FELIPE ROLIM PELLISSARI

RBRP: Protocolo de Reputação Baseado em Papéis para Redes Peer-to-Peer

FLORIANÓPOLIS
2005

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RBRP: Protocolo de Reputação Baseado em Papéis para Redes Peer-to-Peer

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Ciência da Computação.

FELIPE ROLIM PELLISSARI

Florianópolis, Fevereiro de 2005.

RBRP: Protocolo de Reputação Baseado em Papéis para Redes Peer-to-Peer

Felipe Rolim Pellissari

'Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, Área de Concentração em *Sistemas de Computação*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina.'

_____
Carla Merkle Westphall, Dra.
Orientadora

_____
Raul Sidnei Wazlawick, Dr.
Coordenador do Curso de Pós-Graduação em Ciência da Computação

Banca Examinadora:

_____
Carla Merkle Westphall, Dra.
Presidente

_____
Carlos Alberto Maziero, Dr.

_____
Carlos Becker Westphall, Dr.

_____
Luiz Carlos Zancanella, Dr.

_____
Vitório Bruno Mazzola, Dr.

As coisas que tomamos por suposições sem questioná-las ou refletir sobre elas, são justamente as que determinam o nosso pensamento consciente e decidem as nossas conclusões.
John Dewey

AGRADECIMENTOS

Agradeço primeiramente a Deus, pois sem ele não teríamos motivos para viver ou mesmo existir e por colocar em minha vida as mais magníficas pessoas.

Agradeço à minha orientadora professora Dra. Carla Merkle Wespthall por todo o apoio que me prestou durante todo o curso, além da ajuda fundamental no desenvolvimento deste trabalho.

Agradeço à minha família, meu pai Carlos, minha mãe Márcia e minha irmã Cristina, pelas alegrias proporcionadas e pela ajuda e conforto nos momentos difíceis.

Desejo também agradecer aos amigos e colegas de mestrado Rafael, Marcelo, Alexandre, Fabrício, Aujor, Breno, Marcos, Valério, Kleber, Fabiano e Fábio pela ajuda nas disciplinas, pela troca de conhecimento e pelos momentos de confraternização que passamos juntos.

Agradeço aos amigos Ivan, Kazuo e Sandrini pela amizade nos momentos de descanso em Itararé, pelas cervejas bebidas e pelas piadas contadas.

Agradeço também aos companheiros da UEL: Alan, Quiles, Vitão, Chicão, Érica, Josi, Camila, Stênio, Sasilva, Mota, Psico, Renato, Oluap, Alynne, Caldera, Pimenta, João e Alex pela força sem a qual não chegaria sequer a terminar a graduação.

Agradeço ao professor Dr. Carlos Westphall pelas orientações e ajudas em diversas situações.

Agradeço ao LRG e ao INE pelo suporte proporcionado para a realização do curso.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

RBRP: Protocolo de Reputação Baseado em Papéis para Redes Peer-to-Peer

Felipe Rolim Pellissari

Fevereiro/2005

Orientadora: Profa. Carla Merkle Westphall, Dra.
Área de Concentração: Segurança em Redes de Computadores
Palavras-chave: Segurança Computacional, Redes Peer-to-Peer
Número de Páginas: xiii + 82

Grande parte do tráfego de dados existente na Internet atualmente pertence a aplicações peer-to-peer para troca de arquivos. Uma das maiores preocupações dos usuários dessas aplicações é encontrar a fonte mais segura de um determinado recurso. Os protocolos de reputação se propõem a resolver esse problema usando experiências anteriores pelos nós para encontrar a melhor fonte de um determinado recurso. Esta dissertação define o Role-Based Reputation Protocol (RBRP), um protocolo de reputação para redes peer-to-peer baseado em papéis para definição de valores de reputação de um sistema peer-to-peer, fazendo assim uma classificação dos nós de uma rede de acordo com o comportamento exercido pelos nós.

O RBRP, assim como outros protcolos de reputação, auxilia no provimento de segurança em uma rede peer-to-peer elegendo os nós do sistema com maior chance de possuir um recurso válido, ou seja, um recurso sem a presença de vírus ou de conteúdo impróprio, como pornografia infantil. Entretanto, como o RBRP usa uma classificação baseada em papéis para valores de reputação, o protocolo usa uma abordagem diferente para enfrentar o mesmo problema. Essa abordagem está mais próxima dos usuários das redes, portanto deve possuir uma melhor aceitação com relação ao seu uso.

