outras aplicações.

A segunda classificação, advinda das discussões anteriores, é fruto da análise de Redes Virtuais do ponto de vista da facilidade de implementação, segurança, complexidade, capacidade de crescimento em escala, desempenho, considerações sobre as modificações necessárias para ser implementada na Internet, bem como fatores arquiteturais limitantes que as inviabilize. Elegendo a camada de implementação como o segundo táxon, tem-se uma nova subdivisão em Redes Virtuais *Underlay* e *Overlay*.

Surgem neste ponto discussões sobre: em que camada uma funcionalidade é melhor implementada (acima ou abaixo da camada IP); esta escolha impactará em uma capacidade de crescimento em escala maior ou menor; para determinados serviços, este crescimento representa uma questão de maior importância do que a facilidade de implementação? Estas e outras discussões devem ser realizadas quando da classificação das Redes Virtuais em uma destas duas classes para permitir que comparações adequadas sejam feitas entre projetos pertencentes à mesma categoria.

Finalmente, a terceira e última característica relevante diz respeito ao transporte de dados e/ou *metadados* ser realizado pela rede substrato ou pela Rede Virtual. Conforme definido anteriormente, Redes Virtuais *Routing* são aquelas que transportam dados e *metadados* utilizando a mesma rota e, portanto, tendo seus nós virtuais responsáveis por esta tarefa que também representa seu propósito (i.e. realizar algum mecanismo *source routing* ou *loose source routing* que representam, respectivamente alterações completas ou parciais no mecanismo de encaminhamento de pacotes). Uma Rede Virtual *Non-Routing* é aquela que realiza o transporte do dado (i.e. o atual conteúdo referenciado) através de um caminho alternativo (e.g. a infra-estrutura de roteamento IP). De maneira análoga às características do segundo táxon, desempenho, capacidade de crescimento em escala, confiabilidade e segurança são considerações fundamentais.

A maioria dos projetos de Redes Virtuais é introduzida na Internet em sua fase inicial como uma prova de conceito e, portanto, utilizando computadores de propósito geral (e.g. PCs comuns) ao invés de computadores otimizados de propósito específico (e.g. roteadores). Diversos aspectos relevantes devem ser considerados em virtude deste cenário: o primeiro consiste na complexidade, dificuldade em garantir níveis de segurança adequados e disponibilidade dos primeiros equipamentos em detrimento dos segundos. Tal complexidade, em verdade, tal generalidade de propósito os torna menos otimizados para realizar o transporte de dados, ou melhor, dizendo, o encaminhamento de pacotes apresentando um desempenho inferior aos *hardwares* de propósito específico. Estes fatores devem, portanto,

ser levados em consideração quando da proposição de novas arquiteturas para a Internet utilizando a primeira categoria de equipamentos.

Obviamente, uma alternativa possível de ser adotada pelos arquitetos de Redes Virtuais consiste na utilização de equipamentos especializados para a implementação de seu projeto. Porém, esta solução contrapõe-se com a primitiva de crescimento em escala, haja vista que a obtenção de uma abrangência global (i.e. da ordem de grandeza da Internet) não pode estar baseada na necessidade de adoção por parte da comunidade (científica, comercial e doméstica) de um *hardware* customizado de propósito específico.

Isto posto, a alternativa factível, que corrobora as pesquisas aqui realizadas, consiste na utilização de computadores de propósito geral associados com uma cuidadosa seleção de natureza (e volume) dos dados a serem (se necessário) transportados/processados pela nova Rede Virtual. Estabelece-se, finalmente, o terceiro táxon que compreende a "rota dos dados" categorizando as Redes Virtuais "*Routing*" e "Non-*Routing*".

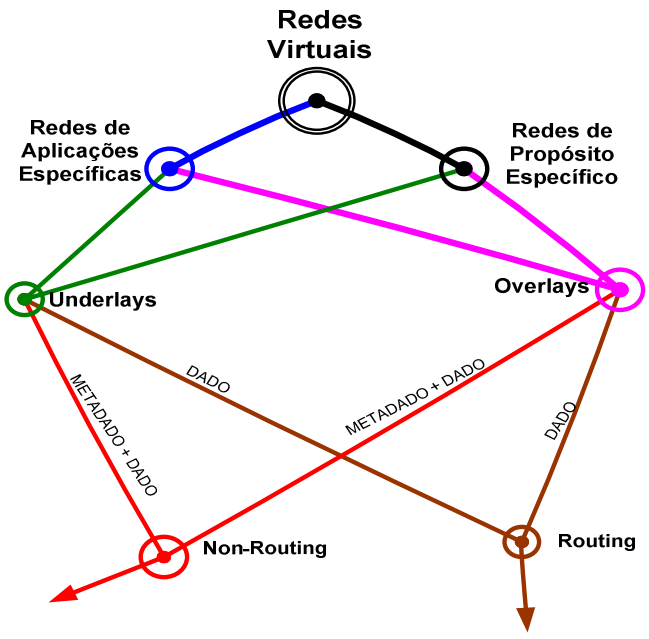

Figura 12. Taxonomia de Redes Virtuais

Com os três táxons aqui propostos, é possível vislumbrar a disposição da taxonomia conforme ilustrado na Figura 12. Podem-se observar as relações de contenção entre os diversos níveis estabelecendo um relacionamento hierárquico entre os três níveis propostos. O primeiro nível define as Redes de Aplicações Específicas e as Redes de Propósitos Específicos. O nível seguinte define quais os subtipos, *Underlay* ou *Overlay*, que

cada uma das Redes do primeiro nível pode assumir. O terceiro e ultimo nível define os oito possíveis tipos subtipos de Redes Virtuais definidos por esta Taxonomia.

Uma classificação baseada na inspeção de campo de diversos projetos de Redes Virtuais foi realizada e apresentada neste trabalho servindo como prova de conceito desta taxonomia comprovando sua capacidade de inclusão, definibilidade, redutibilidade e aplicabilidade.

CAPÍTULO 4 UM ESTUDO DE CASO DA TAXONOMIA PROPOSTA

As vantagens oferecidas pelo uso de Redes Virtuais para o desenvolvimento de novas funcionalidades para a Internet são notórias tanto pelas características supracitadas inerentes às redes deste tipo, quanto pela variedade de projetos que utilizam esta tecnologia. Um apanhado de projetos de redes virtuais será apresentado neste capítulo e sua classificação de acordo com a Taxonomia proposta anteriormente será realizada.

4.1 Projetos Relacionados: análise de diferentes espaços de nomes

Diante da variedade de projetos de redes virtuais encontrados na literatura, um apanhado dos trabalhos de maior relevância, levando-se em consideração a completude, inovação do modelo e a maturidade da proposta, é apresentado abaixo. Uma análise dos projetos *Active Networks Architecture* [63], BitTorrent [58], CoDeeN [64], Coral [61], CoBlitz [65], F5 [66], FARA [10], HIP [11], Hi3[47], i3 [8], IPNL [54], Joost [67], *Network Pointers* [62], OverCast [68], OverQoS [55], Pier [34], RON [7], Skype [69], TRIAD/WRAP [70], SFR [14] e X-*Bone* [71] é realizada nas subseções seguintes.

4.1.1 Active Networks Architecture

Active Networks Architecture consiste em um projeto do grupo *Networks and Systems* (*Telemedia*) do MIT desenvolvido, principalmente, por D. L. Tennenhouse *et al* [72]. Esta Rede Virtual permite que seus usuários customizem as execuções de processamentos de pacotes através da inserção de trechos de códigos (i.e. customizações de *software*) em nós virtuais da rede. Utiliza uma interface de rede programável (em oposição ao modelo estático de encaminhamento de pacotes realizado na Internet em que alterações no método de roteamento não podem ser feitas por entidades externas ao roteador ou influenciadas por customizações de *software* em tempo real) para tal

Permite, portanto, que inferências sejam feitas dinamicamente no processamento de pacotes habilitando novos comportamentos para as aplicações, especialmente para aquelas que implementam mecanismos de *multicast*, fusão de informações e serviços de armazenamento em grande escala (e.g. sistemas de *storage* e *data warehouse*). Entre outras funcionalidades, este *framework* define um novo paradigma de transporte e

processamento de dados em redes, requerendo a acomodação de dois novos modelos: o encapsulado, em que programas adaptadores (*customizers*), ou referências para programas, são transportados diretamente na rede (ao invés de cabeçalhos modificados individuais para cada pacote); e de "roteadores programáveis" em que o encaminhamento de pacotes pode ser customizado utilizando uma interface programável *out-of-band* (i.e. estabelecendo canais de controle em uma via secundária ao transporte dos dados, propriamente dito).

O objetivo final consiste em minimizar os recursos necessários para que um determinado serviço fim-a-fim seja facilmente disponibilizado, logo que um conjunto de funções básicas esteja disponível na plataforma de nós programáveis da Rede Virtual *Active Networks.* Isto habilita, portanto, uma variedade de novos serviços mais complexos e otimizados que estarão baseados neste novo cenário dinamicamente configurável.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.2 BitTorrent

BitTorrent representa uma rede virtual baseada em um protocolo de distribuição de conteúdo que utiliza uma infra-estrutura de Tabela *Hash* Distribuída para realizar a identificação de conteúdos (i.e. dados, arquivos) e suas referências (e.g. URLs) associadas. Foi inicialmente desenvolvido por B. Cohen [58] em 2003 embora apresente, atualmente, diversos *frameworks* derivados e adaptados (e.g. o projeto Emule [73] que implementa um modelo de economia baseado em mecanismos de compensação aos usuários).

Utiliza nós virtuais capazes de segmentar grandes arquivos em pequenas porções, maximizando o desempenho geral da rede de distribuição de arquivos, usuários e aplicações podem se beneficiar desta Rede Virtual de Aplicação Específica.

- Classificação segundo a Taxonomia proposta**: Rede Virtual ASNe Overlay Routing.**

4.1.3 CoDeeN

O projeto CoDeeN consiste em uma rede virtual acadêmica de testes para distribuição de conteúdo CDN (i.e. *Content Distribution Network*) [64] concebida pelo *Network Systems Group* da Universidade de Princeton sob a infra-estrutura do PlanetLab. O CoDeeN representa uma rede de alto desempenho de servidores *proxy* espalhados em diversos nodes do PlanetLab. Sua principal função é a de redirecionamento e replicação de conteúdo

cooperando entre si para prover um serviço *web* de entrega rápida e robusta de conteúdo aos usuários da CoDeeN.

Almejando utilizar a Rede CoDeeN, usuários devem configurar seus navegadores para utilizar um *proxy* CoDeeN adequado. Conforme pode ser observado na Figura 13, um nó virtual CoDeeN consiste em uma instância de um *proxy* operando nos modos de encaminhamento normal e reverso, bem como no encaminhamento lógico de conteúdo e monitoração da infra-estrutura. Assim que um cliente solicita o serviço de um nó CoDeeN, este age como um *proxy* de encaminhamento na medida em que tenta resolver a solicitação. Ao ocorrerem falhas (*cache misses*), este passará a agir como um redirecionador decidindo para onde deverá encaminhar a requisição cliente que geralmente é recebida por outro nó CoDeeN atuando como *proxy* reverso para o servidor original.

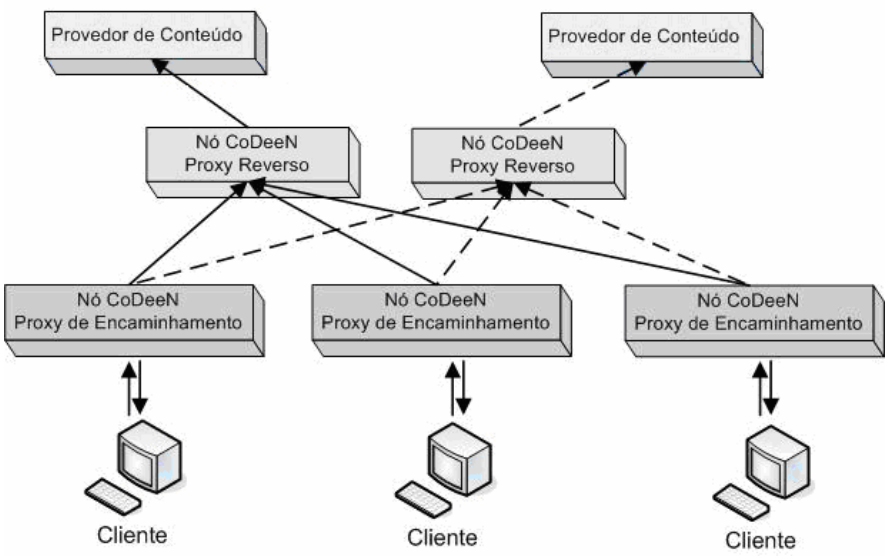

Figura 13. Arquitetura CoDeeN

Na maior parte das requisições, diversas características, tais como localidade, carregamento do sistema, confiabilidade e proximidade são consideradas pelos redirecionadores para a seleção do próximo nó CoDeeN. Mecanismos de confiança e segurança são utilizados para analisar os nós de propósito geral candidatos a *proxy,* bem como para decidir sobre atender ou rejeitar requisições.

- Classificação Segundo a Taxonomia proposta**: Rede Virtual ASNe Overlay Routing.**

4.1.4 Coral

A Rede Virtual de distribuição de conteúdo Coral, outrora denominada Coral *Cache*, consiste em uma rede P2P (*peer-to-peer*) destinada a espelhar conteúdos *web* utilizando a largura de banda de colaboradores. Esta técnica tenta dividir a largura de banda dos nós participantes da Rede Virtual Overlay através do uso de *web proxies* e servidores de nome espalhados pelos nós da infra-estrutura do PlanetLab. Uma camada virtual DNS P2P realiza redirecionamentos transparentes toda vez que consultas Coral são realizadas procurando possíveis entradas atualizadas das páginas buscadas ou armazenando-as instantaneamente.

Para acessar uma página *web* através do armazenamento do Coral (e, por conseqüência, reduzir possíveis sobrecargas à rede original do conteúdo de acesso) basta acrescentar o sufixo .nyud.net ao URI (*Uniform Resource Identifier*) com o endereço da página desejada. Desta forma, o endereço fictício http://meuhost.com.br/subdir seria agora transformado no endereço "coralizado" http://meuhost.com.br.nyud.net/subdir.

Um dos principais objetivos do Coral é evitar que ocorram sobrecargas de acessos que venham a dissuadir colaboradores de executarem suas aplicações pelo temor a momentos de pico. Com este intento, é realizada a indexação dos conteúdos/dados através de uma Tabela *Hash* Distribuída capaz de criar conjuntos auto-organizáveis de nós que buscam informações entre si almejando evitar a comunicação com servidores longínquos e sobrecarregados.

Foi desenvolvido pelo NYU *Secure Computer System* no projeto IRIS.

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.5 CoBlitz

O CoBlitz é um serviço que utiliza a infra-estrutura do CoDeeN para a transferência via HTTP de arquivos de grandes dimensões (de centenas de megabytes a dezenas de gigabytes), de forma escalável, sem exigir alterações nos servidores *web* e nos clientes [74; 65].

Diferentemente de um servidor de *proxy* tradicional (onde um arquivo grande seria integralmente replicado no *cache* do servidor, ocupando espaço de vários arquivos menores e exigindo diversos acessos em disco), o CoBlitz distribui a tarefa para os vários nós do CoDeeN, que ficam responsáveis por armazenar diferentes partes do arquivo. Tal

abordagem diminui a carga em cada nó, tanto de memória necessária como de acessos em disco, além de minimizar o impacto de um nó fora do ar.

Para efetuar o *download* de um arquivo, deve-se prefixar a URL com http://coblitz.codeen.org:3125. Sendo assim, para acessar via CoBlitz uma imagem de um DVD da hipotética URL http://meuhost.com.br/imagem.iso deve-se utilizar a URL http://coblitz.codeen.org:3125/meuhost.com.br/imagem.iso. Os seguintes passos irão ocorrer, conforme ilustrado na Figura 14 (um exemplo de arquitetura CoBlitz com dois clientes):

- Um servidor DNS especializado mapeia o nome coblitz.codeen.org para o nó mais próximo do cliente, onde um agente especializado fica escutando por requisições HTTP na porta 3125;
- O cliente contata este agente, que converte a requisição do arquivo inteiro em requisições de partes pequenas do arquivo. Estas requisições parciais são então distribuídas em paralelo para os diversos nós da rede CDN;
- Cada nó requisita ao servidor original a parte que lhe foi atribuída do arquivo pelo agente CoBlitz. Ao receberem a resposta do servidor, cada nó armazena em seu *cache* sua parte do arquivo, devolvendo uma cópia ao agente CoBlitz;
- O agente recebe as diversas partes do arquivo e as reenvia para o cliente em ordem, como se fosse um único *download*.
- Caso um segundo cliente deseje acessar o mesmo arquivo via CoBlitz, o agente CoBlitz irá acessar a CDN para recuperar os dados do arquivo; o servidor *web* original só será contatado caso algum dos nós da CDN já tenha descartado a informação do arquivo de seus *caches* (neste caso, somente as informações faltantes serão recuperadas do servidor original)

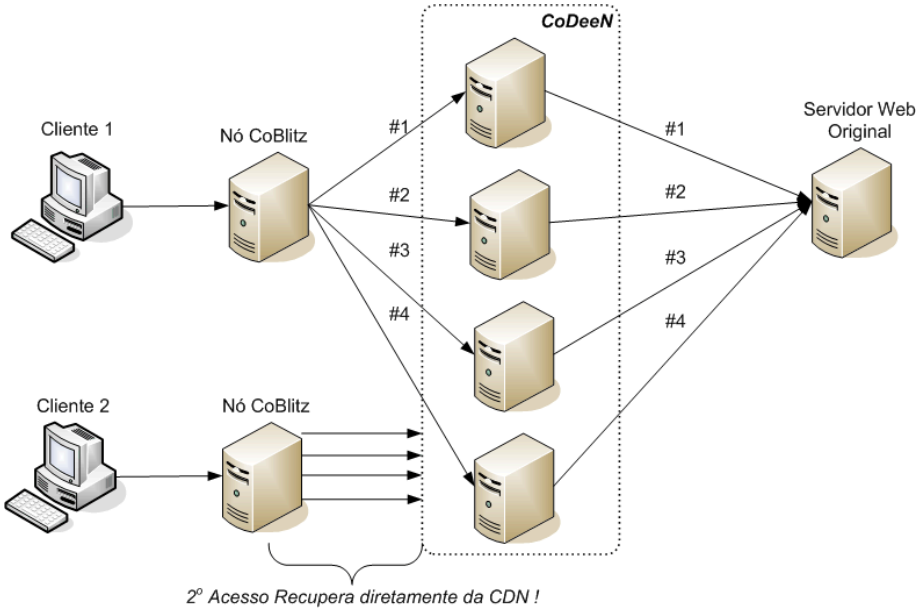

Figura 14. Arquitetura CoBlitz

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.6 F5

O projeto F5 [66] foi concebido como uma Rede Virtual para distribuição de conteúdo capaz de gerenciar o tráfego de rede, proporcionando às aplicações e serviços *web* níveis de segurança e de serviços customizados. Atualmente este utiliza a infra-estrutura do BIG-IP (denominação inicial do projeto) na oferta das funcionalidades de balanceamento de carga combinadas com uma diversidade de protocolos (e.g. BGP [75], OSPF [75], e RIP[75] para roteamento; *cache* estático HTTP para balanceamento de carga; criptografia SSL para segurança; e LDAP e Radius para a autenticação de usuários).

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.7 FARA

Forwarding directive, Association, and Rendezvous Architecture define um modelo conceitual (*framework*) com considerável generalidade e flexibilidade, baseados na problemática da separação de nomes de nós e endereços de rede. Seu principal objetivo consiste em evitar os problemas clássicos de sobrecarga da Internet, e do endereçamento IP, através da seguinte abordagem: definir uma clara separação entre identificadores de aplicações e identificadores da camada de rede; definir mecanismos de encaminhamento mais

eficientes sem necessariamente ser introduzido um novo espaço de nomes globais, tudo isso considerando os mínimos requisitos de segurança admissíveis para uma infra-estrutura abrangente como a Internet.

Através da clara separação dos identificadores de localização e nomeação (de serviços e de rede), torna-se possível desvincular ambos de características mutuamente restringentes ao outro, permitindo maior flexibilidade ao modelo (i.e. mobilidade garantida pela interdependência de aplicações e nós). Consequentemente, esta Rede Virtual *Overlay* possibilita a escolha de um modelo de nomeação e de localização baseados em características inerentes de cada qual.

A generalidade do modelo corrobora as evidências de que os conceitos de localização e identidade, atualmente acoplados numa única nomenclatura, podem realmente ser separados. No FARA, a comunicação fim-a-fim de nós é substituída por uma comunicação entre pares de "entidades" (*Entity*) lógicas estabelecidas através de ligações lógicas que são chamadas "associações". Estas realizam a troca de pacotes através de um "substrato de comunicação". Além disso, dada a generalidade do modelo, componentes adicionais podem ser acoplados à infra-estrutura, tais como servidores *rendezvous* ou serviços dos diretórios FARA.

Uma entidade consiste em uma generalização de uma aplicação a qual, usualmente, é remetida a um endereço de rede em uma comunicação. As entidades são *statefull*, ou seja, armazenam o estado da aplicação e o estado da comunicação. Na prática, uma entidade consiste na menor unidade que pode ser móvel, podendo compreender um *cluster* inteiro ou simplesmente uma *thread* de algum processo, corroborando a idéia de manter um alto grau de abstração e flexibilidade do modelo.

Uma associação (*Association*) é baseada no estado persistente de uma comunicação já estabelecida entre entidades. Entidades pares comunicam-se utilizando tais associações. Na arquitetura FARA, cada pacote possui um identificador de associação AId (*Association Identifier*), que permite às entidades multiplexar e demultiplexar cada mensagem à sua associação específica.

FARA utiliza, também, um substrato de comunicação de rede responsável pela entrega de dados para as associações. Este representa o último componente do modelo FARA, em que uma especificação de interface (API), denominada "linha vermelha" (*The Red Line*), é definida para manter a separação modular das funções de encaminhamento da entidade.

Quando uma entidade quer enviar um pacote a uma de suas associações, utiliza um campo do cabeçalho de destino denominado FD (*Forwarding Directive*) para enviá-lo

através do substrato de comunicação. Este contém as informações necessárias para garantir a entrega dos dados à entidade destino desejada, embora um mecanismo particular de encaminhamento possa reescrever o FD durante o roteamento. Na Figura 15 é ilustrado o mecanismo de encaminhamento de pacotes na arquitetura FARA. Duas Entidades *statefull* A e B estabelecem uma Associação lógica utilizando o Substrato de comunicação e todas as informações referentes à sua comunicação são preservadas durante todo o processo.

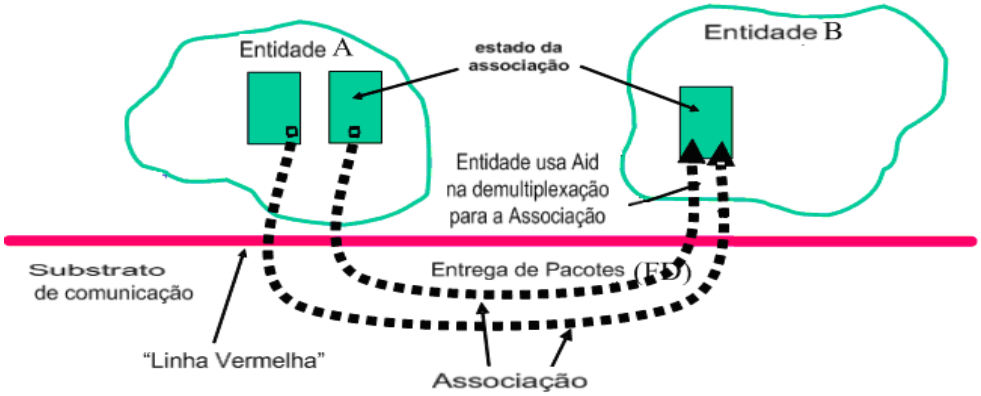

Figura 15. Entrega de Pacotes no Ambiente FARA [10]

O projeto FARA modulariza a arquitetura em dois níveis conceituais: o da abstração das entidades e das associações no nível superior e um segundo, no nível inferior, de um substrato de comunicação independente para o encaminhamento de pacotes (não orientados à conexão). Como o modelo não especifica o formato do FD, quando uma entidade A quer enviar um pacote para B, ela deve definir seu próprio FD (veja a Figura 15). No destino, o AId é utilizado na demultiplexação de pacotes advindos do mesmo substrato de rede. O objetivo principal desta arquitetura *Overlay* consiste na separação da identificação da localização, permitindo o suporte à mobilidade bem como a evolução independente dos mecanismos nos dois níveis.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Non-Routing.**

4.1.8 HIP

O *Host Identification Protocol* (HIP) consiste em uma Rede Virtual *Overlay* que propõe a inserção de uma camada adicional de nomeação situada entre as camadas de Transporte (TCP) e rede (IP), que tem por objetivo identificar univocamente os *hosts* na

Internet. Para isso, utiliza um HI (*Host Identity*), que representa a chave pública de um par de chaves pública-privada.

Ao utilizar o HIP, as camadas superiores de rede TCP/IP, incluindo a aplicação, não são capazes de acessar mais o IP. Em substituição ao endereço de destino IP, elas utilizam o HI destino. Desta forma, a informação relativa à interface de rede (i.e. Ponto de Acesso à Rede) fica oculta e sua manipulação é tratada pela nova camada *overlay*.