O trabalho mostra que, com o pleno uso do RBRP, uma rede pode ter o consumo de largura de banda reduzido. Isto se deve ao fato do protocolo prevenir trocas inválidas, ou seja, o RBRP evita desperdício de largura de banda, já que, além dos danos potenciais que vírus podem trazer a um sistema, a transferência de tal conteúdo gera tráfego na rede.

Por fim, o trabalho mostra a implementação do RBRP em uma rede peer-to-peer para verificar a validade da implantação de um protocolo de reputação em um sistema real. A implementação desenvolvida mostra que um protocolo de reputação baseado em papéis possui uma interface mais amigável, auxiliando na aceitação e no pleno uso do protocolo pelos usuários da rede.

Abstract of Dissertation presented to UFSC as a partial ful_llment of the requirements for the degree of Master in Computer Science.

RBRP: Role-Based Reputation Protocol for Peer-to-Peer Networks

Felipe Rolim Pellissari

Fevereiro/2005

Advisor: Prof. Carla Merkle Westhall, Dra.
Area of Concentration: Network Security
Key words: Computer Security, Peer-to-Peer Networks
Number of Pages: xiii + 82

Most of Internet data traffic belongs to peer-to-peer file-sharing applications. One major concern of these application users is to find the safest source of a resource. Reputation protocols intend to solve this problem using nodes' past experiences to find the best source of a resource. This work defines the Role-Based Reputation Protocol (RBRP), a reputation protocol for peer-to-peer networks based on roles for a peer-to-peer system reputation values definition, classifying the nodes of a network based on these nodes' behavior.

RBRP, as other reputation protocols, helps improving security in a peer-to-peer network selecticing system's nodes with better chance to have a valid resource, that is, a resource without viruses or improper containent, such as child pornography. Although, as RBRP uses a role-based reputation, the protocol uses a different approach to face the same problem. This approach is closer to networks' users, so must have a better acceptance to its use.

The work shows that a network may have its bandwidth reduced by using RBRP. This happens because the protocol prevents invalid resource exchanges, that is, RBRP avoids bandwidth wastes because, beyond viruses' potencial damages may cause to a system, the exchange of such resouces generates network traffic.

Finally, the work presents a RBRP implementation in a peer-to-peer network to evaluates a role-based reputation protocol implantation in a real system. The developed implementation shows that a role-based reputation protocol possess a friendlier interface, helping acceptance among network' users.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

Abreviatura	**Significado**
AES	Advanced Encryption Standard
CORC	Credit Only Reputation Control
DAC	Discretionary Access Control
DCRC	Debit-Credit Reputation Control
DES	Data Encryption Standard
DHT	Distributed Hash Table
DNS	Domain Name Server
DoS	Denial-of-Service
HTTP	Hyper-Text Markup Language
IP	Internet Protocol
LOTOS	Language of Temporal Ordering Specification
MAC	Mandatory Access Control
MIME	Multi-Purpose Internet Mail Extensions
P2P	Peer-to-Peer
RBAC	Role-Based Access Control
RBRP	Role-Based Reputation Protocol
RCA	Reputation Control Agent
RSA	Rivest-Shamir-Adelman
SDSI	Simple Distributed Security Infrastructure
SPKI	Simple Public Key Infrastructure
TCP	Transport Control Protocol
THA	Trust Holding Agents
TI	Tecnologia da Informação
TTL	Time-To-Live

Capítulo 1 Introdução

As redes de computadores revolucionaram os meios de comunicação a partir da explosão demográfica da Internet, a partir do começo da década de 90. Inicialmente, a Internet era composta por grandes servidores que ofereciam uma variedade de recursos e informações aos usuários comuns, ligados na rede através de seus computadores pessoais.

Uma das características desses computadores pessoais era a baixa capacidade de processamento e armazenamento que possuíam. Sendo assim, eram ineficientes no provimento de recursos, portanto apenas requisitavam recursos da Internet sem oferecer nada em troca.

Entretanto, no final da década de 90, esses computadores pessoais conectados à rede, em sua maioria, possuíam certa capacidade de processamento e armazenamento. Por outro lado, os grandes servidores estavam sobrecarregados devido ao aumento do número de máquinas clientes e da maior variedade de serviços que precisavam oferecer.

Devido a esses fatos, o modelo cliente-servidor de comunicação usado na Internet começou a apresentar sérios problemas com escalabilidade. Foi então que um novo modelo de troca de informações surgiu na Internet: as redes peer-to-peer (p2p). Esses sistemas colaborativos possuem como principal característica a autonomia significativa dos nós da rede com relação a um servidor central. Essa característica aproveita-se da capacidade de processamento e armazenamento dos computadores pessoais e, além disso, alivia as necessidades de processamento e armazenamento de um servidor central.