Após uma fase inicial de autenticação entre os *hosts*, todo o tráfego subseqüente flui com segurança através da criptografia do protocolo IPSec [76]. Portanto, a semântica da conexão (i.e. a conexão TCP ou a associação UDP) fica baseada numa identidade fixa representada pelo HI (*Host Identity*) origem e destino, e não mais pelo endereço IP, que é passível de alterações constantes em face de mobilidade.

Na Figura 16, uma ilustração da semântica de conexão HIP é apresentada. Pode-se observar que na nova pilha de protocolos HIP ilustrada na figura, o estabelecimento de uma conexão entre um par de nós não é mais baseado no par **TCP/IP <IP, Porta>,** ficando agora as interfaces programáticas vinculadas à dupla **<HI, Porta>.** A vantagem evidente obtida com esta técnica consiste no fato que ao sofrer mobilidade, um nó pode ter seu endereço IP alterado sem que isso comprometa a semântica de conexões já estabelecidas devido à mesma estar baseada no HI que é fixo, necessitando somente realizar a troca de pacotes de *readdress* para sinalizar a nova localização da entidade.

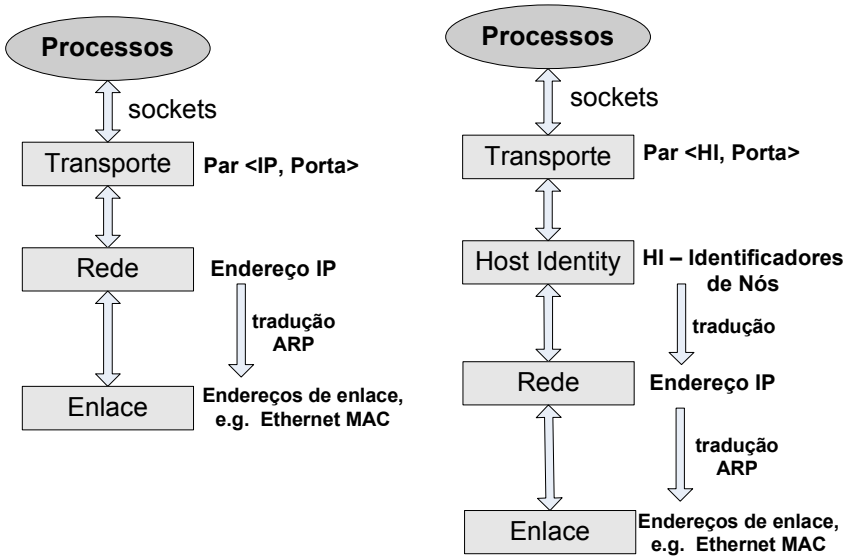

Figura 16. Associação entre TCP/IP (esquerda) e TCP/HIP/IP (direita) [11]

Com o uso do HIP, a mobilidade dos *hosts* em uma comunicação (vale ressaltar que no caso de *double jump*, será necessária a presença adicional de servidores *rendezvous*

[9]), o *multihoming* e a segurança no tráfego de informações são resolvidos sem requerer modificações em todos os equipamentos intermediários, sendo somente necessário que os dois *hosts* finais comunicantes utilizem o protocolo. Obviamente, a introdução de uma nova camada implica em um novo espaço de nomes, sendo esta o foco de investigação deste trabalho.

Antes do estabelecimento da comunicação através do tráfego IPSec, uma fase inicial de sinalizações HIP é estabelecida. A Figura 17 ilustra a troca de pacotes I1, R1, I2 e R2 do tipo desafio-resposta (*challenge-response*) necessária para a autenticação dos pares HIP. Em seguida, a troca de dados acontece normalmente utilizando IPSec [76].

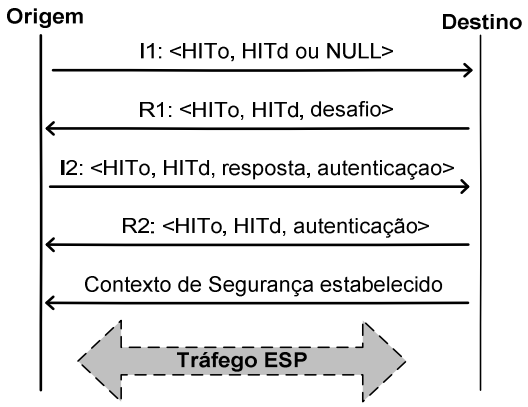

Figura 17. Sinalização inicial de um Tráfego HIP [11]

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Non-Routing.**

4.1.9 HI3

Apresentando objetivos similares ao HIP, o projeto de Rede Virtual Hi3 [47] introduz um novo espaço de nomes baseado em identificadores de nós HI (*Host Identifiers*) da mesma forma como é realizado no HIP. Entretanto, a troca inicial de chaves públicas/privadas se dá através da infra-estrutura de Rede Virtual de Roteamento i3.

Foi originalmente arquitetado por Gurtov *et al* no Instituto de Tecnologia da Informação de Helsinki.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Non-Routing.**

4.1.10 i3

Internet Indirection Infrastructure, ou simplesmente i3, é um projeto que busca a mobilidade através da separação do ato de recepção e do ato de entrega de dados. Para isso, i3 utiliza-se uma Rede Virtual *Overlay* de servidores, que é encarregada de receber os dados dos emissores e encaminhá-los até os destinatários.

Devido ao fato de ser um *overlay* implementado sobre uma DHT *Chord* [22] (que consiste em uma DHT de propósito geral capaz de prover mapeamento unívoco entre espaços de identificadores e um conjunto de nós), sua infra-estrutura apresenta certa complexidade. Para receber um pacote, um *host* i3 precisa primeiro registrar seu *trigger* em um servidor *overlay*. Um *trigger* no contexto deste projeto representa um registro lógico (índice) na DHT, constituído pela tripla <identificador, endereço IP, Número de Porta> (em verdade, há muito mais informações em um *trigger*, tais como o endereço IPv6 e informações sobre NAT). Desta maneira, o *trigger* não precisa desempenhar funções restritas de identificação de um *host*, podendo ser usado de maneira mais granular em aplicações específicas.

Neste cenário, para enviar uma mensagem, o emissor só precisa postar o dado junto com o *trigger* destino (i.e. ambos situados na área de dados do UDP) e o *overlay* i3 é responsável pela tarefa de encaminhá-lo ao destino.

Através da desvinculação do endereço da interface de rede (endereço IP do Ponto de Acesso a Rede) e do identificador de serviço (número de Porta) em relação ao identificador do *host* realizada pelo protocolo i3, tornou-se possível acomodar de forma transparente a mobilidade (inclusive *double jump*) através da simples atualização da porção de localização do *trigger*. Os demais campos permanecem então intactos, identificando univocamente os pares comunicantes.

Além disso, é possível utilizar-se um identificador comum de *triggers* para realizar uma comunicação *multicast* (ou ainda *anycast* [33]), através do registro de múltiplos *hosts* com o mesmo *trigger* público, porém variando os endereços IP e Portas de cada equipamento. Até mesmo técnicas de *source routing* podem ser nativamente implementadas nesta arquitetura através do uso de pilhas de identificadores (i.e. identificadores que apontam para outros identificadores ao invés de apontar para um único endereço físico). O modo operacional de mobilidade, *multicast* e *anycast* é ilustrado na Figura 18.

No primeiro cenário, Figura 18(a), acontece uma mobilidade simples (i.e. *single jump*) com o Receptor (R) que ao deslocar-se passa a ter um novo identificador (R').

Para continuar recebendo os dados, a infra-estrutura i3 (baseada na DHT *Chord* [22]) precisa redirecionar os dados para a nova localização R'. No caso b), uma entrega da pacotes *multicast* utiliza identificadores i3 denominados **id** comuns para os nós R1, R2 e R3. Finalmente, em c), a entrega de dados *anycast* acontece através da combinação dos n primeiros bits de cada id que permita a identificação singular de um nó R2.

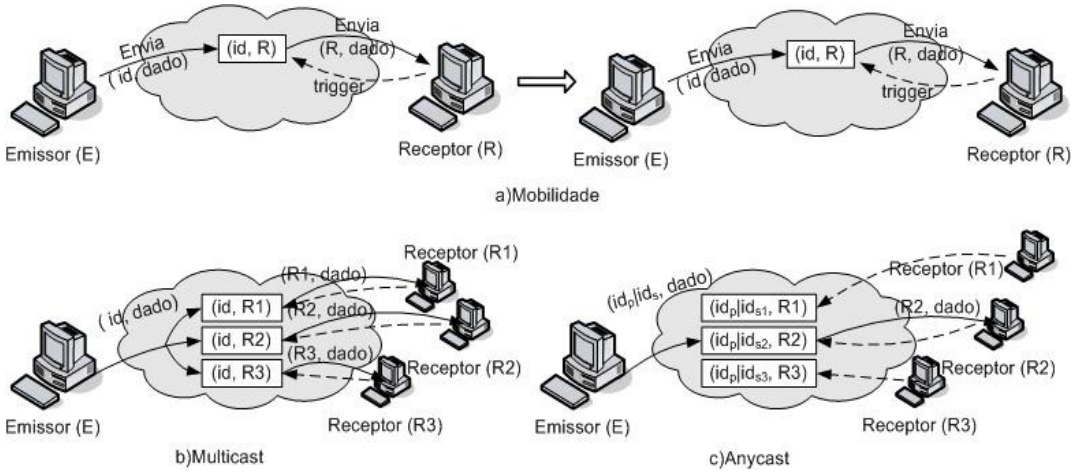

Figura 18. Mobilidade, *Multicast* e *Anycast* na Arquitetura do Protocolo i3. [8]

Um possível agravante em relação à segurança deste ambiente i3 pode ser notado no que diz respeito à responsabilidade inerente dos *hosts* i3 na manutenção das informações de roteamento através da publicação de *triggers*. A vantagem deste mecanismo consiste na grande flexibilidade permitida às aplicações, entretanto, o efeito colateral consiste no fato de criar novas oportunidades para usuários maliciosos influenciarem de maneira inconsistente o encaminhamento de dados. Um segundo problema consiste na sobrecarrega de tarefas embutidas aos equipamentos de propósito gerais (i.e. *hosts*), que agora também são obrigados a tomar decisões sobre o encaminhamento de dados, tarefas típicas de roteadores e servidores (i.e. equipamentos dedicados e de propósito específico).

Uma solução mais adequada para esta infra-estrutura se faz necessária, já que agora as características típicas de roteamento da redes TCP/IP foram modificadas. Para tal, o uso de técnicas de chaves pública/privada foi vislumbrado na infra-estrutura i3, que adotou o conceito de *trigger* público/privado restringindo quaisquer modificações ao *host* proprietário do *trigger* privado. Através deste mecanismo, ataques do tipo *Eavesdropping* (escuta clandestina), *Trigger hijacking* (interceptação ilegal de *trigger*), ataques DoS [13], entre outros não explorados aqui, são evitados.

Finalmente, uma terceira desvantagem a ser explicitada consiste no fato de que o protocolo não se demonstrou apropriado, em seu estado de maturidade atual, para o uso em larga escala, dado que o mesmo se tornou inoperante em face de testes realizados com um número limitado de clientes e servidores. Fica claro que sua utilização em um ambiente como a Internet, no atual estado de maturidade (datado de 2006), ainda não consiste em uma proposta factível para a Internet.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.11 IPNL

IPNL, um acrônimo para *IP Next Layer*, consiste em um projeto de Rede Virtual que objetiva estender as funcionalidades de arquiteturas baseadas em NAT. É capaz de fornecer uma isolação local de domínio e permitir que os mesmos sejam *multihomed* (múltiplos acessos funcionais à Internet) sem poluir a zona comum de roteamento com prefixos locais.

As maiores vantagens do IPNL são:

- Como uma arquitetura que visa estender as funcionalidades do NAT, é capaz de maximizar o reuso da infra-estrutura IPv4 existente, principalmente pela adição de uma nova camada acima do IPv4 que é capaz de realizar o roteamento entre domínios NAT.
- Nomes FQDNs (*Fully Qualified Domain Names* – compostos pela agregação de nomes que remetem à sua localização de domínio) são utilizados como identificador padrão para a troca de pacotes fim-a-fim.
- Estende o espaço de endereços da Internet de forma que o endereço global válido IP componha a parte superior do endereço IPNL, e o endereço privado, sua parte inferior.
- Isola completamente o endereçamento local (composto por nomes IPNL) do endereçamento global (Internet).

Um nó pode ter múltiplos endereços IPNL que podem ser alterados durante o estabelecimento de uma conexão. O FQDN consiste na ferramenta fundamental para manter estes múltiplos endereços IPNL consistentes. Deve ser, portanto, enviado para ambos os pares na troca de pacotes precedente ao estabelecimento de uma comunicação. O tráfego

subseqüente é tipicamente baseado somente em endereços IPNL. Este utiliza ambos os endereços FQDNs e IPNL, porque os endereços FQDN, apesar de roteáveis por roteadores-nl (i.e. *next layer*), possuem comprimento variável, requerendo um custo superior de processamento. Os endereços IPNL possuem a vantagem de apresentarem comprimento fixo e serem eficientemente roteáveis, utilizando-se dos prefixos FQDN somente na configuração inicial do ambiente (*bootstrap*) e manutenção. Uma ilustração da topologia IPNL é mostrada na Figura 19, onde pode ser observada a inserção da camada IPNL entre a camada de Rede e a camada de Transporte do Modelo OSI.

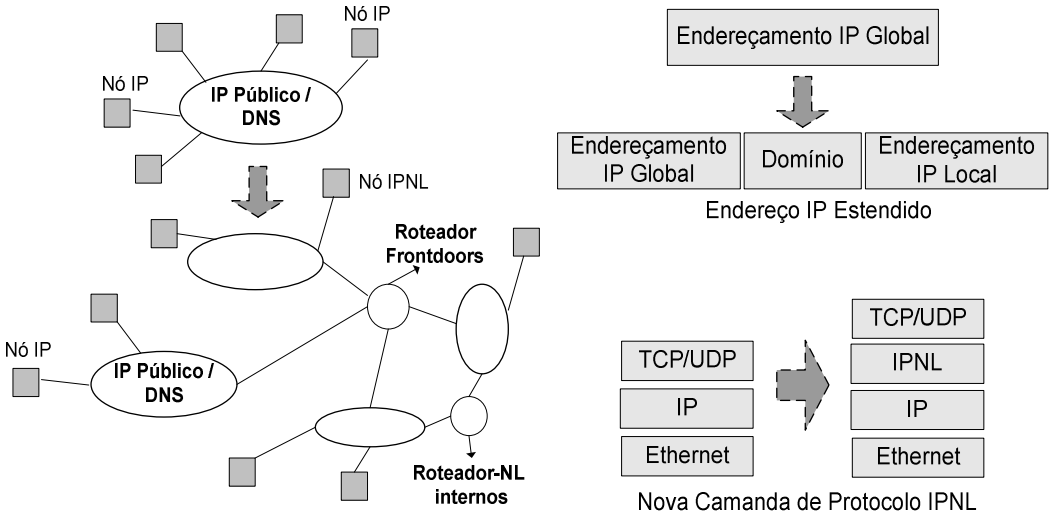

Figura 19. Topologia IPNL [54]

Quando um nó IPNL pertencente a um domínio privado quer se comunicar com outro nó em um domínio externo, consultas DNS podem ser necessárias para a descoberta do endereço destino. Desta forma, roteadores *frontdoor* (veja Figura 19) interdomínios são responsáveis pela tradução entre endereços IPNL públicos e privados. Roteadores-NL (veja Figura 19), por sua vez, são internos a cada domínio privado e são responsáveis por garantir a mobilidade interna de nós com endereços privados, ou seja, nós IPNL.

A topologia IPNL é a mesma que a topologia corrente da Internet: diversos domínios privados conectados à Internet com seus domínios globais públicos inseridos num complexo cenário em que habitam inúmeros *middleboxes* (e.g. domínios NAT, balanceadores de carga) simultaneamente. Como era de se esperar, diversos artifícios precisaram ser desenvolvidos (e.g. roteadores *frontdoors* e roteadores-nl) para acomodar tal arquitetura e,

como em qualquer *overlay*, a inserção de novas camadas apesar de resolver diversos problemas, apresenta uma sobrecarga de processamento.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.12 Joost

Este projeto de Rede Virtual, outrora denominado Joost [59], objetiva proporcionar a distribuição de conteúdos multimídia (i.e. programas de televisão e outras formas de vídeo) através da *web* utilizando, para tal, de uma infra-estrutura base P2P.

Foi concebido por N. Zennström e J. Friis, os mesmos criadores do Skype, almejando disponibilizar a transmissão de conteúdos televisivos pela Internet de forma eficiente.

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.13 Network Pointers

Network Pointer [62] consiste em uma Rede Virtual *Underlay* que propõe um novo mecanismo para obter o desacoplamento da Internet (sobrecarga de funcionalidade em uma única nomenclatura) focalizando a implementação de novas funcionalidades nas camadas pares e inferiores à camada de rede (terceira camada do modelo OSI). Para isso, utiliza uma combinação dos componentes da pilha TCP/IP. Desta forma, o IP se transforma em um mecanismo de acesso à rede.

As funcionalidades e as aplicações baseadas no IP são mantidas funcionando da mesma forma como foram projetadas. Então, um controle adicional implementado sob o IP é necessário pois deseja-se influenciar as funcionalidades da camada inferior à camada de rede (segunda camada do modelo OSI). Isto permite a reconfiguração da camada de enlace (*underlay)* e a habilidade de influenciar os encaminhamentos da camada superior. Os endereços da camada IP são traduzidos em nomes internos de acesso, ou seja, *pointer labels*. Os ponteiros (*pointers*) são responsáveis pelo encaminhamento de *datagramas* para os domínios vizinhos, ou pela modificação de cabeçalhos do pacote (tradução de endereço, tunelamento, compressão do cabeçalhos, entre outros).

Um ponteiro de rede (*Network Pointer*) é uma função de processamento de pacote que possui um endereço local denominado seletor. No exemplo do Ethernet, os pacotes carregariam um endereço de enlace e um seletor. Para executar as funções de roteamento em redes TCP/IP, é, então, preciso contar com o estabelecimento de ponteiros da origem até o seu destino. Em virtude do *Network Pointers* focar modificações nas camadas inferiores à camada de rede, o processo de descoberta de rotas possui um alto custo, na medida em que precisa contar com uma série de *broadcasts* (camada de enlace) do domínio de origem até o destino para descobrir sua localização. Maiores detalhes podem ser encontrados em [62].

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Underlay Routing.**

4.1.14 OverCast

O projeto de Rede Virtual *Overlay* OverCast [68] consiste em um sistema de *multicast* em nível de aplicação que pode ser adotado de forma incremental utilizando a infra-estrutura da Internet. Sua criação foi motivada por aplicações que requerem muita largura de banda para tornarem-se factíveis, tal como vídeo sob demanda. Com o uso de um conjunto estratégico de nós virtuais constroem-se árvores *Overcast* de comunicação em grupo otimizadas para o ingresso e a remoção de novos nós em tempos reduzidos.

Dados são replicados em pontos apropriados na rede virtual de forma a minimizar a banda requerida para atingir múltiplos destinos. Simultaneamente, técnicas de *caching* e replicação de servidores são empregadas. A rede *Overcast* é constituída de um servidor raiz (que pode ser replicado em virtude de tolerância a falhas), suas folhas *Overcast* e clientes HTTP (nenhuma modificação é requerida do ponto de vista do protocolo HTTP nos clientes que se unem aos grupos *multicast*). A árvore de nós virtuais formada é dinamicamente adaptada de acordo com o tráfego enfrentado mostrando-se ideal para aplicações sensíveis a atrasos.

Foi arquitetado idealizado por Jannotti et al na Universidade de Cornell U. e colaboradores da empresa Cisco Systems.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.15 OverQoS

Este projeto compreende uma implementação de Rede Virtual *Overlay* concebida por Stoica e Balakrishnan et al do MIT e da Universidade de U. Berkeley que objetiva prover qualidade de serviço superior à oferecida atualmente pela Internet (i.e. *best-effort*). Com o uso de uma abstração de nós virtuais capazes de controlar taxas de perdas de pacotes, utilizando a técnica CLVL (*Controlled Loss Virtual Link*) [55]; priorização e agregação de pacotes; análise estatística de perdas e largura de banda garantida procura-se garantir níveis de QoS acordados entre nós comunicantes.

Nenhuma inferência quanto à localização de nós OverQoS é realizada e a negociação de parâmetros de QoS acontece em rotas pré-determinadas entre os nós. Aplicações que desejem utilizar os serviços deste projeto devem realizar duas atividades: primeiro, o encaminhamento de seus pacotes via *proxy* OverQoS (este pode estar localizado no primeiro nó *overlay* no caminho, aqui denominado *link* virtual [55], ou no próprio nó da aplicação), e segundo, sinalizar ao *proxy* responsável pela sinalização OverQoS quais os requisitos da comunicação desejados.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.16 Pier

O projeto Pier [34] consiste em uma Rede Virtual baseada em nós de propósito geral de processamento distribuído situada em uma infra-estrutura P2P da ordem de dezenas de milhões de participantes. É capaz de suportar consultas distribuídas em massa análogas ao modelo realizado em banco de dados. Seu principal propósito é o de servir como uma infra-estrutura *Overlay* de armazenamento em bloco para aplicações centradas em informações (i.e. nomes de arquivos, cabeçalhos, logs de sistema e etc) que necessitem de armazenamento e acessos de forma padronizada via rede e com alta disponibilidade, permitindo que as buscas às informações em uma escala como a da Internet sejam bem sucedidas e eficientes.

Considerações sobre a utilização de estruturas de armazenamento sem hierarquia estendem a idéia tradicional de aplicações orientadas ao armazenamento em discos locais para permear uma nova abordagem em que dados são voláteis e devem ser atingíveis em qualquer ponto da Internet. Apesar de tal abordagem não representar uma inovação *de facto* na comunidade, a mesma representa um ruptura no modelo como a Internet é

fundamentada em que a larga escala é atingida baseada em modelos hierárquicos, tais como os protocolos DNS e LDAP ou aplicações Gnutella e KaZaA.

Para atingir tal expansão geográfica e independência de dados (i.e. estruturas de nomeação e roteamento sem hierárquia) utiliza DHTs (baseadas na CAN) como base de sua indexação, conforme pode ser observado na Figura 20. Esta camada substrato recebe consultas da segunda camada, a Pier, e é composta pela agregação de uma Base de Dados, uma DHT e uma Rede Virtual de Roteamento na otimização dos tempos de consultas. A camada Pier adiciona uma camada de serviços entre as aplicações que o utilizam e o armazenamento dos dados *de facto*, conforme pode ser visto.

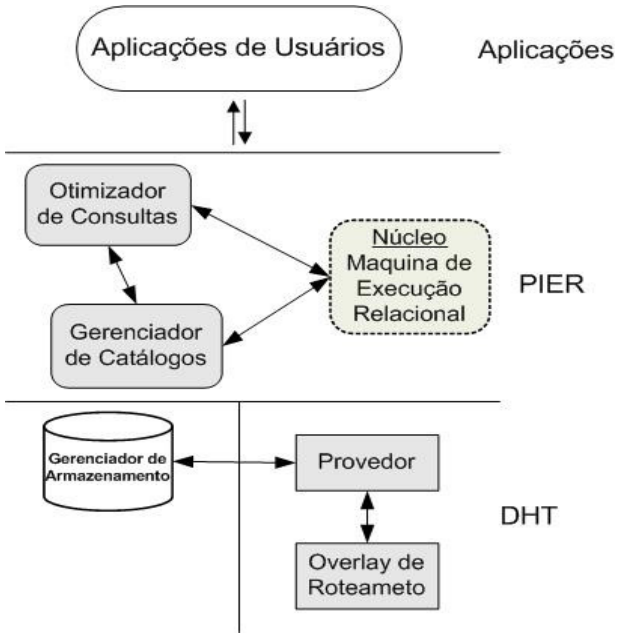

Figura 20. Arquitetura Pier [34]

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.17 RON

RON, ou *Resilient Overlay Networks [7]*, consiste em um projeto oriundo de pesquisas militares (da agência DARPA, *Defense Advanced Research Projects Agency*) e foi concebido por Hari Balakrishnan et al, do MIT. Motivado pelas diversas limitações da Internet quanto a ausência de mecanismos de garantia de qualidade de serviço, técnicas de roteamento pouco flexíveis e baixa tolerância a falhas, este projeto almeja torná-la mais

robusta, resiliente e com reduzidos tempos de recuperação em virtude de falhas nos *links* de comunicação.

Seu aspecto fundamental consiste em desenvolver técnicas que permitam que os nós comunicantes e suas aplicações possam aumentar sua confiabilidade e desempenho cooperativamente. Para isto, nós RON examinam a condição proporcionada pela Internet antes de começar uma comunicação entre si decidindo, então, se utilizará um canal comum (i.e. baseado no roteamento IP tradicional da Internet) ou um canal otimizado para um determinado tipo de tráfego baseado nos nós virtuais RON. Trata-se de uma Rede Virtual de Propósito Específico servindo como infra-estrutura para outros projetos, como por exemplo, o OverQoS, que o utiliza na definição de seus *links* virtuais.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.18 SRF

A Internet, em seu nível lógico, necessita de um serviço de resolução de referências *web* (RRS – *Reference Resolution Service*), que seja capaz de mapear um *link* para seu endereço lógico (endereço IP). Entende-se por referência, no contexto da *web*, URLs que possuam uma estrutura de nomes hierárquica. Sabe-se que no seu estado atual, o DNS impõe limitações à arquitetura da Internet devido a sua estrutura hierárquica e à falta de flexibilidade de seu modelo. Acredita-se então, que um serviço mais flexível baseado em uma infra-estrutura sem hierarquia, apresentaria melhor proficuidade.

Conforme diversos estudos indicaram na literatura, mais especificamente [77] e [78], um RRS para a *web* deve ter pelo menos as duas seguintes características (que não são encontradas no DNS) para identificar adequadamente os objetos da Internet:

- **Referências de objetos persistentes:** Referências não devem conter informações sobre o domínio administrativo ou servidor a qual estão localizadas (como acontece atualmente com o DNS) e devem estar presentes em nível de aplicação, ao invés de representarem um conhecimento do domínio inteiro. Isto permite a migração de conteúdo (por exemplo, a mudança de uma página *web* de uma instituição para outra) de maneira natural, sem requerer, por exemplo, a colaboração da instituição original no encaminhamento HTTP (*Hyper Text Transfer Protocol)* para o novo destino ou mesmo a continuidade da hospedagem do conteúdo.

- **Referências Livres de Contexto**: Como o DNS é capaz de resolver nomes espalhados pelo mundo inteiro, não é incomum que alguns conflitos em relação à referência natural de domínios aconteçam. Como um exemplo clássico, temos os problemas de origem social advindos do uso indevido de nomes ou da disputa judicial pela posse de um nome. A disputa por nomes fáceis de serem memorizados (*human-readable*) é constante. Desta forma, as referências devem ser opacas (e.g. livres de qualquer contexto) e seu mapeamento deve, então, ser feito através de um consulta de nomes *human-readable* traduzidos para referências opacas. Tal separação permite ao RRS livrar-se de questões legais ou sociais e concentrar-se na resolução de nomes.

O projeto SFR (*Semantic Free Referencing*) propõe um novo espaço de nomes (e, conseqüentemente, uma nova camada de resolução de nomes) de referências opacas (não hierárquicas) conhecida pela denominação SFR *tag*, que é capaz de atender às duas premissas acima mencionadas. As referências livres de semânticas, ao contrário dos URLs baseados no DNS, não possuem nenhuma estrutura explícita, um fato que simplifica o método de resolução de nomes que é denotado por um índice (*hash tag index*) baseado na tecnologia de DHTs.

SFR usa DHTs para mapear cada objeto (dado) referenciado a um registro que contenha o metadado do objeto (e.g. seu endereço IP, caminho, informação particular e etc). Uma vez que uma aplicação, tal como um navegador *web*, obtém o metadado referente ao dado procurado, ele é capaz de localizá-lo.

Na Figura 21 é ilustrada a implementação da *web* com o *overlay* de nomeação SFR, em que se torna possível resolver não somente domínios, mas também objetos. Para implementar um sistema de roteamento de referências extensível (*high scalability*), a arquitetura de SFR usa DHTs. Estas mapeiam referências (SFR *Tags*) para registros de objetos (O-*records*), conforme pode ser observado na seta dupla superior da Figura 21. Quando um Cliente *web* deseja obter o SFR Tag de um objeto, este deve obter o *Object Record* (O-record) do destino em uma infra-estrutura de DHT e então fazer uma consulta HTTP padrão ao Servidor *web* responsável.

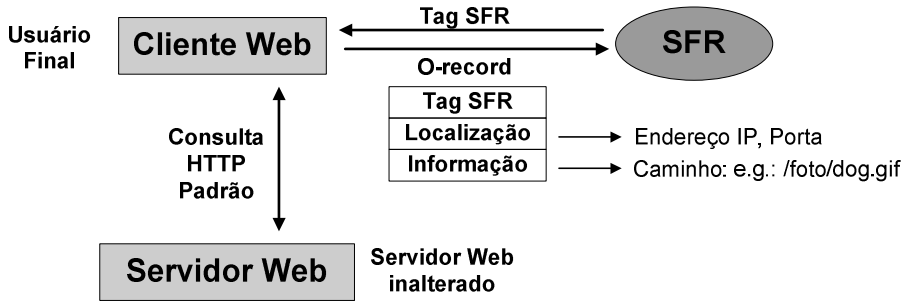

Figura 21. Resolução de Nomes na Web com SFR [14]

Em virtude de o *overlay* SFR utilizar referências opacas (*non human-readable*), o primeiro nível de resolução deve traduzir um nome (e.g. qualquer serviço de canonização de nomes) para uma ou mais referências. Ferramentas de busca devem, portanto, mapear nomes para índices SFR, contendo a localização e o caminho para atingir o dado.

A flexibilidade de se associar múltiplos endereços e caminhos para um *o-record* possibilita a replicação de conteúdos e a mudança de sua localização, enquanto a canonização de SFR *Tags* (opacas) em nomes não significativos (sem informações sobre seu contexto) é capaz de evitar conflitos legais pela propriedade de nomes. Além disso, não se fazem necessárias modificações na infra-estrutura atual da Internet (e.g. serviços *web* ofertados), pois a resolução SFR *tag* é transparente para estas aplicações.

Uma implementação *web*, que utilize o *overlay* de nomeação SFR ao invés do DNS, é capaz de se beneficiar das seguintes características:

- ***Links* persistentes e livre migração** para evitar a necessidade de propagar mensagens relacionadas a mudanças na localização da referência;
- **Flexibilidade de replicação** que permite acomodar conteúdos de diferentes indivíduos sem qualquer necessidade de caracterização do mesmo (e.g. conteúdos específicos relacionados a um domínio particular).
- **Serviço de ponteiros confiável** que possibilite aos indivíduos criar *links* que apontem para outros *links* ao invés de apontar diretamente para seus objetos e
- **Extensibilidade e generalidade** em virtude da generalidade do modelo SFR e da interface independente de aplicação. Outras aplicações além de navegadores *web* (e.g. clientes de *email*, aplicações *peer-to-peer*) poderiam utilizar referências baseadas no modelo SFR.

A solução descrita pelo SFR é considerada mais poderosa do que a solução corrente da Internet baseada no DNS no que tange à persistência de *links.* Ao contrário do DNS, o SFR pode resolver igualmente o *tag* e o caminho para alcançar o objeto antes que qualquer mensagem HTTP seja emitida aos servidores Web.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Non-Routing.**

4.1.19 Skype

Objetiva disponibilizar uma infra-estrutura de Rede Virtual *Overlay* baseada em redes P2P para permitir que usuários possam se comunicar utilizando recursos de voz pela Internet. O Skype [69] foi concebido por N. Nennström e J. Friis.

Em sua comunicação proporciona garantias mínimas de segurança utilizando um serviço de chaves pública/privada aliado à criptografia não opcional em suas comunicações (usa o algoritmo AES-256, *Advanced Encryption Standard,* obtendo suas chaves através de uma negociação simétrica baseada no RSA-1024).

- Classificação segundo a Taxonomia proposta: **Rede Virtual ASNe Overlay Routing.**

4.1.20 TRIAD

Um acrônimo para *Translating Relaying Internet Architecture integrating Active Directories,* TRIAD consiste na proposição de uma nova arquitetura para a Internet que define uma Rede Virtual *Overlay* para permitir o roteamento de conteúdo em larga escala, o uso de mecanismos de *caching*, a transformação de conteúdo, e o balanceamento de carga integrando nomeação, roteamento e transporte em uma estrutura unificada (i.e. *Active Directories*).

A identificação fim-a-fim é baseada no uso de nomes e URLs (*Universal Resource Locators*) de forma que os endereços IP ficam restritos ao papel transitório do encaminhamento de *tags* (identificadores TRIAD). Nesta camada de identificação de conteúdo, um eficiente diretório integrado em cooperação com o roteamento e com o estabelecimento de conexões promove um encaminhamento otimizado de conteúdo,

balanceamento de carga e diminuição, na maioria dos casos, dos tempos de resposta (*round trip times)* de acesso ao conteúdo.

TRIAD implementa um *overlay* de nomeação que suporta um endereçamento baseado na localização do conteúdo através do protocolo WRAP (*Wide-Area Relaying Protocol*). O mesmo foi projetado para atuar sobre a camada de rede, mais especificamente acima do IPv4, para permitir o roteamento baseado no conteúdo e a nomeação de uma infra-estrutura de larga escala com o objetivo de evitar a necessidade da transição da Internet para o IPv6. Dois tipos principais de roteadores foram propostos: Roteadores de Conteúdo (CR – *Content Routers*), que são responsáveis pela integração do roteamento com a nomeação dos objetos e Roteadores de Resolução de Conteúdo (CRR – *Content Resolving Router*), que realizam resoluções análogas ao DNS.

No TRIAD, a camada de nomeação e o roteamento estão intrinsecamente integrados de três maneiras fundamentais. Primeiramente, os roteadores CRR devem realizar a tarefa de resolução de nomes (*name lookup service*), ao invés de um servidor possivelmente localizado fora do caminho (i.e. domínio) de resolução. Em segundo lugar, a consulta (i.e. o *lookup*) realizada pode retornar uma rota ou uma especificação do trajeto ao requerente. O protocolo WRAP apresenta além do endereço de origem e de destino do pacote IP, uma especificação de como realizar o trajeto ao conteúdo desejado da melhor forma, ou seja, permitindo-lhe a escolha de certos parâmetros de qualidade de serviço.

Similar a outros protocolos de encapsulamento IP, o cabeçalho de transporte assim como os dados são acomodados na área de dados (*payload*) do pacote IPv4. O cabeçalho também contém um par IRT (*Internet Relay Tokens*) que mapeia os sinalizadores (*tokens*) de encaminhamento direto e reverso da resolução de nomes. O *token* reverso indica o trajeto já percorrido pelo pacote até o presente ponto e o *token* direto (de encaminhamento) representa o trajeto que o pacote irá realizar. Na Figura 22 o formato do pacote WRAP com seus respectivos campos é ilustrado.

Figura 22. Formato do Protocolo WRAP [70]

Finalmente, o sistema de roteamento precisa estar fortemente acoplado ao diretório no roteador da seguinte forma: a tabela de roteamento deve mapear nomes para *next-hops* (próximo *host* de encaminhamento) ao invés de mapear endereços para *next-hops*. Desta forma, o roteamento identifica os *hosts* e *next*-hops baseado em seus nomes e propaga estas informações para outros roteadores da Internet. Um exemplo de uma resolução de nomes na arquitetura do TRIAD/WRAP é mostrado na Figura 23.

O serviço de diretório do TRIAD age como uma extensão redundante do DNS atual, fornecendo uma elevada disponibilidade no mapeamento de nomes para endereços. Cada equipamento de borda (i.e. N*ext-Hops*) age como um servidor de tradução de nomes DNS capaz de resolver parcialmente as consultas.

Pedidos de resolução de nomes são dirigidos inicialmente a um equipamento de borda, como demonstrado na Figura 23. As informações de roteamento são então armazenadas nos registros dos NH - *Next-Hops* intermediários (tal como explicitado na Figura 23 – de A para A' ou de B para B'), que especificam qual o próximo passo a ser seguido (i.e. próximo NH) para atingir um servidor DNS superior. Como resposta a uma consulta de nomes, um conjunto de possíveis rotas pode ser fornecido à origem. Cada agente de encaminhamento NH (ou seja, A, A', B ou B') pode modificar o resultado da resposta para que a mesma seja consistente para a Origem Ori.

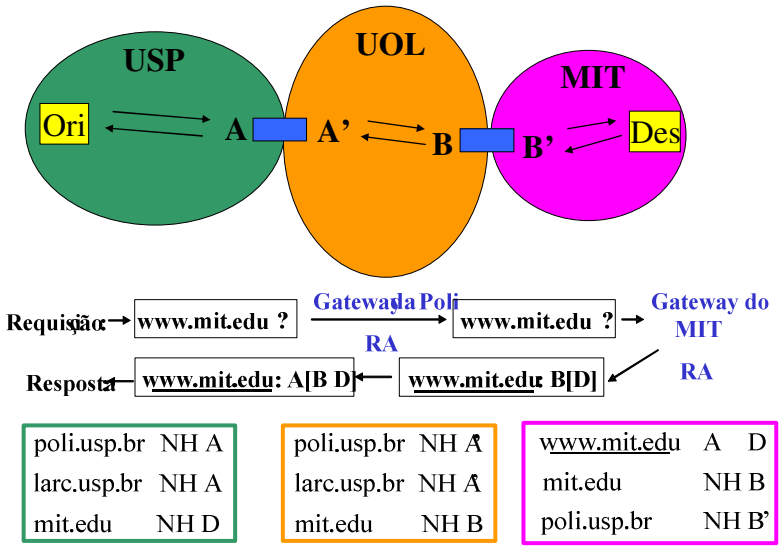

Figura 23. Resolução de Nomes na Arquitetura TRIAD/WRAP

A arquitetura TRIAD/WRAP se propõe a resolver alguns dos problemas associados à Internet. É capaz de implementar um espaços de nomes unívoco, balanceamento

de carga, mobilidade, suporte a redes virtuais privadas, qualidade de serviço de conteúdo associada ao roteamento e à autenticação. Além disso, como é esperado de um ambiente *overlay*, sua adoção pode ser feita de forma incremental, inicialmente sem mudanças adicionais aos nós finais (e.g. computadores de propósito geral) ou aplicações (entende-se por mudanças adicionais, modificações mais exigentes do que as já requeridas pelo uso do NAT).

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.1.21 X-BONE

Este projeto, baseado em Redes Virtuais *Overlay*, tem por objetivo minimizar os esforços requeridos na customização e no compartilhamento de recursos na Internet através da combinação de técnicas de monitoração, *multicasting* e roteamento.

Utiliza diversos protocolos de monitoramento implementados como *deamons* em nós virtuais que se comunicam através de um tunelamento IP. Pode proporcionar um isolamento total entre redes *Overlay* garantindo, inclusive, uma reserva de banda (baseados no RSVP [75]) entre elas.

Foi idealizado por J. Touch et al da agência militar DARPA.

- Classificação segundo a Taxonomia proposta: **Rede Virtual PSNe Overlay Routing.**

4.2 Análise dos Resultados Segundo a Taxonomia Proposta

Após a inspeção de uma grande variedade de Redes Virtuais, aqui parcialmente coberta pelos projetos supracitados, uma tabela comparativa foi concebida para clarificar a distribuição de projetos analisados, pontuando cada um deles dentro de suas classes taxonômicas de atuação: Redes Virtuais PSNe ou ASNe; *Overlays* ou *Underlays* e *Routing* ou *Non-Routing*. Na Tabela 5 pode ser verificada a ocorrência e classificação baseadas na Taxonomia de Redes Virtuais proposta de todos os projetos aqui tratados.

Tabela 5. Classificação Taxonômica de Redes Virtuais

ASNe	Overlay	Underlay	Routing	Non-Routing
BitTorrent	x		x	

CoDeeN	x		x	
Coral	x		x	
Coblitz	x		x	
F5	x		x	
Joost	x		x	
Pier	x		x	
Skype	x		x	
PSNe	**Overlay**	**Underlay**	**Routing**	**Non-Routing**
Active Networks	x		x	
FARA	x			x
HIP	x			x
Hi3	x			x
I3	x		x	
IPNL	x		x	
Network Pointers		x	x	
OverCast	x		x	
OverQoS	x		x	
RON	x		x	
SFR	x			x
TRIAD	x		x	
X-BONE	x		x	

A partir da análise conduzida neste capítulo e posterior abstração de conceitos na forma da Tabela 5, algumas conclusões podem ser delineadas. A taxonomia de Redes Virtual correntemente estudada possibilitou a definição e classificação de diversos projetos candidatos elucidando a multiplicidade de funcionalidades factíveis de serem implementadas na Internet, bem como servindo de motivação para uma escolha mais acurada de características a serem propostas nesta dissertação, influenciando sobremaneira as discussões do Capítulo 5.

O posicionamento destas propostas de Redes Virtuais em suas categorias de atuação taxonômicas permite que critérios mais adequados de comparação sejam feitos, garantindo a propriedade de análises comparativas de pesquisas (ou seja, estabelecendo métricas formais para a realização de classificações e comparações) reduzindo os tempos de

investigação literária necessários. A classificação unívoca de todos os candidatos dentro de classes distintas ilustra a ortogonalidade na escolha dos táxons, que corrobora a usabilidade desta taxonomia.

CAPÍTULO 5 PROPOSTA DE UMA ARQUITETURA DE NOMEAÇÃO PARA A INTERNET UTILIZANDO REDES VIRTUAIS

O objetivo deste trabalho é a proposição de uma arquitetura com múltiplas camadas de nomeação de objetos para a Internet, e consiste no documento principal desta dissertação. Para este fim, diversas implementações e propostas de espaços de nomes hierárquicos e não-hierárquicos foram estudadas e avaliadas conforme retrataram os capítulos três e quatro. Segundo Stoica (2004) [16], um sistema com múltiplas camadas de resolução, conforme visto na Figura 2, se faz necessário para a composição de uma arquitetura "ideal" para a Internet. Nele, nota-se claramente que a pilha de resolução de nomes (tendo-se em vista a arquitetura TCP/IP) é aumentada com novas camadas de nomeação, conforme pode ser visto na Figura 24, com as camadas Resolução de Serviço e Resolução de Nós.

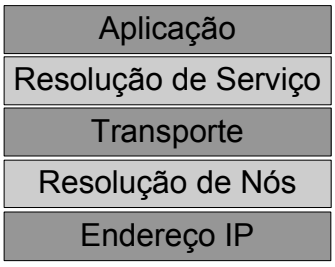

Figura 24. Pilha de resolução de nomes da Arquitetura Proposta

A proposição desta nova arquitetura de nomeação é regida por uma série de princípios que serviram de fundamentação para a proposição da mesma e suscitaram, diante da discussão de suas funcionalidades, a elaboração mais detalhada dos requisitos que deveriam ser compreendidos.

5.1 Princípios

A definição dos princípios arquiteturais realizada neste trabalho pode ser interpretada como uma tentativa de recobrar certas funcionalidades do complexo e completo modelo OSI almejando uma proposta genérica de infra-estrutura de nomeação de redes. O primeiro passo nesta direção consistiu da definição dos atributos e dos objetos que serão

compreendidos pela arquitetura, e neste sentido, certos requisitos aqui destacados representam uma cópia *verbatim* de trabalhos anteriores [79; 17; 16].

Segundo Saltzer (1993) [17], os seguintes objetos podem ser destacados: **dados, serviços e usuários; nós; pontos de acesso à rede e caminho.** Dados e serviços representam as funcionalidades disponibilizadas pela rede e usuários são os que se beneficiam do seu usufruto; **nós** consistem nos computadores que podem armazenar dados, executar serviços e suportar usuários; **pontos de acesso à rede** são portas de acesso a uma rede, ou seja, o local onde um nó se encontra conectado e **caminho** está compreendido entre pontos de acesso à rede, atravessando nós de encaminhamento e *links* de comunicação.

No caso de dados, serviços, usuários e nós, três principais atributos devem ser associados, segundo Shoch (1979) [79]: um **nome**, um **endereço** e uma **rota**. O primeiro deve permitir a identificação de um objeto (e esta deve ser singular); o segundo, o endereço, deve identificar a localização do mesmo (e este é susceptível a mudanças); e, finalmente, a rota irá identificar a concatenação dos caminhos factíveis para se atingir o objeto destino (também susceptível a mudanças).

Inspirado nos frutos dos trabalhos de Shoch e Saltzer como base para a enumeração de princípios diretivos para a proposição de uma arquitetura de nomes, Stoica *et al* [16] propuseram três princípios que servem como guia para o entendimento deste trabalho. Para cada princípio, uma breve explicação será delineada ilustrando a aplicabilidade para o cenário atual.

Princípio 1# *Nomes devem remeter aos protocolos somente os aspectos fundamentais da infra-estrutura de suporte. O estabelecimento de vínculos para detalhes irrelevantes acarreta na limitação da flexibilidade e de funcionalidades.*

Atualmente, requisições de serviço ou dados feitas por aplicações objetivam receber como resposta uma identidade (no caso do serviço) ou um conteúdo (no caso dos dados). Entretanto, nomes baseados no DNS são traduzidos para o endereço IP do nó destino, remetendo à aplicação uma localização particular. Esta resolução viola, portanto, **o Princípio 1#** duas vezes consecutivas: primeiramente, ela associa serviços e dados a um nó em particular e, ainda mais crítico, os associa também à localização deste nó.

A proposta inicial do uso de nomes DNS (segundo a RFC 1498 [17]) para criar um espaço de nomes com semântica associada aos nós, se encontra agora, sendo utilizada para identificar com alta granularidade, também os serviços e dados (vale ressaltar que, sem o sacrifício da semântica, os dados também podem ser incluídos nesta lista) e os associa a um nó erroneamente. Estes vínculos sobrecarregam a semântica de comunicação na medida em que causam um sério número de limitações, como por exemplo, o impedimento natural de mobilidade de dados e serviços, a replicação de conteúdo, entre outros.

O **Princípio #1** requer que as aplicações sejam capazes de se remeter a dados e serviços através de nomes persistentes que não sejam associados ao nó que os hospeda. Para isso, uma camada de Resolução de Serviços (**SID**) deve existir para garantir às aplicações esta habilidade, conforme proposto na Figura 24. Assim, é de fundamental importância, a infra-estrutura de múltiplas camadas de nomes persistentes para este trabalho.

Da mesma forma, protocolos de transporte realizam trocas de dados entre dois nós, e suas localizações de rede são irrelevantes em relação à semântica básica de conexão de transporte. Fica claro que somente na camada de rede é que o endereço IP faz parte natural da semântica do protocolo, o que remete à necessidade de um sistema de resolução de nomes de endereços de rede exclusivamente para endereços de rede, evitando que o protocolo de transporte estabeleça qualquer vínculo com o endereço IP. Neste contexto, insere-se a segunda camada de resolução (de baixo para cima) ilustrada na Figura 24. Nela deverá ser proposto um espaço de nomes para identificar elementos de acesso à rede (identificação de nós) sem remeter/estabelecer qualquer vínculo adicional sobre sua localização topológica.

Princípio 2# *Nomes devem ser persistentes, não podendo estabelecer restrições arbitrárias nos elementos a que referem.*

Como um contra-exemplo deste princípio, temos o espaço de nomes global do DNS e o do endereçamento IP da Internet. No último caso, temos o desempenho do roteamento diretamente vinculado à estrutura de endereçamento IP e, no primeiro caso, nomes que refletem características do domínio administrativo no qual estão sendo hospedados. Conforme mencionado anteriormente, a identificação granular de serviços e dados, bem como a mobilidade ou replicação ficam prejudicadas neste modelo.

Somando a isso, se analisarmos a natureza dos nomes e a dinâmica de sua utilização perceberemos que as pessoas procuram mudar seus nomes de forma freqüente, haja

vista que os mesmos estão intimamente associados a elas ou aos serviços que designam. Desta forma, a natureza embutida de semântica das referências URLs realmente não deve sofrer alterações constantes permitindo um acesso facilitado e sem interrupções. A mesma analogia pode ser feita para os dados, conforme ressalvas foram feitas anteriormente e sem a perda da semântica. Nota-se, então, a importância de desassociar a localização dos nomes e a importância deste princípio.

Neste trabalho, faz-se necessário o uso de referência livre de qualquer tipo de semântica e, portanto, vislumbra-se um espaço de nomes sem qualquer hierarquia e com referências opacas. Obviamente, um serviço de canonização será necessário na associação de nomes com semântica para este novo espaço de nomes e ferramentas de busca devem ser capazes de realizar tal tradução (análogo ao cenário que observamos hoje com as buscas e localizações feitas por máquinas de buscar para o espaço de nomes do DNS).

Princípio 3# *Uma entidade de rede deve ser capaz de realizar resoluções de nome não somente de sua localização particular, mas também de seus representantes.*

Dada uma requisição de conexão (e.g. aplicações conectadas com identificadores de serviço SID) para certo nó destino, o mesmo deve ser capaz de redirecionar tal requisição para um representante (entidade delegada) no caso de não poder atendê-la ou por objetivos de desempenho. Esta característica implica simultaneamente na manutenção das propriedades das entidades originais em relação às entidades delegadas (e.g. aplicações requisitantes devem possuir igual confiança nas entidades representantes), bem como na inserção natural de intermediários (*middleboxes*) na Internet. Um exemplo prático consiste no suporte natural de balanceadores de carga que poderiam agora através de uma única URL, mapear de forma transparente, este conteúdo a múltiplas localidades através da delegação a seus representantes (para dividir o número de acessos a determinados conteúdos).

Outras formas de *middleboxes* também passam a ser suportadas como redirecionadores *proxy*, autenticadores e controladores de acesso entre outros, pois agora (com a junção dos três Princípios) a arquitetura passa a ter controle completo das interações entre os pares comunicantes e da relação entre eles. Do ponto de vista da Arquitetura de Nomes e sob a luz do **Principio 3#**, um nome agora (sem nenhuma restrição semântica ou características irrelevantes embutidas à sua natureza), é capaz de permitir que uma cadeia de informações seja associada a si, permitindo que os **Sistemas de Resolução de Referências**

(RRS) a utilize para formar um novo e rico espaço de nomes. Nele, mudanças nas "informações associadas" aos nomes não mais afetam quaisquer operações de rede. Sendo mais pragmática, informações quanto à qualidade de serviço de conteúdos (i.e. nomes) ou quanto à legitimidade do mesmo ou de qualquer outra natureza comercial podem ser utilizadas e modificadas enriquecendo o espaço de nome sem restringi-lo de qualquer forma.