Em um curto período de tempo diversas aplicações baseadas no modelo peer-to-peer surgiram na Internet. Como algumas delas sequer fazem distinção entre um servidor e um cliente, a escalabilidade dessas redes é muitas vezes superior a uma aplicação cliente-servidor [Emule, 2004]. Entretanto, como os nós da rede normalmente possuem as mesmas funcionalidades, como busca e troca de recursos, autenticação, autorização, a rede está sujeita a sérios problemas de segurança, como a disseminação de vírus e materiais de conteúdo impróprio pelo sistema, já que se um servidor central

de responsabilizar por garantir a segurança da rede a escalabilidade do sistema é comprometida.

Os protocolos de reputação em redes peer-to-peer têm um papel importante no provimento de segurança da rede. Esses protocolos são responsáveis para auxiliar os nós a encontrar a fonte mais confiável de um determinado recurso que desejem utilizar. Normalmente os protocolos baseiam-se em experiências anteriores para calcular um valor numérico que represente um valor de confiança com relação a um determinado nó possuidor do recurso ou mesmo ao próprio recurso (Pellissari et al., 2004).

Quando a aplicação se encarrega de calcular esse valor numérico e atribuir valores numéricos de confiança de acordo com regras pré-definidas, os protocolos funcionam bem somente se tais regras forem válidas. Porém, fica inviável definir regras pré-definidas globais para atribuição de valores de confiança, já que cada nó de uma rede p2p pode necessitar uma política de segurança diferente.

Para seres humanos é difícil atribuir um valor numérico que represente um valor de confiança com relação a outro ser humano. Seres humanos costumam usar rótulos que indicam confiança como: “amigo”, “colega”, “vizinho” ou “aluno”. No mundo real, quando um ser humano deseja saber se outro é confiável, pergunta a um terceiro conhecido de ambos e recebe uma resposta do tipo “ele é amigo”. Nota-se que esses rótulos reproduzem papéis que outros seres humanos representam. Portanto, os protocolos de reputação que seguissem essa idéia estariam mais próximos da realidade dos seres humanos e teriam melhor aceitação em uma rede peer-to-peer cujos nós representam seres humanos.

Este trabalho define o protocolo Role-Based Reputation Protocol (RBRP), um protocolo baseado em papéis para cálculo de reputação e valores de confiança para nós de uma rede peer-to-peer. O trabalho apresenta, de maneira detalhada todo o funcionamento do protocolo, incluindo: a especificação formal, os dados armazenados pelas bases de dados, as mensagens trafegadas, as delegações do usuário e a implementação do protocolo. A principal característica desse protocolo está na forma como reconhece um valor de confiança. Enquanto os demais protocolos pesquisados procuram quantificar em números valores de confiança, o RBRP atribui nomes (ver

seção 5) relacionados com papéis aos valores de confiança. Isso, além de organizar os valores de confiança em grupos reproduzidos por papéis, auxilia os usuários na atribuição de confiança a outros nós. O RBRP também se preocupa em separar os valores de confiança com relação ao tipo de dado desejado. Isso é importante porque, por exemplo, uma boa fonte de arquivos MP3 não é necessariamente uma boa fonte de arquivos-texto.

Outro detalhe importante com relação ao RBRP e a outros protocolos de reputação é a largura de banda utilizada. Alguns autores se preocupam em comprovar que a carga gerada pelas mensagens desses protocolos não gera um aumento significativo na largura de banda utilizada pelo sistema. Esse trabalho comprova matematicamente que o RBRP é capaz de diminuir a largura de banda utilizada por uma rede p2p, já que com a utilização desse protocolo a quantidade de trocas de recursos inválidas diminui significativamente.

1.1 Objetivo e Justificativa

O objetivo geral desse trabalho é desenvolver um protocolo de reputação baseado em papéis, capaz de realizar uma classificação dos nós de um sistema p2p de acordo com os papéis que desempenham na rede. Os objetivos específicos são:

- Desenvolver um conjunto de mensagens que devem compor o protocolo;
- Definir as informações que devem ser armazenadas em cada nó;
- Descrever uma especificação formal do protocolo;
- Implementar o protocolo.

A justificativa para a realização desse trabalho é a ausência de classificação dos nós de um sistema p2p pelos protocolos pesquisados (Kamvar et al., 2003, Damiani et al., 2002, Cornelli, 2002, Singh e Lui, 2003, Gupta et al., 2003, Righi et al., 2004). Essa classificação é útil, pois agrupa os nós com um comportamento semelhante em grupos de confiança.

Pela ausência e importância dos tópicos citados no estado-da-arte em redes peer-to-peer, este trabalho é motivado a preencher tais lacunas na área de pesquisa do tema abordado.

1.2 Organização

Este trabalho está dividido em 7 capítulos. O Capítulo 2 apresenta os aspectos teóricos referentes à segurança computacional, focando principalmente os conceitos de propriedades, mecanismos e política. Além disso, esse capítulo faz uma breve descrição sobre SPKI/SDSI.

O Capítulo 3 descreve os conceitos básicos e arquiteturas relevantes de redes peer-to-peer e conclui fazendo uma descrição mais detalhada sobre o protocolo Gnutella. Esse capítulo também relata alguns conceitos sobre segurança em redes peer-to-peer. No capítulo 4 são apresentados os trabalhos relacionados com reputação em redes peer-to-peer, tema que pertence à área de segurança em redes p2p.

O Capítulo 5 mostra a descrição, arquitetura e modelo formal do protocolo apresentado nessa dissertação, chamado de RBRP, e o seu modo de operação. O Capítulo 6 mostra o modelo de implementação desse protocolo, os resultados obtidos a partir de tal implementação e um estudo matemático sobre a largura de banda consumida em trocas de recursos em redes com o RBRP e sem o uso RBRP.

O trabalho encerra no Capítulo 7 com as conclusões finais. Nesse capítulo também são mostradas idéias para possíveis trabalhos futuros.

Capítulo 2 Segurança Computacional

2.1 Introdução

Antes do surgimento da Internet, havia uma maior preocupação com segurança física no mundo computacional. Gigantescos *mainframes* eram mantidos atrás de muros e cadeados para que os dados ali mantidos permanecessem em sigilo e livres de danos por parte de invasores.

Atualmente esse cenário mudou. Hoje, a grande maioria dos nós participantes da Internet é desconhecida por outras entidades e não é possível assegurar que todos os nós sejam confiáveis. Devido a este fato, hoje em dia são realizados vários estudos sobre segurança na Internet. A segurança física não basta para enfrentar as ameaças digitais presentes no mundo. Segundo (Stallings, 2003), a segurança de sistemas está entre as áreas da computação com maior proeminência, especialmente pela importância da segurança no cotidiano das pessoas e negócios empresariais.

A segurança computacional não se preocupa apenas em garantir a segurança em uma grande rede de computadores como a Internet. Segurança computacional é a disciplina que auxilia a encontrar soluções para enfrentar as ameaças em torno da computação. Portanto, pode-se incluir em segurança computacional não somente problemas advindos do surgimento da Internet, mas também problemas com relação a códigos maliciosos que podem ser distribuídos de outras formas, como disco magnético.

Dentre os problemas existentes na segurança computacional, em especial em sistemas distribuídos como a Internet, pode-se destacar o problema da autenticação de entidades. A autenticação é uma propriedade da segurança que tenta garantir que as entidades envolvidas são quem realmente se intitulam. A autenticação pode ser referente a um usuário do sistema ou a um recurso. Em sistemas distribuídos, principalmente sistemas de grandes proporções como a Internet, existe uma grande preocupação em procurar identificar corretamente usuários e recursos para que usuários ou códigos maliciosos não ataquem o sistema.

Este capítulo define conceitos básicos sobre segurança computacional que são pertinentes e essenciais neste trabalho. Além disso, são apresentados alguns pontos importantes para a construção de uma correta política de segurança, essencial em sistemas distribuídos de larga escala.

2.2 Conceitos

Segundo (Ferreira, 1999), segurança significa *estado, qualidade ou condição de seguro* e a palavra seguro, segundo o mesmo autor significa *livre de risco; protegido, acautelado, garantido*. Segundo (Venter e Elloff, 2003) a segurança computacional se importa com a proteção de ativos digitais armazenados em computadores e redes de processamento de dados. Já segundo (Landwehr, 1981), a segurança é a capacidade de manter um recurso livre de preocupações com ameaças ou vulnerabilidades. Portanto, seja qual for a forma de manter um ativo distante de tais ameaças ou vulnerabilidades, pode-se considerar o ativo como seguro. Sendo assim, pode-se considerar a palavra "seguro" como um atributo de um sistema ou objeto (Landwehr, 2001).