Princípio 4# *Visar a compatibilidade com sistemas legados consiste em característica fundamental para o sucesso e a adoção em larga escala das novas infra-estruturas.*

A nova arquitetura de nomeação deve ser capaz de coexistir em harmonia com a infra-estrutura legada, ainda que por um tempo parcial até que sua adoção venha a ser completa, para garantir que sua proposição seja viável. O **Principio 4#** visa a adoção de ferramentas compatíveis e incrementais, conforme largamente discutido nos assuntos sobre Redes *Overlay*. Além disso, a inserção de funcionalidades em substituição ao modelo atual da Internet, que pudessem causar sua indisponibilidade mesmo que por um curto período, não representa uma idéia atrativa nem do ponto de vista dos fabricantes nem dos usuários, ainda que as mudanças sejam benéficas.

5.2 Análise de Requisitos

Fundamentado na definição anterior dos quatro princípios para a definição de uma Arquitetura com Múltiplas Camadas de Nomeação para a Internet, nos requisitos necessários para acomodar os conceitos de computação pervasiva, computação ubíqua e nos requisitos de projetos de aplicação em larga escala, como o Ambiente Networks (projeto Europeu de integração de soluções para a Internet pertencente ao *framework* FP6) [80], uma análise de requisitos será confeccionada abaixo. Acompanhado de cada requisito geral, uma sucinta justificativa lhe segue.

Requisito Geral 1# Identificação unívoca de dados/serviços/usuários independente da sua localização ou controle administrativo (domínio), ou em relação ao nó que o hospede ou ao seu ponto de acesso à rede (endereço IP).

Requisito Geral 2# Identificação singular e sem hierarquia de nós, independentemente dos serviços que hospedem dados e usuários associados, localização e controle administrativo ou ponto de acesso à rede.

Requisito Geral 3# O novo espaço de nomes para a identificação de dados/serviços/usuários e não deve ser hierárquico ou impor qualquer restrição nos nomes/referências sendo este, portanto, um espaço de nomes livre de semântica.

Requisito Geral 4# Deve possuir a habilidade de mapear nomes provenientes de um espaço de nomes com semântica (e.g. objetos, marcas, etc) para um espaço de nomes livre de semântica.

Requisito Geral 5# O espaço de nomes de dados/serviços/usuários e nós deve possuir a propriedade de ser criptograficamente verificável permitindo que o proprietário de um nome consiga provar sua propriedade resultando em uma autenticação mútua de elementos, ou seja, a autenticidade de uma associação <nome → objeto> deve ser criptograficamente (através de mecanismos, como por exemplo, de chaves públicas/privadas) garantida.

Requisito Geral 6# Como conseqüência do **Requisito Geral 5#,** mecanismos para garantir a troca de chaves criptográficas após a autenticação devem existir para garantir que o fluxo de comunicação seja confidencial e autenticável.

Requisito Geral 7# Nomes de dados/serviços devem ser capazes de codificar (além dos aspectos relevantes à sua natureza) informações sobre a qualidade de serviço do conteúdo (QoSC) que poderá servir de informação para a infra-estrutura base (e.g. de roteadores) atingir um determinado elemento em particular.

Requisito Geral 8# Todas as funcionalidades aqui descritas devem ser implementadas de forma incremental e sem causar nenhuma mudança na estrutura de roteamento ou nomeação (com o DNS como um RRS). Modificações devem ser inseridas somente nos nós finais comunicantes e com a adição de novos RRS.

De posse dos oito requisitos supracitados, dos projetos considerados no capítulo quatro, e as análises conduzidas até agora, este trabalho foi estruturado como uma síntese de características originais para a criação do sistema de nomeação com múltiplas

camadas. Como uma conseqüência direta da criação de novos espaços de nome, da necessidade de autenticação mútua de elementos e do suporte a novas funcionalidades, como a QoSC, os Requisitos de Resolução puderam ser descritos.

Requisitos de Resolução 1# Escalabilidade, disponibilidade e gerenciamento. O novo sistema deve ser capaz de resolver um grande número de requisições (da ordem no número de requições do DNS atual) com a mesma escalabilidade e disponibilidade (é desejável que seja melhor) que o RRS atual (i.e. DNS). Deve ser gerenciável do ponto de vista que nenhuma entidade central de administração deve ser requerida no registro (i.e. nas chegadas ou partidas) de servidores na infra-estrutura.

Requisitos de Resolução 2# Deve ser capaz de operar em paralelo com o RRS atual (DNS). Desta forma, consultas de nomes provenientes do espaço de nomes hierárquicos (e.g. URLs) devem ser encaminhadas para o DNS enquanto as consultas provenientes dos espaços de nomes sem hierarquia devem ser encaminhadas para o novo RRS.

Requisitos de Resolução 3# Deve oferecer soluções para realizar o mapeamento de quatro espaços de nomes: com semântica, de serviço sem semântica e sem hierarquia, de nós sem semântica e sem hierarquia e o espaço de nomes atual dos endereços IP, conforme a pilha de resolução de nomes da Arquitetura proposta e ilustrada na Figura 24.

Requisitos de Resolução 4# Presença de uma Entidade de Certificação (CA) que realize a autenticação de todos os elementos almejados.

Requisitos de Resolução 5# Integração de uma ou mais estruturas de auditoria na disponibilização de informações de QoSC (em relação aos nomes).

Diante da inspeção destes Requisitos Gerais de Resolução, especialmente o **Requisito de Resolução 3#** que visa utilizar espaços de nomes sem hierarquia distribuídos com a necessidade de resolução em tempo reduzido, a melhor, senão a única solução atual consiste no uso de DHTs conforme Kumar (2005) [26].

5.3 Descrição Geral

O principal aspecto da nova Arquitetura de Nomes consiste na criação de dois novos espaços de nomes para a identificação de dois conjuntos de objetos: Dados/Serviços/Usuários (em um espaço de nomes) e Nós (em um segundo espaço). Utilizando os termos cunhados por Stoica *et al* [16], o primeiro conjunto de objetos origina o espaço de nomes designado pelo identificador SID (*Service Identifier,* ou Identificador de Serviços) e o segundo, representado pelos nós, origina o espaço de nomes designado pelo identificador EID (*Endpoint Identifier,* ou Identificador de Nós). Além disso, não se pode deixar de considerar o espaço já existente dos Pontos de Acesso à Rede, que atualmente é representado pelo endereçamento IP e que deve permanecer inalterado, porém agora, atuando de forma singular sem a sobrecarga de funções (de nomeação e endereçamento), conforme almejado no início deste trabalho.

Os efeitos destas mudanças na descrição de uma conexão podem ser classificados como profundos e tangíveis. No modelo atual, um objeto da camada de aplicação (i.e. um dado, um serviço ou um usuário) é associado com uma URL que por sua vez é associada com um endereço IP. Diferentemente, no novo modelo aqui proposto, um objeto da camada de aplicação é associado a um SID, isto é, aponta para o SID corrente que está localizado em um nó específico. Este nó, por sua vez, esta associado ao seu(s) EID(s) que está vinculado ao seu endereço IP.

Desta forma, mudanças no endereço IP do nó somente acarretarão em uma atualização das funções de mapeamento (EID, IP). Igualmente, alterações nas propriedades dos objetos da camada de aplicação, do ponto de vistas de dois nós comunicantes, só acarretará em modificação nas funções de mapeamento (SID, EID) permitindo que tais alterações sejam transparentes para os usuários. Percebe-se que agora, os usuários não precisam se preocupar mais com detalhes "irrelevantes" da infra-estrutura. Além disso, vale observar a aderência desta infra-estrutura com os **Requisitos Gerais 1#** e **2#**, plenamente satisfeitos conforme proposto.

Como a mudança/inserção de identificadores/referências acarretará, obviamente, em uma cadeia de novos mecanismos de resolução para acessar os objetos da camada de aplicação (i.e. seu endereço), a infra-estrutura deverá possuir novos RRS para a resolução dos nomes, conforme representado na pilha de espaço de nomes da Figura 25.

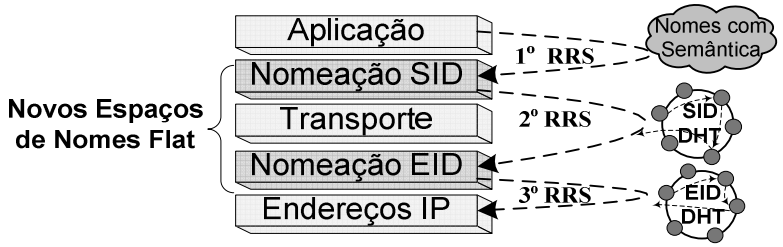

Figura 25. Nova Pilha de Resolução de Nomes e Interconexões

Analisando a Figura 25, para manter a concordância com o **Requisito Geral 3#** (ausência de semântica nos novos espaços de nomes), as duas novas camadas EID e SID serão resolvidas por dois RRSs implementados em DHTs (em observância aos **Requisitos de Resolução 1#** e **2#**). A DHT SID irá mapear um SID para uma lista de EIDs, enquanto a DHT EID irá mapear um EID para uma lista de endereços IP. Além disso, um terceiro RRS se faz necessário para fazer um mapeamento canônico entre os nomes com semântica para as referências SID (sem semântica), livrando as pessoas da penosa tarefa de decorar os nomes opacos (i.e. sem semântica) e de certa forma extensos do SID. Esta funcionalidade torna este sistema completamente compatível com o **Requisito de Resolução 3#**.

5.4 Componentes do Sistema

Remetendo-se aos elementos discutidos na introdução sobre os aspectos desejáveis na Internet atualmente, diversos componentes podem ser definidos na Arquitetura Com Múltiplas Camadas de Nomes. Esta etapa é de fundamental importância para a definição detalhada dos requisitos necessários para cada componente arquitetural. Com o intuito de prosseguir nesta análise, uma ilustração de um processo de comunicação será apresentada na Figura 26 onde dois nós comunicantes inseridos nesta nova Arquitetura de Nomes trocam informações. Neste cenário, novos elementos previamente discutidos são finalmente incluídos, tais como a Entidade de Certificação (CA) e a Entidade de Auditoria e Qualidade de Serviço de Conteúdo – QoSC, em acordo com os **Requisitos Gerais 5#, 6# e 7#**

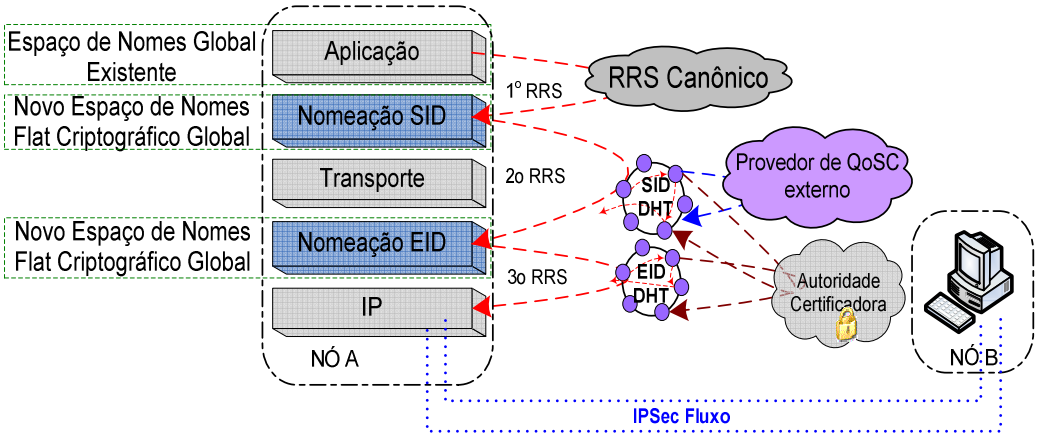

Figura 26. Resolução de Nomes e Comunicação fim-a-fim na Arquitetura de Nomes com múltiplas camadas

Baseado nas discussões anteriores e na modelagem de comunicação da Figura 26, um levantamento de componentes intrínsecos da Arquitetura Com Múltiplas Camadas de Nomes foi estabelecido:

Componente a. *Endpoint* (nó A e nó B).

Componente b. Espaço de Nomes Canônicos (com semântica).

Componente c. Subsistema de Resolução Canônica de Nomes Semânticos.

Componente d. Espaço de Nomes SID (Camada de Nomeação SID).

Componente e. Subsistema de Resolução SID (RRS SID em DHT).

Componente f. Espaço de Nomes EID (Camada de Nomeação EID).

Componente g. Subsistema de Resolução EID (RRS EID em DHT).

Componente h. Autoridade de Certificação (CA do **Requisito Geral 5#**).

Componente i. Provedores de Qualidade de Serviço de Conteúdo e Auditoria (QoSC do **Requisito Geral 7#**)

Componente j. Fluxo IPSec Criptografado (do **Requisito Geral 6#**)

Ainda que alguns componentes tenham sido superficialmente abordados, uma descrição consolidada de cada objeto será delineada abaixo para o melhor entendimento do leitor.

Componente a. *Endpoint* consiste em qualquer tipo de equipamento adequado para ser empregado por usuários no acesso a um objeto da camada de aplicação. O termo adequado aqui se remete a qualquer equipamento (i.e. uma estação de trabalho, um laptop ou até um PDA) com a capacidade de executar aplicações clientes (sejam elas aplicações web ou cliente-servidor e etc). Também deve ser capaz de interagir com as aplicações ordinárias (i.e. disponibilizadas como serviços pelos servidores) situadas agora em uma arquitetura TCP/IP "aperfeiçoada" (ou seja, modificada pelos requisitos supracitados norteadores da nova Arquitetura de Nomes).

Componente b. Espaço de Nomes Canônicos (com semântica) podem ser entendidos como um conjunto de referências (e.g. nomes pessoais, marcas registradas, apelidos, codinomes e etc) utilizado por usuários tanto para localizar uma aplicação ou dado quanto para permitir que aqueles procurem um conteúdo ou aplicação em uma ferramenta de busca (e.g. Google [81], Altavista [82], Yahoo [83]). Análogo ao que ocorre atualmente necessitar-se-á que uma terceira entidade possa colaborar na busca e recuperação de nomes (semânticos), como propõe o **Componente c.** Subsistema de Resolução Canônica de Nomes Semânticos. Este seria capaz de exercer funções de mapeamento de um espaço de nomes sem restrição (composto por qualquer tipo de nome, com ou sem semântica associada) formado pelo **Componente b.** para um conjunto finito de referências *flat* SID (i.e. sem hierarquia) formado pelo **Componente d.**.

Componente d. Espaço de Nomes SID (Camada de Nomeação SID) consiste no conjunto de termos/referências que identifica univocamente objetos da camada de aplicação independentemente da sua localização (i.e. a localização é passível de ser alterada sem que isso acarrete em qualquer modificação ao objeto referenciado). O mapeamento da localização do objeto é realizado, então, pelo **Componente e.** Subsistema de Resolução SID que realiza uma consulta a uma DHT, a partir da referência *flat* obtida pela resolução do **Componente c.** e obtém o(s) respectivo(s) EID do(s) nó(s) procurado(s).

Analogamente, um processo similar acontece com o **Componente f.** Espaço de Nomes EID (Camada de Nomeação EID), que compreende um espaço de nomes sem hierarquia capaz de identificar univocamente um nó independente da sua localização ou mesmo dos objetos da camada de aplicação correntes (i.e. ambos podem mudar sem que nenhuma alteração ocorra ao identificador EID). Da mesma forma, o mapeamento aqui é realizado por funções do **Componente g.** Subsistema de Resolução EID que a partir de um

EID é capaz de retornar um conjunto de endereços IP (ou outros endereços/protocolos de acesso à rede) pela consulta à DHT correspondente.

A Autoridade Certificadora (CA), representada pelo **Componente h.**, permite que os nós comunicantes verifiquem suas identidades criptográficas através de um repositório de certificados garantindo ou negando sua autenticação mútua. O **Componente i.** Provedor de QoSC e Auditoria justifica sua existência baseado na premissa de que a informação disponibilizada (i.e. o *metadado*) pelo proprietário do objeto (i.e. o dado) não deva ser confiável. Este fato é facilmente explicável devido à possibilidade do mesmo em alterar os índices de qualidade do ambiente almejando atrair ilicitamente mais usuários para sua aplicação.

Desta forma, para avaliar a veracidade e a transparência das informações disponibilizadas para todo o sistema, uma terceira entidade competente se faz necessária na garantia da confiabilidade dos *metadados*. Do ponto de vista dos provedores de serviços, este modelo de negócios tipicamente gera uma grande sinergia de interessados, na medida em que os mesmos podem confiar nos índices aferidos aos seus objetos. De forma reativa, do ponto de vista dos consumidores de serviço, uma grande expectativa é criada sobre a acuracidade das informações de QoSC publicadas aumentando os níveis de exigência do sistema.

Finalmente, o **Componente j.** Fluxo IPSec Criptografado representa uma escolha estratégica do sistema. Primeiramente, fica clara a ausência de qualquer alteração no processo de comunicação de objetos de aplicação, haja vista, a não modificação do mecanismo de roteamento tradicional (o do modelo TCP/IP). Como pôde ser observado na cadeia de resolução de identificadores, o passo final consiste na obtenção do endereço IP, fato que permite aos usuários acessar os serviços/dados através dos protocolos convencionais de roteamento da Internet.

Baseado nos componentes até aqui detalhados e, principalmente, na última característica observada do **Componente j.** (a não modificação das funções de roteamento), pode-se classificar esta Aquitetura de Nomes sem Hierarquia Com Múltiplas Camadas como uma **Rede Virtual** do tipo **ASNe**, particularmente com o propósito específico de implementar uma infra-estrutura de nomeação (e, por conseqüência dos **Componentes e, g, h,** e **j**, também como de segurança e mobilidade). Em virtude de esta implementar sua funções virtuais acima da camada IP, a mesma se apresenta na sub-divisão dos **Overlays** e na classe de **Non-Routing** (conseqüência da manipulação de metadados e das propriedades vistas do Componente j.).

- **Taxonomia da Arquitetura de Nomeação Com Múltiplas Camadas:** *Rede Virtual, PSNe (nomeação, segurança e mobilidade), Overlays, Non-Routing.*

Esta Arquitetura apresenta uma característica fundamental advinda de sua natureza de rede *Overlay* que lhe permite uma implementação incremental (conforme o **Requisito de Resolução 2#**) ainda permitindo que futuras modificações sejam feitas à infra-estrutura de roteamento (as vantagens/desvantagens desta possível modificação não fazem parte do escopo deste trabalho). Em verdade, a única característica adicionada diretamente à rede IP consiste na troca de chaves de sessão criptográfica que, através da mútua certificação, habilitam a confidencialidade diretamente na camada IP utilizando os recursos consagrados (i.e. já documentados e padronizados) do protocolo IPSec [76].

5.5 Cenários Operacionais

Nesta seção serão descritos os detalhes do sistema em funcionamento bem como da interação entre os componentes quando em operação. Uma breve descrição dos papéis desempenhados pelos participantes permitirá agrupá-los em duas categorias principais: a dos proprietários dos objetos da camada de aplicação e a dos usuários destes objetos. Obviamente, a relação entre os proprietários e usuários em uma arquitetura de nomes como esta é relativamente fluída, podendo-se admitir, de fato, dois papéis simultâneos para a maioria dos participantes. A partir desta categorização primária, três principais classes de operações são delineadas:

- **Cenário I:** Um proprietário de um objeto da camada de aplicação habilita seus serviços (i.e. disponibiliza seus serviços à comunidade de usuários).
- **Cenário II:** Um usuário acessa um objeto da camada de aplicação.
- **Cenário III:** Um objeto da camada de aplicação, já publicado e acessível, é movido ou replicado, ficando temporariamente inacessível.

Ainda que elementares estas definições cobrirão a absoluta maioria de operações desempenhadas na Arquitetura de Nomeação, restando outras operações complementares como gerenciamento dos componentes ou combinações lineares dos três

cenários ilustrados. Uma descrição detalhada destas operações juntamente com a função de cada componente ativo no processo será delineada a seguir.

5.5.1 Cenário I: Um proprietário de um objeto da camada de aplicação deseja habilitar seu serviço

Almejando publicar suas aplicações, proprietários devem registrar seus identificadores (i.e. referências aos seus objetos) na infra-estrutura de nomeação. Vislumbrar-se um cenário em que nenhum de seus nomes esteja registrado (para finalidade de completude do caso ilustrado). Ou seja, este é o exemplo em que um proprietário pretende ingressar no mercado com uma nova "marca" desejando disponibilizar serviços em um novo servidor.

O primeiro passo a ser tomado por um proprietário consiste na obtenção de um EID para identificar seu servidor, que poderá, então, ser mapeado para o IP corrente atribuído ao seu nó (aquele que hospeda seus serviços). Em virtude da natureza de gerenciamento distribuída do subsistema de resolução de referências EID RRS implementado em uma DHT, todos os proprietários devem escolher/obter um identificador para seu EID. Então devem associá-lo a um ou mais endereços IP, assinar/certificar o mapeamento estabelecido para o objeto com a Autoridade Certificadora responsável (esta também será utilizada pelo SID) e submetê-lo à DHT EID.

O processo de seleção de um EID é bem definido, consistindo no *hash* criptográfico de um identificador de nó HI (*Host Identifier*), que representa a chave criptográfica pública de um par de chaves pública/privada (gerado aleatoriamente). A natureza de um EID, representado por um identificar opaco (i.e. sem semântica) compreendido por uma longa cadeia de bits (na ordem de centenas) com representação hexadecimal apresenta uma probabilidade de colisão (cenário em que dois proprietários escolhem o mesmo EID) muito baixa (da ordem de 2^{-100}). Uma discussão mais apurada será delineada no próximo capítulo.

Em um cenário hipotético de colisão, a DHT somente reescreverá o EID no caso em que simultaneamente este e a senha do proprietário venham a ser iguais. Então, a dupla <IP, certificado> apontada por tal EID será atualizada, cenário em que o próprio proprietário deseje alterar seus dados, ou um atacante tenha obtido o êxito em furtar a senha e conhecer o *hash* da vítima. Não se pode descartar, em absoluto, uma terceira possibilidade que consiste na chance de isto acontecer por acaso, entretanto esta probabilidade seria representada pela combinação da chance de colisão do certificado gerado juntamente com a

probabilidade da escolha fortuita da mesma senha pelos dois proprietários, um cenário praticamente surreal.

De posse do EID registrado, o proprietário pode proceder agora para a obtenção de um SID registrado. Para tal, este deverá obter um SID (gerado aleatoriamente) e concatenar a este uma lista de elementos requeridos pelo sistema no acesso a objetos da camada de aplicação (e.g. disponibilidade, confiabilidade, latência, portas, rotas, protocolos, EIDs). Em seguida, deverá obter um certificado assinado pela organização certificadora do sistema e também poderá obter um terceiro certificado, designado por uma entidade externa (neste contexto, externa remete-se a uma terceira organização que possui autoridade e confiabilidade consagrada) para garantir a QoSC (Qualidade de Serviço de Conteúdo) dos seus objetos.

De posse destas informações, o proprietário encontra-se apto a registrar as entradas referente ao mapeamento do seu objeto da camada de aplicação no subsistema de resolução de referências RRS SID, que é implementado na forma de uma DHT. As mesmas considerações quanto à probabilidade de colisão de chaves EID podem ser feitas aqui, visto que o tamanho máximo das chaves deve ser o mesmo e a DHT utilizada também.

O processo pode ser finalizado com o registro das informações sobre seu serviço em um diretório da camada de aplicação de duas possíveis maneiras. O proprietário pode solicitar, ativamente, que este catalogo seja feito e publicado através da submissão de termos de pesquisa associados ao seu SID ou, passivamente, durante o processo proativo de catalogo realizado pelas máquinas de busca (i.e. *search engines* tais como Google, Altavista, Yahoo, Cadê [84; 85] e etc) em que termos semânticos serão associados ao seu SID. De maneira geral, este mecanismo não apresenta mudanças fundamentais quando comparado ao mecanismo de registro/publicação de informações realizado pelas máquinas de busca da Internet, sendo a única mudança relevante (e, positiva) o novo apontamento de objetos para SIDs ao invés de para IPs.

5.5.2 Cenário II: Um usuário acessa um objeto da camada de aplicação

Neste cenário um usuário obterá acesso a um objeto da camada de aplicação. Após três resoluções de nomes, localiza-se o endereço IP do servidor que hospeda o serviço procurado e possibilita-se, portanto, que a comunicação se estabeleça através da infra-

estrutura de roteamento da Internet, com mínimas ou nenhuma alteração (que se existirem, estarão localizadas nas aplicações, que tipicamente utilizarão o endereço IP em suas operações finais de conexão).

O primeiro passo, do ponto de vista dos usuários, consiste em obter o SID correspondente ao objeto procurado. Isto pode ser feito de duas formas, primeiro, e menos provável, memorizando-se o SID desejado (a diversidade de métodos passíveis de serem utilizados para publicar um SID não representa o objetivo principal deste trabalho, contudo, não se faz a suposição de que os usuários deverão decorá-lo, assim como os usuários hoje não memorizam freqüentemente os endereços IP), ou através da obtenção precedente de uma busca por termos similares em uma máquina de busca.

Obtido o SID procurado, o usuário pode submetê-lo ao subsistema de resolução de referências SID RRS almejando como resposta uma lista de informações de como acessar o objeto e suas informações de QoSC. Neste contexto, pode-se esperar que informações ilegítimas possam ser retornadas ao usuário fruto de colisões maliciosas ou acidentais. Em face destes problemas, o usuário pode requisitar à CA da arquitetura informações sobre a assinatura destas informações, podendo então, optar pelas tuplas legítimas.

Obtidas as informações legítimas, um dos campos desta tupla será uma lista de EIDs que o usuário submeterá ao subsistema de resolução de referências EID RRS almejando uma lista de endereços IP como resposta. A partir deste ponto, ambas as partes, servidor e cliente, do objeto da camada de aplicação podem iniciar uma conexão utilizando a infra-estrutura regular de roteamento IP da Internet. Obviamente, a resposta a uma consulta EID pode conter diversos endereços IPs ilegítimos e, de forma análoga, o usuário pode recorrer à CA da arquitetura para garantir que as informações são confiáveis (i.e. possuem uma assinatura válida) ou devem ser descartadas (i.e. não possuem assinatura correspondente).

5.5.3 Cenário III: Um objeto da camada de aplicação é movido ou replicado.

Um objeto da camada de aplicação mudará sua localização sempre que mudar seu endereço IP (i.e. seu ponto de acesso à rede registrado dentro do EID) ou qualquer informação de acesso (EID, rota, porta, informações de QoSC) do SID. Em ambos os casos é assumido que o usuário já obteve o SID e o EID correspondente e que a comunicação, ou já esta estabelecida (o que significa que o usuário terá que submeter o EID novamente para obter o novo endereço IP ou o SID para obter as novas informações), ou irá ser estabelecida. Neste

último caso, o usuário deverá obter o EID, SID ou IP correspondente de um *cache* local, o que implicará no mesmo processo de nova submissão de informações precedido de uma falha no acesso a um objeto (com a informação local).

Pode-se inferir, baseando-se em uma variedade de fatos, que os endereços IP apresentarão mudanças mais freqüentes que as referências EIDs, que por sua vez, apresentarão mudanças mais freqüentes que os SIDs. Um argumento concreto nesta direção consiste na própria natureza de cada identificador, haja vista que endereços IP são muito mais efêmeros em virtude de fatores advindos dos protocolos DHCP/NAT, da mobilidade de nós ou de cenários de *multihoming*. Os identificadores EID, por sua vez, só apresentarão mudanças no momento em que um objeto da camada de aplicação for movido de nó para outro, cenário muito menos freqüente do que o primeiro dados os aspectos tecnológicos e práticas arquiteturais na implantação de aplicações, tais como: o uso crescente de máquinas virtuais, a replicação de conteúdo e a adoção de balanceadores de carga. Vale ressaltar aqui que o objetivo do **Princípio 3#** era comportar naturalmente a inserção de *middleboxes* na Arquitetura de Nomes Com Múltiplas Camadas, fato que pode ser vislumbrado agora neste ambiente. Além disso, ao analisar a natureza do SID que possui informações detalhadas sobre o objeto da camada de aplicação referenciado (as quais podem ser alteradas livremente), um cenário ainda mais improvável de sofrer mudanças se estabelece.

Diante das considerações supracitadas, a nova arquitetura apresentará um dinamismo diferente em virtude da ocorrência de mobilidade de objetos da camada de aplicação. A nova prática adotada pelos usuários quando for detectada a interrupção de uma conexão consiste em primeiro contatar o subsistema EID RRS para obter um novo endereço IP, e então, na persistência da ruptura da comunicação, contatar o subsistema SID RRS por novas informações. Se ainda assim, a interrupção na comunicação persistir, o subsistema de Canonização RRS deverá ser explicitamente acessado (i.e. uma intervenção humana ativa será necessária).

Nesta conjuntura, a questão remanescente versa sobre a transparência apresentada pela aplicação que sofreu a ruptura de comunicação. Obviamente, algumas aplicações, tais como as que possuem requisitos de tempo real (i.e. aplicações isócronas, como transmissões de vídeos *online* ou voz sobre IP), apresentarão perdas perceptíveis aos usuários devido à sua natureza inerente que requer a provisão de dados instantânea. Aplicações interativas, bem como as do tipo requisição/resposta (*request/response*) e em lote (*batch applications*) podem lançar mão de diversos artifícios para contornar esta ruptura, ou

seja, para tornarem-se mais transparentes, tais como o armazenamento em memórias ágeis (*bufferization*), a compactação e o uso de códigos de recuperação.

Para ilustrar este cenário, podemos citar a Rede Virtual *Overlay* HIP que estabelece conexões TCP baseadas não mais em endereços IP, mas em identificadores de nós análogos ao EID apresentado. Combinando este recurso com temporizadores adequados em uma aplicação, o HIP criou cenários em que a mobilidade de IPs é tolerada nas comunicações já em trânsito sem causar-lhes uma quebra de semântica. Desta forma, rupturas temporárias seguidas de um novo processo de resolução de referências finito (i.e. inferior aos limites de tempo dos temporizadores), tornam-se transparente aos usuários.

5.6 Requisitos dos Componentes

Em função dos subsídios provenientes da análise dos cenários operacionais tornou-se possível especificar os requisitos necessários para cada componente se tornar operacional.

Endpoints (nó A e nó B)

Os *endpoints* consistem nos elementos que se localizam e autenticam mutuamente através da rede virtual *overlay* que implementa a Arquitetura de Nomeação proposta. Almejando realizar tal tarefa, estes devem ser capazes de executar funções em uma arquitetura com novos elementos na pilha TCP/IP, sistemas de resolução de referências adicionais, bem como, aplicações cliente modificadas.

Espaço de Nomes Canônico (com semântica)

Este espaço de nomes é composto por um conjunto de caracteres/palavras que representam um dado ou aplicação (e até mesmo um usuário) de maneira compreensível/legível (ou seja, com uma semântica associada aos termos). Remove-se dos usuários da Arquitetura de Nomes a árdua, senão impossível, tarefa de memorizar longas cadeias de identificadores hexadecimais.

Subsistema RRS de Canonização

Responsável pelo mapeamento canônico (i.e. regular, certo, pontual, ortogonal) de nomes semânticos para referências SIDs, atende aos seguintes requisitos:

- Uma resolução canônica pode retornar um ou mais SIDs dependendo da consulta de resolução solicitada (e.g. uma consulta executada com termos genéricos retornará, provavelmente, diversos serviços similares). A distinção entre os diversos SIDs pode requerer intervenção manual (e.g. através da seleção do SID correto da lista de possibilidades exibida ou pela execução de uma nova consulta ao subsistema RRS de Canonização, agora, com novos parâmetros de busca);
- Considerando-se a relativa imutabilidade de SIDs, tanto cadastros estáticos (em diretórios), quanto dinâmicos (e.g. uma máquina de busca que indexe os serviços disponibilizados e crie uma base de dados para o mapeamento de SIDs) devem ser empregados.

Espaço de Nomes SID (Camada de Nomeação SID)

O espaço de nomes SID identifica serviços e dados apresentando os seguintes requisitos:

- A escolha de SIDs deve ser arbitrária;
- Considerando-se um dado ou serviço, seu SID não deve ser modificado em virtude de alterações em sua localização geográfica (física ou lógica).
- Cada nó (i.e. servidor de nomes) pode acomodar um número finito relativamente grande de serviços e dados, apresentando como conseqüência um tamanho considerável para representá-los. Cada SID deve ser composto por no mínimo uma cadeia de 256 bits (ou seja, deve ser numericamente maior do que o espaço de nomes EID, haja vista que deve compreender seus valores).

Subsistema RRS de Resolução SID (DHT SID)

O subsistema de resolução RRS é responsável pelo mapeamento de SIDs para a informação requerida por seus usuários: sua localização e parâmetros de serviço. Os seguintes requisitos são necessários:

- Um SID pode ser mapeado a um ou mais EIDs;
- Cada mapeamento EID será associado a:
    - **i.** Informações necessárias ao acesso de serviços e dados naquele nó particular: Aplicações/protocolos (e.g. http, SSH), portas (e.g. TCP ou UDP) e caminho (e.g. caminho de um dado no sistema de arquivos). Estas informações, por sua vez, devem ser disponibilizadas e assinadas pelo provedor de Conteúdo;

ii. Informações de QoSC disponibilizadas pelo nó: latência (de rede) para acessar um objeto e carga (referente ao nó que hospeda o objeto). Igualmente, estas informações devem estar assinadas, só que desta vez por uma terceira entidade (externa à Arquitetura de Nomes) confiável de informações de QoSC;

iii. Tempo de Ciclo de Vida (i.e. TTL ou *Time-to-Live*) que refletirá as informações sobre mobilidade de um nó e auxiliará as resoluções de *cache*;

iv. Uma assinatura válida disponibilizada pelo Provedor de Conteúdo garantindo a veracidade das informações disponibilizadas.

- SIDs podem ser resolvidos localmente (dentro de um nó para teste, por exemplo) ou globalmente (através do uso de uma DHT);
- Os mapeamentos devem ser armazenados (em *cache*) utilizando-se os valores associados de TTL como referência. Uma tentativa de busca sem sucesso a um SID deve invalidar a entrada de *cache* armazenada (o nó pode ter se movido) forçando uma nova resolução;
- Cada mapeamento de serviços e dados deve possuir a assinatura de seu proprietário para evitar a ocorrência de falsas associações;
- Mudanças no mapeamento devem ser realizadas sempre que ocorrer a mobilidade/mudança dos serviços e dados (e.g. entre nós distintos, dentro do mesmo nó, mas com novos valores de caminho ou porta, ou quando ocorrerem mudanças nos parâmetros de QoSC).

Espaço de Nomes EID (Camada de Nomeação EID)

O espaço de nomes EID identifica os nós em uma rede de computadores apresentando os seguintes requisitos:

- Valores de EID não devem ser escolhidos aleatoriamente. Estes devem refletir a credencial criptográfica do nó, objetivando evitar a despersonalização de EIDs (i.e. tentativa *spoofings* forjando a identidade de terceiros).
- Cada EID deve possuir pelo menos 128-bits de comprimento (em se tratando do uso do OpenDHT o tamanho máximo atual, de Agosto de 2007, seria de 160-bits) para viabilizar um espaço de nomes grande o suficiente (i.e. com baixa probabilidade de colisão), bem como para facilitar a implementação (o uso do mesmo tamanho de identificador do endereço IPv6 pode facilitar uma possível migração do TCP para ser associado ao EID, ao invés do IPv6).

Subsistema de Resolução EID RRS (DHT EID)

O subsistema de resolução EID é responsável pela localização dos nós (i.e. este realiza o mapeamento entre o EID ao seu ponto de acesso à rede corrente, ou no caso da Arquitetura TCP/IP, o seu endereço IP). Os seguintes requisitos são necessários:

- Um EID pode ser mapeado para um conjunto de endereços IP de modo a comportar situações de *multihoming*;
- Cada mapeamento deve possuir um TTL associado que refletirá a mobilidade dos nós e auxiliará a resolução de nomes através de *caches*;
- Um EID pode ser mapeado para um endereço IPv4 ou IPv6;
- EIDs podem ser resolvidos localmente (por exemplo, dentro de um nó para realizar testes) ou globalmente (como é o caso de utilizar uma DHT no processo);
- Mapeamentos devem ser armazenados em memória (i.e. em *cache*), utilizando um TTL associado como guia. Requisições mal sucedidas de comunicação devem invalidar a respectiva entrada em *cache* (e.g. o nó pode ter sofrido uma mobilidade) acarretando em uma nova resolução;
- Cada mapeamento deve ser assinado com a assinatura do seu proprietário de forma a evitar falsas associações;
- Mudanças nas associações devem ser feitas sempre que houver mobilidade em serviços e dados ou mudanças nos parâmetros de QoSC.

Subsistema CA

O subsistema CA é responsável por auxiliar nós e aplicações em sua garantia da legitimidade e integridade dos respectivos EIDs e SIDs. O subsistema CA apresenta os seguintes requisitos:

- Cada nó deve possuir uma lista de autoridades CA raiz e seus correspondentes certificados (i.e. chaves públicas);
- Cada nó deve ser capaz de consultar a CA correspondente de forma a determinar a validade dos certificados, ou seja, obter informações como o certificado do proprietário de algum objeto (i.e. serviços, dados ou usuários) ou informações do certificado de QoSC verificando se estes não foram revogados.

Entidade Provedora de QoSC

Um provedor de QoSC, que compreende uma entidade terceira ao sistema, é responsável pela asserção sobre a validade da resolução das informações do SID. Os seguintes requisitos devem ser atendidos:

- A entidade auditora de QoSC deve ter um certificado assinado por uma CA pertencente à lista raiz de CAs, de forma a permitir que os nós possam confiar nas informações publicadas;
- O provedor de QoSC deve publicar a quais políticas está submetido quando da asserção de informações para os provedores de conteúdo;
- Um nó deve ser capaz de requisitar a uma CA informações sobre a validade do certificado da entidade provedora de QoSC de forma a verificar se seu certificado não foi revogado;
- Nós devem ser capazes de definir preferências aos provedores de auditoria de QoSC de forma a permitir uma decisão automática de serviços e dados mais adequados aos seus interesses sempre que houver diferentes provedores de QoSC disponibilizando informações competitivas ou conflitantes;
- Um nó pode, optativamente, definir uma lista de Provedores de QoSC não confiáveis.

Subsistema de Fluxo Criptografado

O subsistema de fluxo criptografado é responsável pela segurança no fluxo de informações das comunicações atendendo aos seguintes requisitos:

- Os nós devem ser capazes de se autenticar mutuamente através do uso de suas chaves pública/privada. O EID, que consiste no *hash* criptográfico da chave pública do nó, deve ser usado e ser suficiente para permitir que ataques do tipo *man-in-the-middle* [86] sejam bem sucedidos;
- Após a autenticação, um protocolo de rede de comunicação segura, como por exemplo, o IPSec deve ser utilizado, sem modificações, tanto para a negociação de chaves de sessão simétricas quanto para o tráfego criptografado.

5.7 Relacionamento dos Componentes Arquiteturais

A concisa definição dos requisitos demandados por cada componente fornece uma visão mais sólida do papel desempenhado por estes, entretanto seu relacionamento

permanece ainda fluído. Uma ilustração sucinta da relação entre eles é ilustrada na Figura 27, em que uma consulta realizada por um usuário em um nó A é remetida a um nó B utilizando os diversos componentes acima descritos.

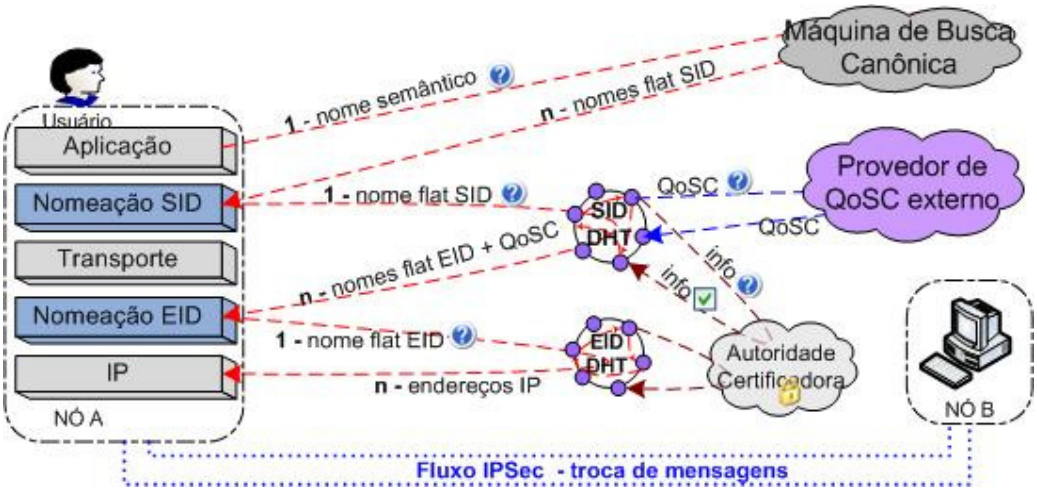

Figura 27. A Estrutura de Componentes e Relacionamentos da Arquitetura de Nomes

Observando o hipotético cenário da Figura 27 que nó A deseja obter um objeto remoto localizado em um nó B utilizando os componentes da nova Arquitetura de Nomes sem hierarquia, o primeiro passo consiste na identificação do SID associado ao nome semântico deste objeto. Para tal, será necessária uma consulta ao RRS de Canonização e este deve retornar um conjunto de possíveis identificadores realizando o mapeamento:

- **1 nome semântico -> n referências flat SID**

Obtido este conjunto de possíveis SIDs, o nó cliente (com ou sem a interferência do usuário) deve selecionar o SID candidato (cenário análogo ao que ocorre atualmente, por exemplo, quando um usuário realiza uma consulta a uma máquina de busca e necessita selecionar uma entre várias possíveis respostas) e então consultar a DHT SID para obter o(s) referido(s) EID. Um conjunto de EIDs pode estar associado a um único SID, conforme previsto na proposta arquitetural. Um hipotético cenário em que isso é desejável consiste em mecanismos de balanceamento de carga ou replicação de conteúdo. E, portanto, diferentes QoSC estarão associados a cada resposta obtida de acordo com as características disponibilizadas pela entidade provedora de informações QoSC. A cardinalidade deste mapeamento bem como características hipotéticas de serem fornecidas pelas respostas é a seguinte:

- **1 referência SID -> n referências flat EID (identificador, caminho, porta, latência, carregamento, ..., etc)**

Cada identificador de nós EID pode, e provavelmente deve, apresentar características diversas de QoSC e a escolha de qual candidato é mais adequado representa um critério definido pelo cliente. Esta escolha pode consistir em um mecanismo automatizado que priorize certa característica em detrimento de outra, como por exemplo, sempre escolher o servidor com menor carregamento ou maior largura de banda, ou ser selecionada manualmente.

A solicitação de características de QoSC não consiste em um requisito mandatório do sistema e sua ausência não deve inviabilizar o funcionamento do mesmo, ficando o cliente responsável pela decisão de prosseguir ou não em face a indisposição das mesmas. A disposição/atuação do Provedor de QoSC pode ser tão dispersa quanto o modelo de economia que norteará esta arquitetura e sua escolha representa uma questão aberta e fluída, assim como o era o futuro das Máquinas de Busca nos seus primórdios (com inserções manuais de novas entradas e sem uma política econômica bem definida).

Atualmente, diversos são os incentivos para sua subsistência econômica daquelas sendo sua fonte principal de capital a publicação de anúncios privados criteriosamente selecionados de acordo com as buscas realizadas por seus clientes e altamente automatizada é a inserção de novas entradas, como pode ser observada através do algoritmo de busca e classificação adotado pelo Google, o *Map Reduce* [87].

Desta forma, pode-se vislumbrar alguns cenários para a disposição e o acionamento do Provedor de QoSC, como será ilustrado abaixo. Ressalvas devem ser feitas aqui, entretanto, quanto à necessidade de o mesmo ser uma entidade confiável e terceira ao sistema, ou seja, há um conflito claro de interesses entre proprietários de conteúdos e Provedores de QoSC, pois estes serão responsáveis por classificar aqueles e, portanto, não é desejável que representem simultaneamente ambos os papéis.

- O Provedor de QoSC pode ser acionado pela Máquina de Busca Canônica e esta disponibilizará (automaticamente ou manualmente) uma conjunto de informações associadas às referências consultadas. É possível a existência de cenários em que os clientes optem por um "modo avançado", em que estes desejem escolher minuciosamente quais critérios de QoSC representam sua preferência. Ou, por um "modo básico" em que a Máquina de Busca define que a melhor referência para seus

clientes neste modo será norteada por um critério (e.g. latência) ou por um conjunto deles (e.g. latência, largura de banda e carregamento do servidor).

- O Provedor de QoSC pode ser acionado simultaneamente pelo cliente no momento da consulta à DHT SID quando da procura pelo conjunto de referências EID. E neste caso, novamente, é possível imaginar a existência de escolhas manuais ou automatizadas na definição final de qual EID acessar.
- Finalmente, é possível imaginar um terceiro cenário, que tem grande potencial de ser o modelo inicialmente adotado, que é aquele em que o auditor de QoSC representa uma entidade relativamente desvinculada das consultas feita pela Máquina de Busca ou ao RRS SID. Neste caso, quando os clientes desejarem verificar a existência de informações de QoSC de um certo conteúdo, estes terão, por exemplo que navegar pelo site de QoSC e verificar quais os indicadores estão disponíveis para um certo proprietário e então definir sua classificação pessoal sobre este proprietário.

Por fim, selecionado qual o EID candidato, deve-se remeter ao subsistema de resolução RRS EID, ou seja, à DHT EID para obter o conjunto de possíveis endereços associados ao servidor procurado (e.g. cenário de servidor em *multihoming*). O mapeamento esperado é o seguinte:

- **1 referência EID -> n endereços IP**

Neste momento, ambas as partes podem ser sensibilizadas (i.e. um pedido de conexão pode acontecer), autenticadas e um fluxo seguro (e.g. IPSEC) pode ser iniciado.

Considerações sobre a mobilidade de objetos podem ser feitas neste momento. Conforme discutido anteriormente, a conexão nesta nova proposta arquitetural não será mais baseada no par <IP, Porta>, mas sim no par <SID, Porta> e mobilidades de EIDs e IPs são mutuamente suportadas. A ocorrência de mudanças nos endereços IP, caso mais freqüente, requer a atualização na associação <EID, IP> e correrá, por exemplo, quando houver a mobilidade de um nó entre domínios distintos. Mudanças no identificador EID, menos freqüentes, também são suportadas e estas requerem atualizações na associação <SID, EID> para que a semântica da conexão seja mantida.

No cenário hipotético em que uma aplicação seja migrada de um servidor mais sobrecarregado para um terceiro servidor menos congestionado em um processo naturalmente suportado de balanceamento de carga por esta nova Arquitetura, seu EID seria modificado ainda que a aplicação mantenha seu SID intacto. Obviamente, mudanças no identificador de

um objeto (i.e. um dado, serviço ou usuário) são possíveis e modificam seu SID e, neste caso, a semântica da conexão não será preservada.

5.8 Considerações sobre a Implementação

Esta subseção não tem por objetivo tecer considerações sobre a escolha específica de certo paradigma de implementação ou linguagem específica, mas sim ajudar a definir um esquema/*framework* guia de implementação da Arquitetura de Nomes aqui proposta. Três principais aspectos devem ser considerados na discussão sobre a implementação deste sistema.

Primeiramente, uma clara separação deve ser feita aqui, quanto à implementação de uma prova de conceito como estudo de viabilidade da coexistência de dois espaços de nomes (hierárquicos e opacos) como a que será apresentada no próximo capítulo e uma implementação completa de um produto final almejando a adoção em larga escala na Internet.

Um produto final desta arquitetura deverá contar não somente com a maturidade de implementação do mesmo, como foi o caso de décadas de desenvolvimento da arquitetura TCP/IP em detrimento da arquitetura OSI, mas também da adoção em grande dimensão por parte da comunidade de pesquisadores e indústrias.

Posta esta separação, o segundo e próximo aspecto a se considerar consiste na forma geral de implementação. Duas principais abordagens devem ser consideradas: a implementação da Arquitetura de Nomes diretamente nos nós dos clientes, inserindo porções de códigos em seu sistema operacional e/ou até mesmo em suas aplicações ou a utilização de um *proxy* que será interposto entre os nós finais, i.e. *endpoints*, e o RRS (situado nos *endpoints*, é claro), a ser usado na seleção das consultas de nomes para o sistema de nomes atual RRS DNS ou para o novo sistemas de nomes RRS (neste caso o SID).

Finalmente, há a necessidade de se verificar o quão custosa será a adoção dos requisitos supracitados nos *endpoints*, sejam estes componentes novos (e.g. um novo espaço de nomes) ou apenas modificações incrementais nas camadas/protocolos já existentes. Em concordância com a utilização de redes virtuais *overlay* (como consiste o caso aqui proposto), este parece ser o método cuja paulatina adoção apresenta melhor aceitação em virtude de sua própria natureza incremental, com mínimas mudanças nos componentes da rede substrato e com maior probabilidade de ser bem sucedida. Conforme discutido no capítulo de Redes

Virtuais, este método quando comparado à proposição de soluções completamente inovadoras (i.e. que venham a substituir completamente o modelo atual) mostrou-se mais viável.

Quanto ao primeiro aspecto, a decisão de uma implementação inicial de uma prova de conceito pode ser interpretada como uma verificação natural das primeiras etapas do ciclo de vida desta arquitetura. Neste sentido, considerar que esta implementação seja adotada pela comunidade, imatura como se encontra nas primeiras fases de qualquer projeto, compreende uma consideração não realística. Porém, este protótipo permite, entre várias outras possibilidades, o estudo de sua usabilidade e aceitação na comunidade científica como uma nova proposta de arquitetura de nomes para a Internet. Espera-se que com o aumento de sua complexidade, sucessivas revisões e avanço de sua maturidade, que esta venha a se tornar uma implementação estável digna de substituir com excelência o modelo atual TCP/IP proporcionando todos os benefícios outrora mencionados e permitindo a análise de seu desempenho em face ao modelo corrente.

Em se tratando do segundo aspecto combinado ao primeiro, a convergência para a adoção de uma solução baseada em *proxy,* haja vista que esta permitiria sua adoção com mínimas mudanças nos *endpoints*, bem como ínfimas, senão nulas, mudanças nas aplicações cliente consiste em uma disposição mais adequada. Um *proxy* pode ser adaptado para selecionar não somente as requisições dirigidas para o RRS corrente (i.e. o DNS) como para o novo RRS, bem como para qual subsistema RRS (EID ou SID) uma requisição deve ser encaminhada, permitindo a mobilidade e reutilização de códigos. Além disso, a utilização de um *proxy* consiste em um estratégia interessante do ponto de vista da flexibilidade oferecida por uma interface capaz de se comunicar com outras implementações de sistemas RRS (i.e. diferentes tipos de DHT), permitindo a avaliação de uma vasta gama de algoritmos de roteamento de DHTs e estratégias de *cache*.

Finalmente, acredita-se que nas primeiras fases de implementação da Arquitetura de Nomes, a adoção de uma DHT com alcance global seja de fundamental importância para garantir uma acurada análise de viabilidade da mesma em face de sua capacidade de crescimento e uso em grande escala. Desta forma, a DHT candidata consiste no projeto OpenDHT [30] que apresenta uma interface aberta (i.e. pública e de fácil acesso) altamente flexível, com tamanho de chave variável cujos mecanismos de manipulação (i.e. interfaces de inserção, remoção e consulta) são altamente otimizados e atendem aos requisitos necessários de abrangência global, dada sua larga disponibilidade em nós do PlanetLab espalhados por diversas instituições acadêmicas e comerciais.

5.9 Prova de Conceito

Esta prova de conceito compreende um sistema de resolução de referências SID distribuído para objetos (i.e. dados, serviços e usuários) e *metadados* semânticos baseados em *proxies.* Vale ressaltar que, como prova de conceito, diversas simplificações foram realizadas principalmente em virtude da adoção de certas tecnologias mais flexíveis em detrimento a outras mais eficientes. Como exemplo, podemos citar a adoção da OpenDHT cuja disponibilidade é ainda um grande gargalo visto o seu estado de maturidade comprometendo diversos testes devido a períodos de indisponibilidade ou intermitência do serviço e a altas taxas de latência nas resoluções quando comparado ao DNS. Sua adoção foi realizada, contudo, pois esta representava a candidata cuja dispersão geográfica era mais fiel à realidade da Internet, por conseqüência, se aproximando do modelo final almejado.

O objetivo principal do desenvolvimento de uma API simplificada consistiu em realizar uma implementação em modo de usuário de resolução de SIDs, haja vista que modelos similares para resolução de EIDs já se encontram disponíveis na literatura, como por exemplo, através do protocolo i3, HIP, Hi3, TRIAD, IPNL e etc.. Uma análise detalhada de protocolos para resolução EID seguida de testes de bancadas foi realizada em paralelo a esta implementação e seus resultados podem ser encontrados em [48]. Ainda que imaturo como esteja este sistema e baseado nos ambiciosos objetivos da Arquitetura de Nomes para a Internet, a experiência de sua implementação auxiliou na previsão dos diversos obstáculos a serem enfrentados quando da adoção *de facto* desta Arquitetura com todos os espaços de nomes integrados (semânticos, SID e EID).

5.9.1 Estrutura da API

Uma descrição concisa da API será realizada e o completo entendimento do sistema deve ser obtido com a leitura integral de todas as seções e através do seguinte trabalho [49]. A descrição aqui realizada engloba unicamente os componentes relacionados à estrutura do Sistema implementado não sendo descritos os demais componentes auxiliares de teste, criptografia ou comunicação XML/RPC que podem ser encontrados através da inspeção detalhada do código. A estrutura de pacotes utilizada segue indicada.

DHT

Este pacote contém as classes de construção e manipulação do espaço de nomes SID bem como do sistema de busca RRS SID construído para este sistema. É sua função preparar o ambiente, popular, remover e consultar a DHT SID. Uma ilustração deste pacote pode ser observada através do diagrama de classes da Figura 28.

- ✓ Principais classes: Environment.java; DHTManager.java; Gateway.java; ClientPut.java; ClientGet.java e ClientRm.java.

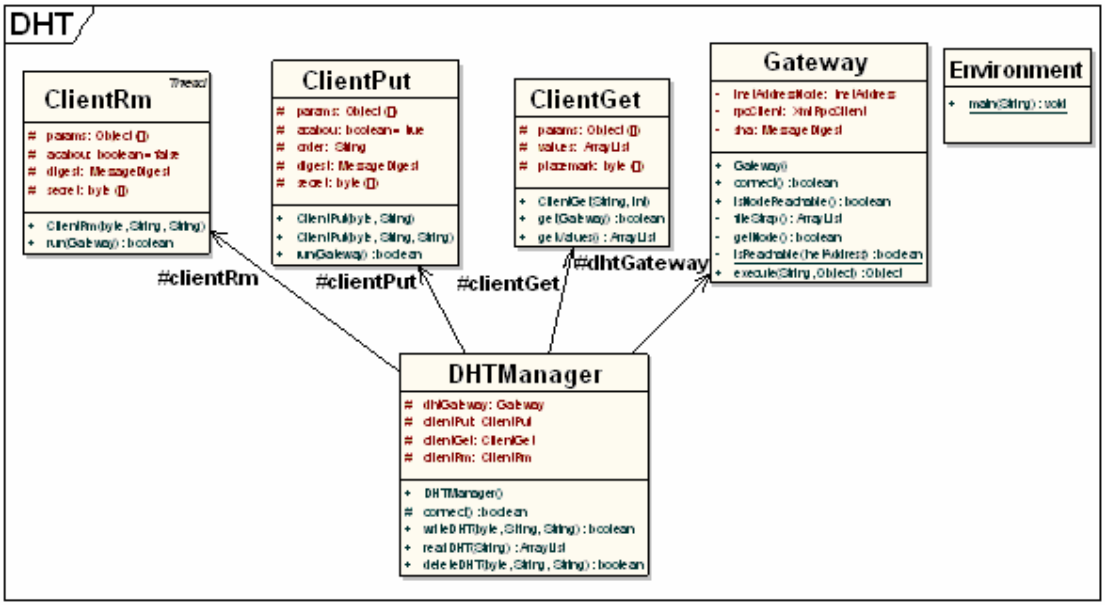

Figura 28. O Pacote DHT

Entities

Responsável pela criação dos objetos e metadados associados. É sua função criar o objeto e gerar o identificador hexadecimal referente ao objeto. Vale ressaltar que, como prova de conceito, diversas simplificações foram realizadas, visto que na proposta arquitetural, por exemplo, os metadados seriam responsabilidade de uma terceira identidade e aqui estão sendo populados por um subsistema interno à Arquitetura. Observe o diagrama de classes da Figura 29.

- ✓ Principais classes: FlatObj.java e HashSID.java.

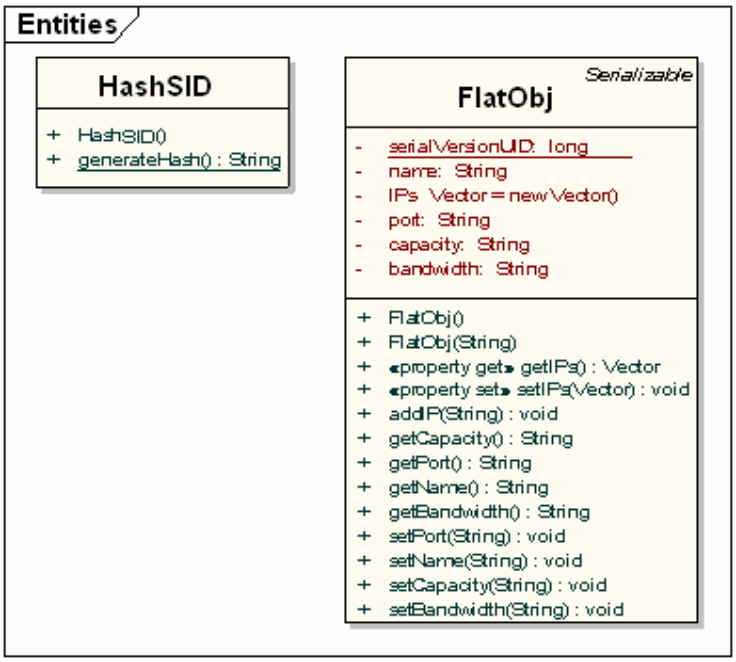

Figura 29. O Pacote Entities

Proxy

Pacote que permitiu a integração dos espaços de nomes hierárquico da Arquitetura TCP/IP e sem hierarquia do Arquitetura aqui proposta adaptando o Proxy Scone para realizar consultas tanto à DHT SID quando ao DNS. Seu diagrama de classes pode ser observado na Figura 30.

- ✓ Principais classes: HttpURLEditor.java e ProxyPlugin.java.

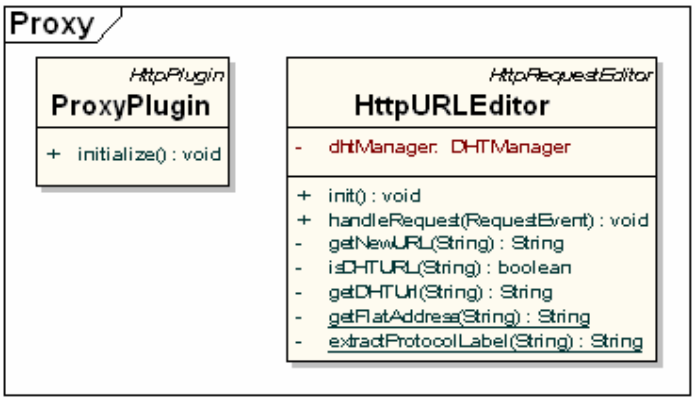

Figura 30. O Pacote Proxy

Utils

Responsável pela serialização dos objetos (transformando-os em bytes) e posterior recuperação quando da realização de inserções e consultas à DHT SID. Veja seu diagrama de classes na Figura 31.

- ✓ Principal classe: Serializer.java.

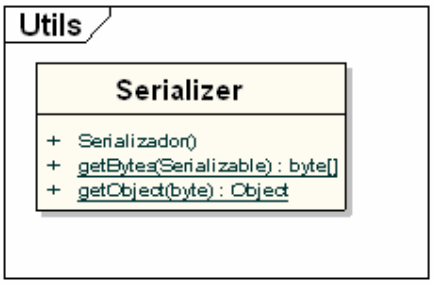

Figura 31. O Pacote Utils

Libs

Este pacote contém as bibliotecas adicionais (i.e. outras que não aquelas disponíveis no Java *Development Kit* 1.5) necessárias para o funcionamento do Sistema.

- ✓ Principais arquivos: wbij45.jar; ws-commons-util-1.0.1.jar; xmlrpc-client-3.0.jar e xmlrpc-common-3.0.jar.

5.9.2 Considerações Arquiteturais

Para a construção deste Sistema RRS SID diversas considerações foram realizadas devido à infra-estrutura adotada. Informações detalhadas a respeito de cada componente/classe podem ser obtidas a partir da documentação do código (javadoc). Os objetos SID, aqui representados pela classe FlatObj.java consistem no principal componente deste ambiente e um novo espaço de nomes foi criado para acomodá-los e a seus metadados. A cada SID deve poder ser associado um conjunto de EIDs e a cada EID um conjunto de IPs.

Como a DHT candidata para acomodar este novo espaço de nomes foi a OpenDHT, que é baseada na implementação Java da DHT Bamboo [39] acessível tanto via Sun RPC e XML RPC, procedimentos de chamada remota foram utilizados na inserção, remoção e recuperação dos mesmos. Quanto à semântica utilizada no espaço de nomes SID, a mesma foi definida na prática pelas restrições de armazenamento da OpenDHT, conforme indicado a seguir:

Semântica do OpenDHT (inserções)

- *key:* byte *array*,máximo de 20 bytes (160 bits)
- *application: string*
- *client_library: string*
- *value:* byte *array*, máximo de 1024 bytes

- *ttl_sec: four-byte integer*, máximo de 604,800 *seconds* (uma semana)
- *secret_hash:* SHA-1 *hash* do segredo que é utilizado nas remoções seguras.

Os campos *application* e *client_library* são utilizados apenas para propósitos de construção de históricos (i.e. *log*) e podem ser preenchidos, respectivamente com o nome da aplicação e com o nome da biblioteca utilizada para acessar o OpenDHT. Os campos *key* e *value* correspondem à chave e seu valor associado, aqui denominado identificador SID e metadado. Finalmente, o *ttl_sec* compreende o tempo desejado de armazenamento de um objeto e *secret_hash* o campo que se não nulo, possibilitará inserções e remoções seguras. O Sistema aqui desenvolvido não obriga que este campo seja preenchido necessariamente, contudo, este define o padrão de comportamento ativo ou passivo da Tabela *Hash* quando da ocorrência de colisões, acidentais ou não, permitindo sobrescrever os dados de um cliente, ou impedindo diante do desconhecimento da senha associada.

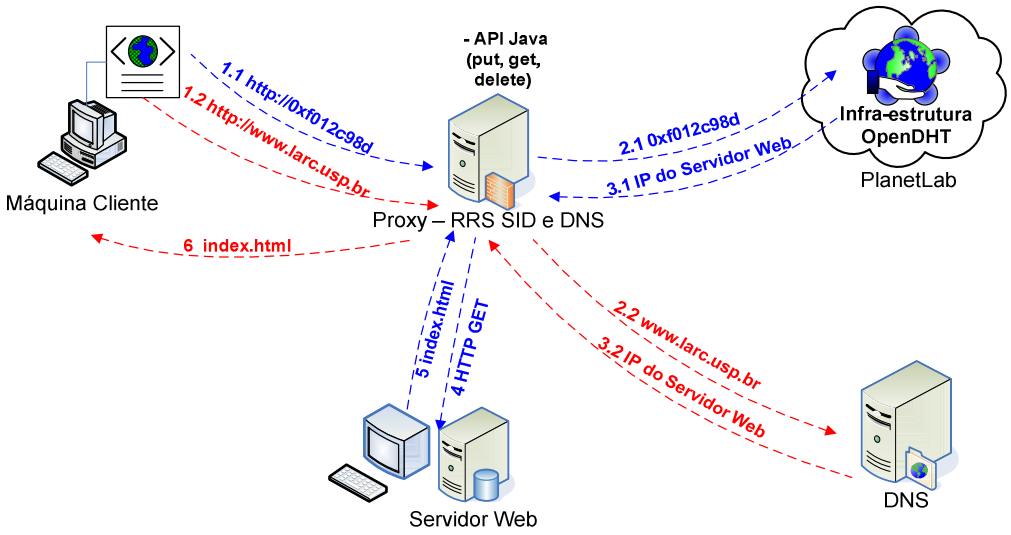

Figura 32. Resolução de Nomes em um Ambiente Utilizando a Rede Virtual de Nomes *Overlay*

Quando uma consulta HTTP é realizada dentro da nova Arquitetura, seja ela uma busca por um nome DNS (ver os itens 1.2, 2.2, 3.2, 4, 5 e 6 da Figura 32) ou por um SID (ver os itens 1.1, 2.1, 3.1, 4, 5 e 6 da Figura 32), a mesma é inspecionada pelo Proxy e este deve tomar as devidas providências para sua resolução. No cenário convencional, quando uma máquina cliente realiza uma busca de nome hierárquico, primeiro deve realizar a resolução de

seu nome, através da utilização do DNS, para o IP do servidor correspondente, em seguida obter o referido dado através de uma solicitação ao servidor *Web*.

Em um ambiente como o proposto na Arquitetura original, considerando a existência de nomes flat e de um *Proxy* intermediário, uma consulta HTTP em modo usuário a um nome sem hierarquia segue da seguinte forma. Primeiro, haverá a busca por um SID conhecido e seus metadados associados. Caso o cliente desconheça o SID correspondente, este deve se remeter a uma Máquina de Busca Canônica, elemento não compreendido por esta Prova de Conceito e que deve compreender uma entidade terceira à Arquitetura. Esta busca deve então ser intermediada pelo *Proxy*, elemento que possui o conhecimento do Sistema de Rede Virtual hibrida de nomes e este deve se comportar como um RRS SID e realizar a resolução do referido SID para um conjunto EID através da busca à Infra-estrutura pública unidimensional do OpenDHT. No presente caso, por se tratar de uma Prova de Conceito em modo usuário de referências SID, espera-se que a nova resolução ao segundo espaço de nomes seja realizada pelo RRS EID, como já é realizado hoje, com o uso do protocolo HIP ou i3.

Em seguida, localizado seu endereço correspondente, o processo pode prosseguir normalmente como se originado através da Arquitetura TCP/IP, seguindo os passos 4, 5 e 6 da Figura 32. Vale ressaltar que a única modificação inserida na máquina cliente consiste na configuração de seu navegador para suportar o uso do Proxy ficando o cliente livre do tratamento de qualquer detalhe de implementação da nova Arquitetura de Nomes. Esta prova de conceito, simplificada como é, permitiu, porém, a coexistência dos dois espaços de nomes, e garantiu a abstração por parte do cliente de detalhes do Sistema característica desejada e obtida através da utilização de Redes Virtuais *Overlay*. Diversas modificações, porém precisaram ser realizadas no *Proxy*, para suportar este novo espaço de nomes e utilizar a infra-estrutura distribuída do PlanetLab.

CAPÍTULO 6 RESULTADOS E DISCUSSÕES

Diante da estruturação deste trabalho em uma jornada que combinou uma miríade de testes, estudos e conseqüentes resultados parciais, apresentados nos 4 primeiros capítulos, na composição dos fundamentos da Arquitetura de Nomeação Com Múltiplas Camadas para a Internet proposta no capítulo 5, e subseqüentes resultados finais, este capítulo apresenta uma compilação dos resultados mais relevantes. Subseqüentes discussões são realizadas na tentativa de elucidar quais os principais desafios a serem enfrentados quando da sua adoção, sem consistir, contudo, em uma análise exaustiva de questionamentos ou soluções.

A decisão da utilização de uma Rede Virtual *Overlay Non-Routing* como uma alternativa para a implementação de novas funcionalidades para a Internet, consistiu em uma medida baseada em diversos argumentos. Em se tratando da adoção de Redes Virtuais, esta decisão adveio do uso crescente de técnicas de virtualização/abstração na inserção de novas funcionalidades para a Internet, conforme demonstraram os projetos analisados no capítulo 4. Em oposição a esta alternativa, que pode ser adotada de forma incremental e cujo crescimento vertiginoso de propostas naturalmente aumenta o seu grau de maturidade impulsionando-a para uma padronização e acelerada criação de novas versões de implementação mais estáveis e confiáveis, têm-se a adoção de propostas que venham a substituir o modelo atual da Internet. Conforme discutido anteriormente, ainda que esta última abordagem apresente resultados estruturais mais sólidos, sua implantação mostrou-se inviável na prática devido à ausência de motivação/colaboração de fabricantes e usuários para tal. Pode-se destacar como ilustração, a resistência sofrida até então de propostas de mudanças estruturais, tais como a do IPv6, cuja adoção em larga escala ainda se mostra diminuta.

Ao expandir as possibilidades do exemplo anterior de utilização de redes virtuais para um cenário ainda mais complexo, que implemente além de funções de roteamento, a nomeação, a mobilidade e a segurança, é possível viabilizar um cenário adequado para aplicações de Internet *Banking*, Vídeo sob Demanda, Escritório Remoto ou Replicação de Dados a serem implementadas como uma camada de aplicação otimizada. E, possivelmente, menos propensa a apresentar problemas em virtude da simplicidade de propósito com um ciclo de vida de implementação reduzido e, consequentemente, com uma competitividade aumentada. Em essência, acredita-se que um movimento similar ao que

ocorre com a utilização de técnicas de virtualização [88; 89] de processamento e redes locais [90; 91] deva ser aplicado ao cenário da Internet resultando em ganhos similares.

A abordagem de uma proposta de Nomeação *Non-Routing* e *Overlay* foi fundamentada basicamente nas discussões quanto à adoção de *Overlays* em detrimento a *Underlays,* e *Non-Routing* em detrimento a propostas que afetassem a infra-estrutura de roteamento TCP/IP realizadas no capítulo três, em: 3.1.2 O Conceito de *Overlays* e *Underlays*; 3.1.2.1 Redes Virtuais *Overlay*; 3.1.2.2 Tipos Específicos de *Overlays* e *Underlays* e 3.1.3 O Conceito de Transporte *Routing* e *Non-Routing*. Entretanto, estas discussões emergiram de análises de testes de bancada de rede realizadas com os protocolos IPv4, IPv6, i3 (particularmente, o cliente OCALA [92] em modo de implementação usuário), HIP (nas versões OpenHIP [93] modo kernel e InfraHIP [84]), Hi3 e IPSec com (AES 128, AES 192 e 3DES) e sem criptografia (*null*).

A metodologia adotada nesta investigação consistiu na análise espectral de diferentes protocolos nativos e *overlays,* que foram inseridos na pilha TCP/IP, diante de uma ampla variação de parâmetros (e.g. tamanho da janela, tamanho do *socket*, tamanho da mensagem, *jumbo frames* e etc) e cargas em redes Gigabit (e.g. *stream* e transacional). Os casos de testes tiveram seu perfil delineado pelas aplicações de *benchmark* netperf [94] (e.g. netperf/TCP/IP e netperf/TCP/HIP/IP) e iperf [95], no que tange à geração de tráfego de rede e variação de parâmetros. As análises das perturbações geradas, do ponto de vista computacional (e.g. consumo de CPU, memória, *context switches* e etc.), foram coletadas com ferramentas de monitoramento como o sar/sa1 [85] e do ponto de vista de rede, com o ethereal [96]. Devido à complexidade e extensão deste trabalho, maiores detalhes podem ser encontrados na literatura, em [48; 97], contudo as principais conclusões serão apresentadas aqui pois afetaram sobremaneira diversas decisões arquiteturais desta proposta.

Com o objetivo de verificar a viabilidade da utilização de *overlays* como infra-estrutura de transporte de dados e metadados em face de diferentes tipos de cargas, especialmente em canais de alta velocidade (e.g. *GigaBit Ethernet*), pôde-se verificar que o uso de *overlays*, sejam eles do tipo Routing e Non-Routing ainda apresentam impactos de desempenho consideráveis quando comparados ao uso dos protocolos nativos Ipv4 e Ipv6. Entretanto, os benefícios proporcionados por estes protocolos (e.g. HIP, i3, HI3), que inserem camadas adicionais de resolução na pilha TCP/IP aumentando a sobregarca do sistema, porém, habilitando novas funcionalidades, tais como a confidencialidade, autenticação e segurança de dados, consiste em motivação suficiente, tanto para fins comerciais quanto acadêmicos, na sua adoção na Internet.

Como esta proposta objetiva a utilização de fluxos seguros com o protocolo IPSec, é importante observar que as análises mostraram impactos de performance em seu uso, principalmente com criptografia, em comparação aos protocolos IPv4 e IPv6, tanto do ponto de vista do consumo de CPU (veja Figura 33) quanto de memória (ver [48; 97]). Este comportamento foi seguido por uma utilização inferior da largura de banda (ver Figura 34), contudo, acredita-se que os benefícios de segurança oferecidos pelo uso deste protocolo, aliado aos artifícios já disponíveis em mercado de cartões de rede com aceleração criptográfica em hardware, já justifiquem sua adoção.

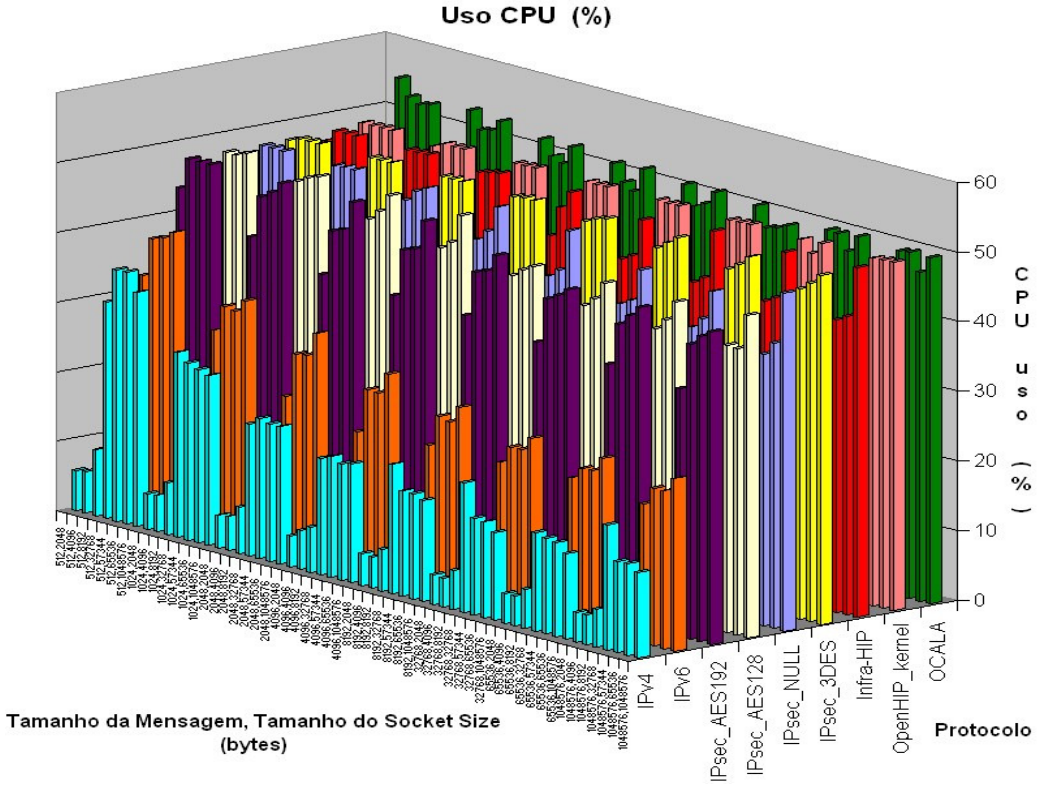

Figura 33. Consumo de CPU de Redes Virtuais Overlays em Canais Gigabit

Vale ressaltar, através da inspeção da Figura 33, que as variações no consumo de CPU apresentaram comportamento similar para todos os protocolos, com o consumo máximo de CPU ocorrendo com os tamanhos mínimos de *socket* e tamanhos máximos de mensagem para todos os protocolos (comportamento esperado devido à excessiva fragmentação das mensagens de tamanhos superiores em reduzidos *sockets*, conhecido como efeito de *packetization*[48; 97]*).* Em oposição, e já esperado, um comportamento oposto, ao efeito da fragmentação, foi acompanhado na vazão do sistema (ver Figura 34), que apresentou os melhores resultados para os maiores tamanhos de *socket* e mensagem (este resultado foi

ainda mais interessante quando do uso de *jumbo frames*, conforme pode ser visto em [48; 97]).

Através da observação dos protocolos IPv4 e i3 (com mínimo consumo de CPU e máxima vazão, em um extremo e vice-versa, no outro extremo) questionamentos quanto a inserção de Redes Virtuais *Overlay Routing*, que realizam um mecanismo de roteamento alternativo ao TCP/IP, como é o caso do cliente OCALA-i3 surgiram. Além dos impactos analisados tanto no consumo de CPU, memória e inferior largura de banda, ao estabelecer cenários com mais de meia dúzia de clientes (sendo pelo menos dois destes, servidores i3 *rendezvous* [8]), o sistema Ocala-i3 versão 2.1 não foi capaz de expandir-se de forma aceitável, tornando-se indisponível.

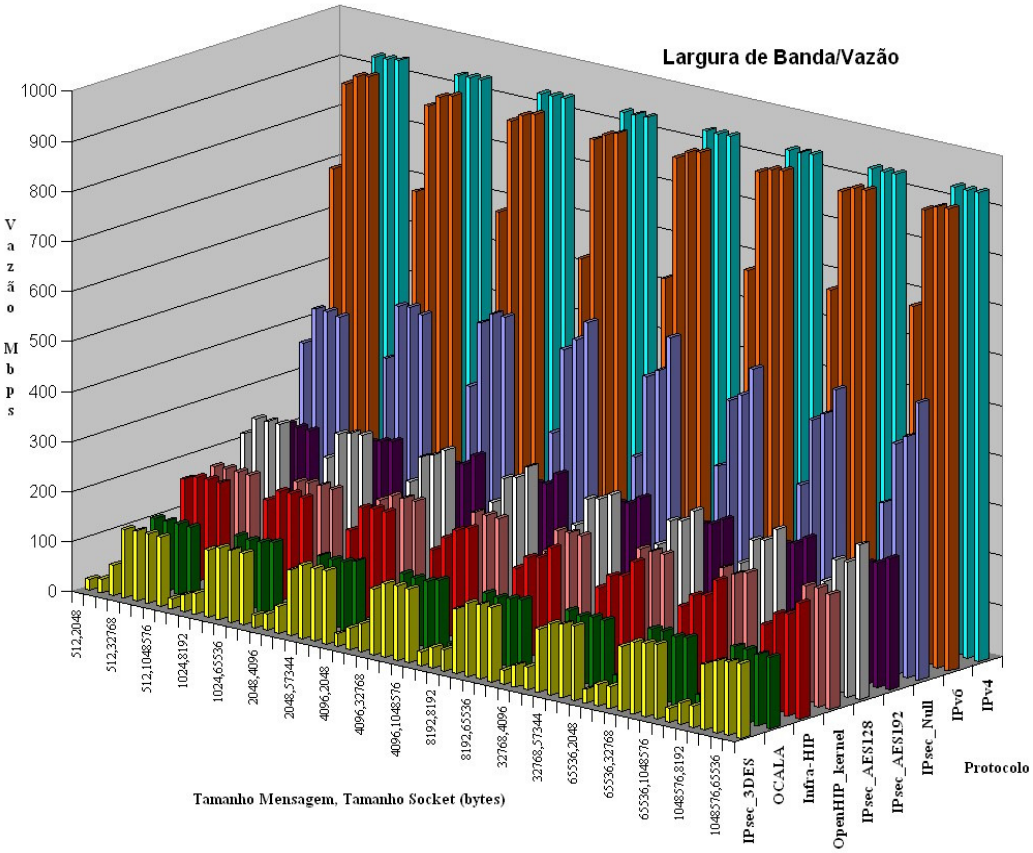

Figura 34. Largura de Banda Verificada em Canais Gigabit

Desta forma, mais um caso para a adoção de Redes Virtuais *Overlay Non-Routing* foi justificado (além dos argumentos estruturais já apresentados no capítulo três sobre a não adoção de *Routing Overlays*), no atual estado de maturidade das propostas deste tipo de *Overlays* de roteamento. Grandes desafios quanto a adoção de Redes Virtuais de roteamento

são esperados, haja vista o empenho de décadas do desenvolvimento dos protocolos IPv4 e IPv6.

Além disso, outro fato a se observar, através da monitoração de atividades do *kernel*, foi a verificação de um número excessivo de *context switches* (i.e. chaveamentos de modo *kernel* e usuário) realizados pelo i3 em comparação ao IPv4 (ver [48; 97]), o que pode ser entendido observando-se o número de camadas inseridas pelo Ocala-i3 na pilha de protocolos TCP/IP (veja Figura 35 em que se ilustra a inserção de duas novas camadas, com mais um encapsulamento IP pela nova camada do Proxy Ocala, à direita). Este comportamento, quando comparado ao protocolo HIP, despertou, novamente, questionamentos sobre a implementação a ser adotada neste trabalho, sugerindo uma proposta na direção de *Non-routing Overlays*.

Observações quanto ao comportamento do Hi3, um protocolo baseado no HIP (um *Overlay Non-Routing*) que utiliza a infra-estrutura do i3 (um *Overlay Routing*) foram realizadas. Quanto a sua vazão de rede, esta apresentou comportamento bem próximo ao OpenHIP nativo, conforme pode ser observado na Figura 34, e superior ao i3, fato que pode ser explicado devido ao Hi3 só utilizar a infra-estrutura do i3 para garantir uma troca inicial de chaves HIP, sendo as trocas de pacotes subseqüentes realizadas somente com a utilização do HIP (i.e. TCP/HIP/IP).

Quanto a utilização de CPU (ver Figura 33) e memória ([48; 97]), este apresentou os maiores valores, sendo superior ao OpenHIP e i3. Inspecionando-se sua arquitetura híbrida, pode-se determinar que tal comportamento foi devido à sobrecarga causada pela análise realizada em cada pacote na decisão se o mesmo deveria ser encaminhado a infra-estrutura do Proxy Ocala-i3 ou enviada diretamente ao *Overlay* OpenHIP.

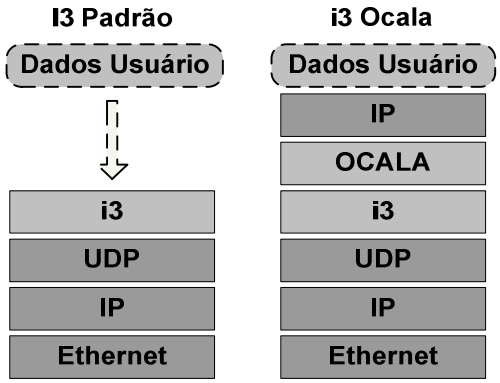

Figura 35. Pilha de Camadas i3 Padrão e Ocala-3

De maneira geral, o bônus obtido com uma riqueza em funcionalidades é sempre acompanhado pelo ônus de impactos no consumo de recursos, tais como CPU, memória ou largura de banda. Este foi o caso do protocolo i3, cujos veementes benefícios (e.g. mobilidade *single-jump* e *double-jump*, suporte a *multicast* e *anycast*, algumas formas de proteção de ataques DoS) apresentaram um alto custo computacional e comprometeram o crescimento em escala. Corroborando esta premissa, podemos observar na Figura 36, a sobrecarga onerada por cada protocolo no tráfego de rede com a inserção de seus cabeçalhos adicionais no transporte de dados. Simulando um ambiente hipotético de uma SAN (*Storage Area Network*) em que grandes volumes de dados em rede representam um desafio para o sistema, pode-se imaginar que certas propostas inviabilizariam o seu funcionamento, como é o caso do Ocala-i3, apresentando a maior sobrecarga.

Os impactos observados quanto à sobrecarga causada por cada protocolo, entretanto, pode ser mitigado no futuro com o crescente uso de técnicas de TCP *offload* [98] e contínuo super provisionamento de recursos computacionais e de rede observado nas últimas décadas. Embora restritos estes testes em redes de alta velocidade, possibilitaram um questionamento intensivo sobre os principais desafios futuros na implantação desta arquitetura em ambientes reais e sujeitos a altas taxas de dados.

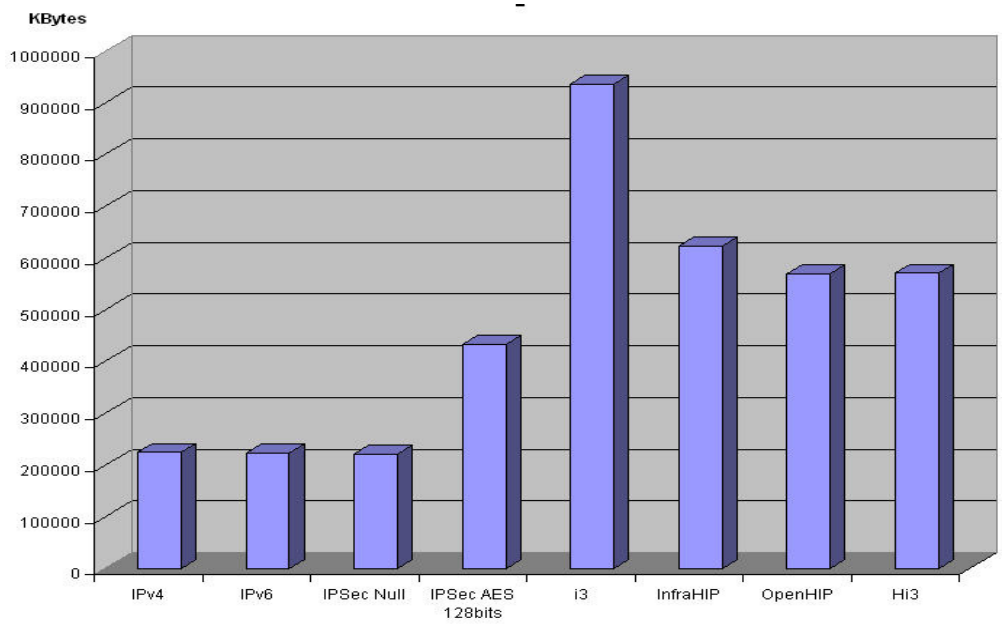

Figura 36. Sobrecarga Verificada pelas Redes Virtuais *Overlays* no Transporte de Dados

Em vista das investigações realizadas, a proposição de uma nova arquitetura para a Internet composta por novas camadas de nomes e serviços de resolução adicionais representa uma fonte de infindáveis discussões em diversos aspectos, tais como o desempenho

(computacional e de rede), crescimento em escala, segurança, motivação para a adoção por parte dos usuários e fabricantes, modelo de economia regente, e etc.

Diante disso, inúmeras questões requerem uma considerável maturação, tanto tecnologicamente (i.e. qual tecnologia usar: redes virtuais com adoção incremental, ou outra tecnologia de substituição?) quanto em vista do crescimento em escala (i.e. onde testar um modelo de rede com as dimensões da Internet? Projetos como o Emulab [99] e PlanetLab [46] apresentariam um cenário satisfatório de bancada de teste?) e também do ponto de vista da viabilidade econômica (i.e. qual a motivação para fabricantes e usuários adotarem este modelo?) entre outras.

A proposição de dois novos espaços de nomes e, consequentemente, três novos serviços de resolução RRS pode duplicar ou triplicar o número de resoluções de nomes, conforme pôde ser observado nas Figura 26 e Figura 27, quando comparado ao RRS atual da Internet, o DNS. Desta forma, espera-se enfrentar um impacto considerável quanto ao aumento da latência na resolução de nomes, fato que pode ser mitigado com a adoção de *caches*.

Entretanto, este artifício requer um profundo estudo prévio à sua adoção, principalmente no que tange à freqüência de atualizações de dados (espera-se, pela inspeção informal do modelo atual TCP/IP, que a atualização de identificadores referentes à camada de rede, ou seja, de nós seja intensa; enquanto a atualização de identificadores de serviços/aplicações seja mediana e, finalmente, que a atualização de nomes semânticos da camada de aplicação seja baixa). Em seguida, uma análise formal do comportamento dos *caches* se faz necessária, e para isso uma modelagem de filas ou Rede de Petri, fiel à complexidade da arquitetura real deve ser concebida o que consiste em fonte de um novo e abstruso trabalho. Finalmente sua implementação/implantação em um ambiente de larga escala, ou bancada de testes seria necessária.

Outro tópico de fundamental importância consiste nas implicações do tipo de DHTs escolhida para a indexação de chaves. Diversos modelos de DHT existem atualmente: as unidimensionais, como a Chord, OpenDHT, Pastry, Tapestry que realizam o roteamento baseadas em uma disposição lógica de nós na forma de um anel simples; as multidimensionais, como a CAN, que realizam o roteamento de chaves baseadas em um espaço de nomes (i.e. chaves) n-dimensional; as esteganográficas, como a Mnemosyne [100], que realizam o armazenamento de chaves de forma totalmente aleatória, geralmente baseado em operações matemáticas realizadas com a senha fornecida pelo cliente, garantindo a privacidade absoluta dos objetos armazenados, entre outras.

A escolha de qual DHT utilizar consiste em um delicado balanceamento entre o estado ótimo do espaço de armazenamento de chaves, isto é, o conhecimento que cada nó formador da DHT possuirá sobre o total de chaves N, versus a ordem de busca do algoritmo empregado no roteamento, ou seja, o número médio de nós consultados para atingir a chave procurada. A multiplicidade de DHTs disponíveis pode variar tão diversamente conforme apresentado na Tabela 6.

Através da inspeção desta tabela pode-se concluir que o processo de adoção de uma candidata em detrimento de outra deve considerar vários fatores, alguns, estruturais, os quais foram expostos na tabela, e outros, tais como a relevância do projeto de pesquisa, sua maturidade, disponibilidade e abrangência geográfica, que foram, por exemplo, fatores adicionais considerados neste trabalho para a adoção da OpenDHT. Detalhes minuciosos da complexidade dos algoritmos de busca versus padrão de armazenamento de cada DHT não compreendem o escopo deste projeto, maiores informações podem ser encontrados em Ramasubramanian (2004) [37].

Tabela 6. Ordem de Resolução Média versus Espaço Amostral de Armazenamento Médio [37]

Categoria, segundo Ramasubramanian (2004) [37]	**Exemplos**	
	Ordem de Consulta	**Ordem de Armazenamento**
Prefixos	**Plaxton apud [37], Pastry, Chord, Tapestry, OpenDHT**	
	O(log N)	O(log N)
Gráficos de Bruijin	**Koorde apud [37], [Wieder & Naor] idem**	
	O(log N/log log N)	O(log N)
Butterfly	**Viceroy apud [37]**	
	O(log N)	O(1)
Farsite	**Farsite apud [37]**	
	O(d)	$O(dN^{1/d})$
Kelips	**Kelips apud [37]**	
	O(1)	$O(\sqrt{N})$
Onisciência	**[Gupta, Liskov, Rodrigues] apud [37]**	
	O(1)	O(N)

De posse dos fatores estruturais de cada DHT candidata e de outros fatores adicionais relevantes, uma terceira questão a ser analisada consiste no ciclo de vida dos

indexadores. Entende-se por ciclo de vida, neste contexto, o tratamento que cada DHT adotará quando da ocorrência de colisões, ou seja, uma tentativa de realizar um armazenamento de um objeto utilizando um índice já existente. Conforme anteriormente discutido, funções de mapeamento não perfeitas estão sujeitas a este tipo de problema e comportamentos passivos (i.e. que permitem a sobrescrita do objeto no referido índice), ativos (não permitindo sobrescrita) ou híbridos (i.e. podem ser ora passivos, ora ativos) precisam ser considerados. O modelo utilizado neste trabalho, híbrido, permite que senhas sejam associadas a certos índices desempenhando um comportamento ativo, contudo, índices sem senhas associadas também podem ser empregados e estes sofrerão sobrescritas de seus valores no caso de colisões.

Considerações sobre a motivação para a adoção desta nova Arquitetura de Nomes baseada em DHTs precisam ser feitas, haja vista a reorientação de intentos no que diz respeito à administração dos objetos (i.e. dados, serviços e usuários). No modelo tradicional da Internet, baseado no TCP/IP, cada proprietário de nomes é responsável pela resolução de suas referências. Com a adoção desta nova Arquitetura, um proprietário de um nó pertencente à DHT, isto é, um nó de resolução de referências (nó RRS), não é capaz de realizar nenhuma inferência quanto às referências armazenadas, não sendo mais responsável exclusivamente por seus objetos.

Desta forma, uma questão fundamental que surge quanto à colaboração de novos participantes é: qual o modelo capaz de viabilizar, do ponto de vista econômico, a colaboração de novos nós RRS? Que tipo de política econômica de tarifação deve ser empregada neste intento (micro pagamentos, modelo de recompensa, baseado em doações, modelos pré-pagos, pós-pagos e etc.)? Este questionamento é de fundamental importância para a subsistência da proposição deste trabalho e sua solução, que deve compreender um modelo de economia para reger esta Arquitetura, representa um completo e desafiador novo trabalho que pode significar o sucesso ou fracasso de sua adoção.

Quanto à colaboração de entidades terceiras ao sistema, que encontrarão motivação para tal a partir do estabelecimento de um modelo econômico atrativo, esta deve impulsionar a sinergia dos módulos existentes (e.g. camada SID, RRS SID, RRS EID e EID) com os módulos futuros (e.g. Máquina de Busca Canônica, Entidades Certificadoras e Entidade QoSC).

Finalmente, um significante desafio que precisa ser transposto consiste no estabelecimento inicial da Arquitetura do ponto de vista do Serviço de Canonização de nomes. Como este serviço deve ser inserido no ambiente de produção da Internet? Quem seria o responsável pela sua implantação/manutenção? O modelo de economia, acima discutido,

apresentaria motivações para que colaboradores quisessem disponibilizar serviços de Canonização?

Diversas possibilidades são admissíveis de serem implementadas. Do ponto de vista da inserção inicial deste serviço, uma possibilidade consiste em utilizar o próprio DNS para alcançar os servidores de Canonização, de forma análoga como é realizado atualmente quando se procura uma Ferramenta de Busca (i.e. *search engines*). Da mesma forma, certos serviços/aplicações estratégicos, como navegadores para a Internet, poderiam disponibilizar nativamente (i.e. *hard-coding*) ferramentas que possuam o conhecimento dos servidores de Canonização (e.g. como acontece atualmente com certos navegadores, como o Mozilla Firefox [101], que possuem recursos nativos para realizar consultas em diversas máquinas de busca previamente conhecidas).

Do ponto de vista da colaboração de empresas/usuários para a constituição destes servidores, acredita-se que a motivação para tal seja, também, análoga ao modelo atual da Internet em que empresas, tais como o Google, Altavista, Yahoo disponibilizam máquinas de busca sem custo adicional para os usuários, obtendo como fonte de receita a prestação de serviços/anúncios para terceiros.

CAPÍTULO 7 CONSIDERAÇÕES FINAIS

Em adição aos resultados técnicos e práticos expostos durante o trabalho, uma importante contribuição deste trabalho à comunidade científica é a compilação de diversas referências esparsas sobre Redes Virtuais e a proposição de mudanças para a Internet em um único texto que tenha como propósito o tratamento científico do projeto de um sistema de nomes para a Internet. A inovação deste trabalho se encontra na combinação destes componentes em uma Arquitetura singular, mais flexível e capaz de suportar diversas características discutidas e ausentes no modelo atual TCP/IP.

Diversas proposições foram realizadas no decorrer desta dissertação e, ainda que algumas tenham sido formalmente esclarecidas através de testes de bancada ou da definição de uma Taxonomia de classificação seguida de uma análise de campo, outras tantas questões permanecem em aberto. Muitos fatores, tais como a brevidade deste trabalho, a necessidade de complexas modelagens (i.e. definição de um modelo de filas, ou rede de Petri [102]) que representem o comportamento da arquitetura com a adoção de *caches*, a definição de um modelo de economia que governe esta infra-estrutura, entre outros, permanecem não resolvidos. Desta forma, algumas considerações quanto aos benefícios obtidos e trabalhos futuros precisam ser delineadas para o melhor entendimento deste trabalho, bem como uma conclusão do mesmo.

7.1 Premissas e Contribuições

Em se tratando de um trabalho almejou a proposição de uma Arquitetura de Nomeação com Múltiplas Camadas para a Internet através do uso de Redes Virtuais, ainda que esta seja uma meta extremamente ambiciosa, diversas abordagens similares são encontradas na literatura [17; 8; 16; 11; 14] e foram aqui discutidas no intento de estabelecer paralelos e dissimilaridades. Um conjunto de premissas foi estabelecido, baseado principalmente nos trabalhos de Saltzer[17], Shoch[79] e Stoica[16] com adequações para a realidade contemporânea da Internet no século XXI, uma complexa rede de computadores cada vez mais congestionada, inundada por inúmeros *middleboxes* e com um número crescente de recursos e usuários requerendo diferentes níveis de qualidade de serviço.

Das proposições anteriormente discutidas no Capítulo 5, vale ressaltar: a necessidade de adoção incremental da nova Arquitetura, através do uso, por exemplo, de Redes Virtuais em conformidade com o Requisito Geral 8# e com o Requisito de Resolução 2#; a concordância com trabalhos da literatura já fundamentados, e.g. princípios de Saltzer, Shoch e Stoica; o uso de um espaço de nomes sem hierarquia para identificar objetos (i.e. dados, serviços e usuários) em concordância com os Requisitos 1#, 2# e 3#; a não interferência em funcionalidade da Arquitetura TCP/IP já fundamentadas, tais como o roteamento e a nomeação DNS como discutido no Requisito 8#; e a máxima reutilização possível de códigos e idéias outrora apresentados na literatura na composição da Arquitetura (a inovação desta proposta será advinda da combinação singular de elementos já existentes em uma disposição tal que componha uma nova e mais funcional Arquitetura de Nomes).

No cumprimento destas premissas, bem como de todos os requisitos discutidos no Capítulo 5, uma inovadora Arquitetura de Nomes é obtida. Sua adoção possibilita os seguintes benefícios para a Internet:

- Mobilidade de nós finais naturalmente suportada, incluindo *double jumps* (i.e. a mobilidade simultânea dos dois nós comunicantes ao mesmo tempo);
- Mobilidade de dados (conteúdo e aplicações);
- Autenticação de nós;
- Autenticação de dados;
- Privacidade;
- Ausência de arbitrações administrativas na escolha e na publicação de novos dados, serviços e usuários e
- Suporte natural a serviços de garantia de Qualidade de Serviço de Conteúdo.

7.2 Conclusões

A proposta de mudanças fundamentais na estrutura da Internet, uma rede heterogênea e de escala planetária, pode representar em um primeiro momento, uma tentativa demasiadamente ambiciosa e sujeita a deveras críticas, em virtude de seu sucesso estar diretamente vinculado à colaboração de empresas e usuários na larga adoção das novas tecnologias propostas. A relativa lenta adoção do IPv6, quando comparada à expectativa inicial de substituição do IPv4, ilustra as incertezas e dificuldades enfrentadas na proposição de mudanças tecnológicas massivas para a Internet.

Neste sentido, Redes Virtuais representam uma oportunidade real de inovação na medida em que possibilitam a inserção de novas e avançadas funcionalidades codificadas em nós virtuais pertencentes à Internet. Estas por sua vez, podem ser inseridas sem estabelecer uma relação de competitividade com a infra-estrutura existente (i.e. em termos de desempenho, confiabilidade e etc.) podendo, por outro lado, coexistir e complementar suas funções, como é o caso da proposição de nomes aqui realizada.

A proposta de uma Arquitetura de Nomeação Com Múltiplas Camadas para a Internet baseada em Redes Virtuais *Overlay* de Propósito Específico para a disposição de dois novos espaços de nomes criptográfico sem hierarquia e, consequentemente, requerendo novos sistemas de resolução de nomes RRS representa uma pequena, mas essencial, mudança. De forma análoga, a introdução de uma funcionalidade para identificar univocamente serviços e dados desvinculadamente da localização do(s) servidor(es) que os hospedam apresenta uma mudança ainda mais significante. Quando unidas, estas duas relativamente simples características apresentam um efeito ainda mais importante, aumentando a confiabilidade e utilização da Internet além de acelerar o desenvolvimento e introdução de novas funcionalidades, aplicações e serviços.

Obviamente, impactos no desempenho em virtude da inserção de novas funcionalidades que alteram o modo como dados e metadados são manipulados (i.e. mapeados e transportados) são previstos. Como conseqüência, exaustivas investigações sobre funcionalidades que venham a diminuir os impactos advindos da inserção de novas camadas e do aumento na complexidade, tais como o uso de *caches,* irá provavelmente impulsionar a Arquitetura de Nomes em direção a uma possível padronização que indiretamente acarretará em melhores práticas de implementação com graus mais elevados de confiabilidade e desempenho.

A busca por uma infra-estrutura de nomes mais flexível que não seja baseada em nomes DNS nem restrita por características da localização ou identificação IP dos nós, em um nível lógico para a Internet, é largamente conhecida [17; 103; 16; 104; 14]. Vislumbra-se, entretanto, que o sistema de referências do futuro deverá ser compreendido por uma infra-estrutura intermediária entre o DNS atual e o requisito geral aqui proposto sob a denominação **Requisito Geral 8#** (um esquema de resolução hibrido, i.e. em parte hierárquico e em parte flat, que associe certas referências a determinados provedores de conteúdo pode vir a ser o modelo mais adequado do ponto de vista econômico).

Consequentemente, qualquer eventual mudança no mecanismo de roteamento tradicional IP ou na infra-estrutura de nomes “original” da Internet seria entendida, neste

trabalho, ainda que o objetivo principal seja o de atingir uma completa abrangência na Internet (i.e. com mecanismos de autenticação bem definidos, conexões resilientes, migração transparente, replicação de conteúdo flexível, e etc.) como um planejamento a ser atingido, primariamente, através do uso de Redes Virtuais *Overlay*.

Ao longo do tempo, acredita-se que o ônus pela adoção de Redes Virtuais sobre uma rede substrato (e.g. a Internet) irá emergir com o bônus da geração de uma nova arquitetura, de forma análoga à origem da Internet (uma Rede Virtual *Overlay* sobre a rede telefônica) influenciada pela antiga infra-estrutura PSTN. Contudo, agora, com a presença de uma infra-estrutura de nomes *Overlay* muito mais sólida, confiável e flexível permiti-se o desenvolvimento de novas aplicações e sua adoção em larga escala (diminuindo sobremaneira o custo inicial de implantação).

7.3 Trabalhos Futuros

Muitas são as possibilidades de projetos futuros e recomendados que podem advir deste trabalho, conforme já discutido na seção anterior em que diversas questões abertas foram apontadas. Em adição à aqueles, os seguintes trabalhos são derivados:

- ✓ Uma investigação técnica baseada em uma bancada de teste das diversas implementações de DHT disponíveis na literatura apontadas no Capítulo 2 e discutidas no Capítulo 7, quanto ao seu desempenho no cenário aqui proposto;
- ✓ A expansão da análise de campo realizada no Capítulo 4 para atender a um número de Redes Virtuais satisfatório (ainda que não seja possível aumentar uma ordem de grandeza em virtude da ausência de tal quantidade numérica na literatura). Esta expansão deve ser seguida de um teste de bancada, análogo ao realizado por [48], para consolidar a validação da Taxonomia [45] na literatura tornado-a uma referência *de facto*.
- ✓ Simulação (e.g. em ferramentas robustas como o *software* OPNet [105]) seguida de implementação completa da Arquitetura de Nomes, tanto para o suporte de novos protocolos, que pode ser feita com a adição de novos *plugins* ao *Proxy*, quanto para a integração com a camada inferior EID e seu RRS seguida por uma exaustiva bateria de testes de bancada em ambientes de produção próximos ao real, i.e. conforme encontrados nos projetos PlanetLab e Emulab. Esta implementação poderia ser então,

comparada à proposição já consagrada do projeto NodeID (*Ambient Networks*) [80] em vigência na comunidade Européia.

REFERÊNCIAS

[1] CARPENTER, B.;BRIM, S. RFC 3234 - Middleboxes: Taxonomy and Issues. **Network Working Group**.Ietf. 2002. Disponível em: <http://www.ietf.org/rfc/rfc3234.txt?number=3234>. Acesso em: 20 jan. 2007.

[2] GUARDINI, I. et al. Ipv6 Operational Experience within the 6Bone. In: INTERNET SOCIETY CONFERENCE, Yokohama, Japan. **Proceedings**. 2000.

[3] BRADEN, R. et al. From protocol stack to protocol heap: role-based architecture. In: 1ST ACM WORKSHOP ON HOT TOPICS IN NETWORKS, ACM Press, Princeton, NJ, USA. **Proceedings**. 2002.

[4] BAKER, F.;SIMPSON, W. A. RFC 1661 - The Point-to-Point Protocol. **Netwok Working Group**.Ietf. July. 1994. Disponível em: <http://www.ietf.org/rfc/rfc1661.txt?number=1661>. Acesso em: 13 fev. 2007.

[5] DROMS, R. RFC 2131 - Dynamic Host Configuration Protocol. **Bucknell University**.N. W. Group. Marc. 1997. Disponível em: <http://www.faqs.org/rfcs/rfc2131.html>. Acesso em: 17 dez. 2006.

[6] XING, S.;PARIS, B.-P. Mapping the growth of the Internet. In: THE 12TH INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS, Los Angeles, CA, USA. **Proceedings**. 2003. p. 199 - 204.

[7] ANDERSEN, D. et al. Resilient Overlay Networks. In: 18TH ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES (SOSP), ACM Press, Chateau Lake Louise, Banff, Canada **Proceedings**. 2001. p. 131-145.

[8] STOICA, I. et al. Internet Indirection Infrastructure. In: ACM SIGCOMM, **Proceedings**. 2002.

[9] NIKANDER, P. et al. Integrating Security, Mobility, and Multi-Homing in a HIP Way. In: NETWORK AND DISTRIBUTED SYSTEMS SECURITY SYMPOSIUM, San Diego, CA, USA. **Proceedings**. 2003. p. 87-99.

[10] CLARK, D. et al. FARA: reorganizing the addressing architecture. In: Proceedings of the ACM SIGCOMM WORKSHOP ON FUTURE DIRECTIONS IN NETWORK ARCHITECTURE, ACM Press Karlsruhe, Germany. **Proceedings**. 2003.

[11] MOSKOWITZ, R.;NIKLANDER, P. Host Identity Protocol Architecture. IETF. January. 2004.

[12] PETERSON, L. et al. A Blueprint for Introducing Disruptive Technology into the Internet. In: FIRST ACM WORKSHOP ON HOT TOPICS IN NETWORKS (HOTNETS-I), Princeton, NJ, USA. **Proceedings**. 2002.

[13] MOORE, D. et al. Inferring Internet Denial-of-Service Activity. In: USENIX SECURITY SYMPOSIUM, Washington, D.C. **Proceedings**. 2001. p. 9-22.

[14] WALFISHA, M. et al. Untangling the Web from DNS. In: 1ST USENIX/ACM SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION (NSDI '04), San Francisco, CA, USA. **Proceedings**. 2004.

[15] RAMASUBRAMANIAN, V.;SIRER, E. G. The design and implementation of a next generation name service for the Internet. In: SIGCOMM '04 SPECIAL INTEREST GROUP ON DATA COMMUNICATIONS, ACM Press, Portland, OR, USA. **Proceedings**. 2004.

[16] BALAKRISHNAN, H. et al. A Layered Naming Architecture for the Internet. In: SIGCOMM, Portland, OR, USA. **Proceedings**. 2004.

[17] SALTZER, J. RFC 1498 - On the naming and binding of network destinations. **Network Working Group**. Ietf. August. 1993. Disponível em: <http://www.ietf.org/rfc/rfc1498.txt?number=1498>. Acesso em: 12 mar. 2007.

[18] BREWER, E. A. Lessons from Giant-Scale Services. **IEEE Internet Computing**, v.5, n.4, July, p.46-55. 2001.

[19] RIPEANU, M. et al. Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design. **IEEE Internet Computing Journal**, v.6, n.1, 2002.

[20] PLAXTON, C. G. et al. Accessing nearby copies of replicated objects in a distributed environment. In: ACM SYMPOSIUM ON PARALLEL ALGORITHMS AND ARCHITECTURES, **Proceedings**. 1997. p. 311-320.

[21] RATNASAMY, S. et al. A scalable content-addressable network. In: ACM SIGCOMM, San Diego, CA, USA. **Proceedings**. 2001. p. 161 - 172.

[22] STOICA, I. et al. Chord: A scalable peer-to-peer lookup service for internet applications. In: ACM SIGCOMM, **Proceedings**. 2001. p. 149-160.

[23] ZHAO, B. Y. et al. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. **U.C. Berkeley**. 2001.

[24] CASTRO, M. et al. Security for structured peer-to-peer overlay networks. In: FIFTH SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION (OSDI'02), Boston. **Proceedings**. 2002.

[25] SCHNEIER, B. Applied Criptography: Protocols, Algorithms, and Source Code in C. 1996.

[26] KUMAR, P. et al. Bandwidth and latency model for DHT based peer-to-peer networks under variable churn. In: ICW '05: PROCEEDINGS OF THE 2005 SYSTEMS COMMUNICATIONS, **Proceedings**. 2005. p. 320.

[27] RHEA, S. et al. Handling Churn in a DHT. In: USENIX Annual Technical Conference, **Proceedings**. 2004.

[28] ZHANG, Z.;LIAN, Q. Reperasure: Replication Protocol Using Erasure-Code in Peer-to-Peer Storage Network In: 21ST IEEE SYMPOSIUM ON RELIABLE DISTRIBUTED SYSTEMS (SRDS'02), IEEE Computer Society, Suita, Japan. **Proceedings**. 2002.

[29] DRUSCHEL, A. R. P. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM INTERNATIONAL CONFERENCE ON DISTRIBUTED SYSTEMS PLATFORMS (MIDDLEWARE), Heidelberg, Germany. **Proceedings**. 2001. p. 329–350.

[30] RHEA, S. et al. OpenDHT: A Public DHT Service and Its Uses. In: ACM SIGCOMM, **Proceedings**. 2005.

[31] DABEK, F. et al. Towards a Common API for Structured Peer-to-Peer Overlays. In: 2ND INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS (IPTPS 03), ACM Press, Berkeley, CA. **Proceedings**. 2003.

[32] CASTRO, M. et al. SplitStream: High-bandwidth multicast in a cooperative environment. In: 2ND INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS (IPTPS 03), Berkeley, CA, USA. **Proceedings**. 2003.

[33] ZHAO, B. Y. et al. Tapestry: A resilient global-scale overlay for service deployment. **IEEE Journal on Selected Areas in Communications**, p.41–53. 2004.

[34] HUEBSCH, R. et al. Querying the Internet with PIER. In: 19TH INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), **Proceedings**. 2003. p. 321–332.

[35] F. DABEK, M. F. K., D. KARGER, R. MORRIS, AND I. STOICA. . Wide-area cooperative storage with CFS. . In: 18TH SYMPOSIUM ON OPERATING SYSTEMS AND PRINCIPLES, Chateau Lake Louise, Banff, Canada. **Proceedings**. 2001.

[36] MUTHITACHAROEN, A. et al. Ivy: A read/write peer-to-peer file system. In: FIFTH SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION, Boston, MA, USA. **Proceedings**. 2002.

[37] RAMASUBRAMANIAN, V.;SIRER, E. G. Beehive: O(1) Lookup Performance for Power-Law Query Distributions in Peer-to-Peer Overlays. In: NETWORKED SYSTEM DESIGN AND IMPLEMENTATION (NSDI), San Francisco, California. **Proceedings**. 2004.

[38] PETERSON, L. et al. PlanetLab Architecture: An Overview. **PlanetLab Consortium**. May. 2006.

[39] RHEA, S. The Bamboo Distributed Hash Table. 2007, 2004.Disponível em: <http://bamboo-dht.org>. Acesso em: 22 nov. 2006.

[40] OpenDHT Server List. **Opendht Organization**. Disponível em: <http://www.opendht.org/servers.txt>. Acesso em: 23 abr. 2007.

[41] CALIFORNIA, I. S. I. F. U. O. S. RFC 793 - Transmission Control Protocol. September. 1981. Disponível em: <http://www.faqs.org/rfcs/rfc793.html>. Acesso em: 13 mar. 2007.

[42] WEINREICH, H. et al. Scone Framework. Disponível em:<http://www.scone.de/>. Acesso em: 14 jan. 2007.

[43] FOUNDATION, F. S. GNU General Public License. Boston, MA, USA.Disponível em: <http://www.gnu.org/copyleft/gpl.html>. Acesso em: 11 fev. 2007.

[44] ANDERSON, T. et al. Overcoming the Internet impasse through virtualization. **IEEE Computer Society,** v.38, n.4, Apr 2005, p. 34- 41. 2005.

[45] SILVA, M. A. D. L. E. et al. A Proposal for a Taxonomy for Virtual Networks. In: 2ND REAL OVERLAYS AND DISTRIBUTED SYSTEMS (ROADS), ACM SIGCOMM, Warsaw, Poland. **Proceedings**. 2007.

[46] CHUN, B. et al. PlanetLab: An Overlay Testbed for Broad-Coverage Services. **ACM SIGCOMM Computer Communication Review**, v.33, n.3, 2003.

[47] NIKANDER, P. et al. Host Identity Indirection Infrastructure (Hi3). In: THE SECOND SWEDISH NATIONAL COMPUTER NETWORKING WORKSHOP 2004 (SNCNW2004), **Proceedings**. 2004.

[48] QUAINI-SOUSA, J. et al. Suitability of Overlays as a General-Purpose Data Communication Substrate on Gigabit Channels. In: 18TH IASTED INTERNATIONAL CONFERENCE ON

PARALLEL AND DISTRIBUTED COMPUTING AND SYSTEMS (PDCS 2006), Dallas, Texas, USA. **Proceedings**. 2006.

[49] QUAINI-SOUSA, J. et al. An Overlay-based Flat Layered Architecture for the Internet. In: FIRST INTERNATIONAL CONFERENCE ON REAL OVERLAYS AND DISTRIBUTED SYSTEMS (ROADS 2007), Belem, Para, Brasil. **Proceedings**. 2007.

[50] AMIR, Y.;DANILOV, C. Reliable Communication in Overlay Networks. In: INTERNATIONAL CONFERENCE ON DEPENDABLE SYSTEMS AND NETWORKS, Institute of Electrical and Electronics Engineers, Inc., San Francisco, CA. **Proceedings**. 2003.

[51] HAGENS, R. A. et al. RFC 1070 - Use of the Internet as a Subnetwork for Experimentation with the OSI Network Layer. **Network Working Group**.Ietf. 1989. Disponível em: < http://www.ietf.org/rfc/rfc1070.txt?number=1070 > Acesso em:

[52] ERIKSSON, H. MBONE: the Multicast Backbone. In: COMMUNICATIONS OF THE ACM, ACM Press New York, NY, USA New York, NY, USA **Proceedings**. 1994. p. 54 - 6.

[53] ADKINS, D. et al. Towards a more functional and secure network infrastructure. **Computer Science Division (EECS), University of California**. Berkeley. 2003.

[54] FRANCIS, P.;GUMMADI, R. IPNL: A NAT-extended Internet architecture. In: ACM SIGCOMM CONFERENCE ON NETWORK ARCHITECTURES AND PROTOCOLS 2001, ACM Press, San Diego, CA, USA. **Proceedings**. 2001.

[55] SUBRAMANIAN, L. et al. OverQoS: An Overlay based Architecture for Enhancing Internet QoS In: FIRST SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION (NSDI '04), ACM Press, San Francisco, California, USA. **Proceedings**. 2004.

[56] KOVACIC, S. F. General taxonomy of system[ic] approaches for analysis and design. In: IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN AND CYBERNETICS, **Proceedings**. 2005. p. 2738.

[57] BEDFORD, D. Redefining the Search Question. 2005. Disponível em: <http://www.collectionscanada.gc.ca/obj/014005/f2/014005-05209-c-e.pdf> Acesso em: 20 mai. 2007.

[58] POUWELSE, J. A. et al. The Bittorrent P2P File-Sharing System: Measurements and Analysis. In: 4TH INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS (IPTPS), Ithaca, New York, USA. **Proceedings**. 2005.

[59] ZENNSTRÖM, N.;FRIIS, J. Joost. Disponível em:<http://www.joost.com/>. Acesso em: 1 fev. 2007.

[60] ZENNSTRÖM, N.;FRIIS, J. Skype. 2007. Disponível em:<http://www.skype.com>. Acesso em: 24 mar. 2007.

[61] MICHAEL J. FREEDMAN, E. F., AND DAVID MAZIÈRES. Democratizing Content Publication with Coral. In: 1ST USENIX/ACM SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION, USENIX, The Advanced Computing Systems Association, San Francisco, California, USA. **Proceedings**. 2004.

[62] TSCHUDIN, C.;GOLD, R. Network Pointers. In: 1ST ACM WORKSHOP ON HOT TOPICS IN NETWORKS, Princeton, NJ. **Proceedings**. 2002.

[63] TENNENHOUSE, D. L.;WETHERALL, D. J. Towards an Active Network Architecture. In: SIGCOMM COMPUTER COMMUNICATION REVIEW, ACM, San Jose, CA, USA. **Proceedings**. 1996.

[64] WANG, L. et al. Reliability and security in the CoDeeN content distribution network. In: USENIX ANNUAL TECHNICAL CONFERENCE, **Proceedings**. 2004.

[65] PARK, K. S.;PAI, V. S. Scale and Performance in the CoBlitz Large-File Distribution Service. In: THIRD SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION (NSDI 2006), San Jose, CA. **Proceedings**. 2006.

[66] **F5 Networks Inc,** 2007. Disponível em:<https://secure.f5.com/Infopage/index.jsp>. Acesso em: 02 mar. 2007.

[67] **JOOST.** 2007. Disponível em: <http://www.joost.com>. Acesso em: 14 mai. 2007.

[68] JANOTTI, J. et al. OverCast: Reliable Multicast with an Overlay Network. In: FOURTH SYMPOSIUM ON OPERATING SYSTEM DESIGN AND IMPLEMENTATION (OSDI), **Proceedings**. 2000.

[69] **SKYPE.** 2007. Disponível em: <http://www.skype.com/intl/pt/products/skypeout/>. Acesso em: 11 fev. 2007.

[70] GRITTER, M.;CHERITON, D. R. An Architecture for Content Routing Support in the Internet. In: USENIX SYMPOSIUM ON INTERNET TECHNOLOGIES AND SYSTEMS, **Proceedings**. 2001.

[71] TOUCH, J. Dynamic Internet Overlay Deployment and Management Using the X-Bone. In: EIGHTH ANNUAL INTERNATIONAL CONFERENCE ON NETWORK PROTOCOLS, IEEE Computer Society, Osaka, Japan. **Proceedings**. 2000. p. 117-135.

[72] TENNENHOUSE, D. L.;WETHERALL, D. J. Towards an Active Network Architecture. In: SIGCOMM COMPUTER COMMUNICATION, ACM Press, New York, NY, USA. **Proceedings**. 1997.

[73] BREITKREUZ, H. Emule Project. 2002. Disponível em:<http://www.emule-project.net>. Acesso em: 23 jun. 2007.

[74] **CoBlitz Homepage.** Disponível em: <http://codeen.cs.princeton.edu/coblitz/>. Acesso em: 13 jul. 2007.

[75] COMER, D.;STEVENS, D. L. **Internetworking with TCP/IP**. ed. Prentice Hall. 2000.

[76] HOUSLEY, R. RFC 3686 - Using Advanced Encryption Standard (AES) Counter Mode with IPsec Encapsulating Security Payload (ESP). **Network Working Group**.Ietf. January. 2004. Disponível em: < http://www.ietf.org/rfc/rfc3686.txt?number=3686 >. Acesso em: 18 ago. 2006.

[77] O'DONNELL, M. Open network handles implemented in DNS. **Internet Draft-Odonnell-Onhs-Imp-Dns**. 2002.

[78] O'DONNELL, M. A proposal to separate Internet handles from names. 2003. Disponível em: < http://people.cs.uchicago.edu/~odonnell/Citizen/Network_Identifiers >. Acesso em: 03 abr. 2007.

[79] SHOCH, J. Inter-Network Naming, Addressing, and Routing. In: IEEE COMPCON, **Proceedings**. 1979. p. 72-79.

[80] NIEBERT, N. et al. Ambient Networks – Research for Communication Networks Beyond 3G. In: 13TH IST MOBILE & WIRELESS COMMUNICATIONS SUMMIT 2004, Lyon, France. **Proceedings**. 2004.

[81] **Google Brasil.** 2007. Disponível em:<http://www.google.com.br/>. Acesso em: 12 mai. 2007.

[82] **AltaVista,** 2007. Disponível em:<http://www.altavista.com/>. Acesso em: 12 mai. 2007.

[83] **Yahoo**, 2007. Disponível em:<http://www.yahoo.com>. Acesso em: 12 mai. 2007.

[84] **InfraHIP Project,** 2006**.** Disponível em: <http://infrahip.hiit.fi/>. Acesso em: 12 mai. 2007.

[85] GODARD, **S. SysStat.** Disponível em: <http://perso.orange.fr/sebastien.godard/>. Acesso em: 23 abr. 2007.

[86] OZMENT, A.;SCHECHTER, S. E. Bootstrapping the Adoption of Internet Security Protocols. In: FIFTH WORKSHOP ON THE ECONOMICS OF INFORMATION SECURITY (WEIS 2006), Cambridge, UK. **Proceedings**. 2006.

[87] MOREIRA, J. E. et al. Scalability of the Nutch search engine. In: 21ST ANNUAL INTERNATIONAL CONFERENCE ON SUPERCOMPUTING, New York, NY, USA. **Proceedings**. 2007.

[88] BARHAM, P. et al. Xen and the Art of Virtualization. In: 19TH ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES (SOSP 2003), ACM Press, Bolton Landing, NY, USA. **Proceedings**. 2003. p. 164 -177.

[89] ROSE, R. Survey of System Virtualization Techniques. March, 8. 2004. Disponível em: <http://citeseer.ist.psu.edu/720518.html >. Acesso em: 26 jun. 2007.

[90] KALLAHALLA, M. et al. SoftUDC: A Software-Based Data Center for Utility Computing. 37**:** p.38 - 46, 2004.

[91] RIGBY, S.;DARK, M. Designing a Flexible, Multipurpose Remote Lab for the IT Curriculum. In: 7TH CONFERENCE ON INFORMATION TECHNOLOGY EDUCATION, **Proceedings**. 2006. p. 161-164.

[92] JOSEPH, D. et al. OCALA: An Architecture for Supporting Legacy Applications over Overlays.

[93] **OpenHIP Project**. 2006. Disponível em: <http://www.openhip.org/>. Acesso em: 09 jul. 2007.

[94] **NetPerf Project,** 2006**.** Disponível em: <http://www.netperf.org/netperf/NetperfPage.html>. Acesso em: 05 fev. 2007.

[95]**IPerf Project.** 2006. Disponível em:<http://dast.nlanr.net/Projects/Iperf/>. Acesso em: 21 mar. 2007.

[96] **Ethereal - Network Protocol Analyzer**. 2003. Disponível em:<http://www.ethereal.com/>. Acesso em: 19 abr. 2006.

[97] REDÍGOLO, F. et al. Evaluating Overlays for Gigabit Channels Data Communication. In: 3RD INTERNATIONAL CONFERENCE ON TESTBEDS AND RESEARCH INFRASTRUCTURES FOR THE DEVELOPMENT OF NETWORKS AND COMMUNITIES, Orlando, Florida, USA. **Proceedings**. 2007.

[98] MOGUL, J. C. TCP Offload Is a Dumb Idea whose Time Has Come. In: 9TH WORKSHOP ON HOT TOPICS IN OPERATING SYSTEMS (HOTOS IX), Lihue, Hawaii, USA. **Proceedings**. 2003.

[99] **EMULAB. Total Network Testbed**. 2007. Disponível em: <http://www.emulab.net/>. Acesso em: 13 jan. 2007.

[100] HAND, S.;ROSCOE, T. Mnemosyne: Peer-to-Peer Steganographic Storage. In: 1ST INTERNATIONAL WORKSHOP ON PEER-TO-PEER SYSTEMS (IPTPS '02), Boston, MA, USA. **Proceedings**. 2002.

[101] **Mozilla Firefox Project**. 2007. Disponível em:< http://www.mozilla.com/firefox/>. Acesso em: 15 jun. 2007.

[102] SUN, H. et al. A Generic Availability Model for Clustered Computing Systems. In: PACIFIC RIM INTERNATIONAL SYMPOSIUM ON DEPENDABLE COMPUTING, Seoul, Korea. **Proceedings**. 2001. p. 241.

[103] CASTRO, M. et al. One Ring to Rule Them All: Service Discovery and Binding in Structured Peer-to-Peer Overlay Networks. In: 10TH ACM SIGOPS EUROPEAN WORKSHOP, Saint-Emilion, France. **Proceedings**. 2002.

[104] PARK, K. et al. CoDNS: Improving DNS performance and reliability via cooperative lookups. In: 6TH USENIX OPERATING SYSTEM DESIGN AND IMPLEMENTATION, ACM SIGOPS, California, USA. **Proceedings**. 2004.

[105] **TECHNOLOGIES, O. OPNET Modeler.** 2001. Disponível em: <http://www.mil3. com/products/modeler/home.html>. Acesso em: 06 fev. 2007.