Um sistema é seguro se ele mantém as seguintes propriedades fundamentais (Landwehr, 2001):

- **Confidencialidade**: A confidencialidade é o ato de preocupar-se em manter uma informação sigilosa ou secreta, ou seja, tornar uma determinada informação acessível somente após sua devida autorização. Em sistemas distribuídos, deve-se

ter um cuidado especial com mensagens em trânsito, a fim de evitar a obtenção de informações através da espionagem de um canal de comunicação. É particularmente importante em sistemas militares.

- **Integridade**: Manter a integridade da informação significa manter os dados sem modificação, a menos que exista a devida autorização para sua modificação. Podem ser considerados erros de integridade, além de acesso não-autorizado e modificação da informação, perda da informação ou erros de comunicação. Em uma comunicação, é importante tornar a comunicação livre de erros de repetições de mensagens, conhecidos como proteção de *replay*. A integridade pode ser alcançada de duas formas diferentes: ou prevenindo a ocorrência de falhas em geral ou detectando e recuperando-se delas (Moffett, 1995). Um problema também que pode ser considerado de integridade é a cópia não-autorizada de dados (Landwehr, 1981).

- **Disponibilidade**: Manter a informação disponível a usuários autenticados e autorizados durante todo o tempo é um requisito para a segurança da informação e é considerado o de mais difícil obtenção, já que, como máquinas às vezes falham, é na pratica impossível manter a informação disponível 100% do tempo principalmente em um sistema de comunicação. Assim como a integridade, a disponibilidade pode ser alcançada tanto na forma de prevenção quanto na forma de detecção e recuperação usando dispositivos de *backup*.

Para se manter algo distante de ameaças e vulnerabilidades, é preciso primeiro reconhecê-las. Uma ameaça define ou identifica circunstâncias, condições ou eventos que forneçam algum potencial de violação de segurança, ou seja, de violação das propriedades fundamentais.

Uma vulnerabilidade segundo (Ferreira, 1999) significa *qualidade ou estado de vulnerável* e vulnerável significa *diz-se do lado fraco de um assunto ou de uma questão, ou do ponto pelo qual alguém pode ser atacado ou ferido*. Uma vulnerabilidade é uma falha ou característica indevida que pode ser explorada para concretizar uma ameaça.

Um ataque é identificado como um conjunto de ações conduzidas por uma entidade não autorizada visando violações de segurança. Um invasor pode aproveitar vulnerabilidades dos sistemas ou, até mesmo, aproveitar ameaças que existam no ambiente para concretizar o ataque.

2.3 Políticas de Segurança

O principal recurso da sociedade para garantir segurança advém da imposição de regras ou leis definindo um padrão de comportamento à sociedade. O não cumprimento de uma lei resulta em uma punição ou castigo ao infrator. Desta forma, as leis impedem ou ao menos tentam inibir pessoas de infringi-las, garantindo direitos básicos aos cidadãos.

Dentro do âmbito da computação, a segurança não é diferente. Segundo (Landwehr, 2001), a computação sem uma política de segurança é como uma sociedade sem leis. Política de segurança pode-se entender como os objetivos gerais de uma organização com relação aos riscos de segurança, e os planos para lidar com esses riscos em acordo com os objetivos (Moffett, 1995).

Uma política de segurança deve responder às seguintes perguntas (Moffett, 1995):

1. Quais itens devem ser protegidos e quais seus valores?

2. Quais são as ameaças a esses itens?

3. Quais ameaças devem ser eliminadas e por quais meios?

Fica claro, observando a primeira questão, que um perímetro deve ser definido dentro da política de segurança de uma organização. Esse perímetro deve definir principalmente os limites de segurança dentro da organização, ou seja, o domínio de segurança.

Um dos objetivos de definir-se um domínio de segurança é distinguir usuário de intrusos (Landwehr, 2001). Um intruso é um indivíduo que atravessa um perímetro de

segurança sem autorização. Uma vez que um intruso pode, dentro do sistema, comportar-se como um usuário autêntico, mas obtendo acesso à informação confidencial ou não-autorizada, precisa-se de meios para garantir a autenticidade dos usuários.

Pode-se observar que intrusos são um tipo de ameaça que responde à pergunta dois. Outros tipos de ameaças em uma organização podem ser encontrados, podendo-se tomar como exemplo o vírus. Um vírus é um programa que pode "infectar" outros programas, modificando-os e corrompendo-os (Stallings, 2003).

As ferramentas conhecidas para detecção e remoção de vírus são conhecidas como *antivírus*. Considerando, então, que na pergunta dois acima a ameaça ao sistema é um vírus, a resposta da pergunta três pode ser respondida como um antivírus. É importante notar que a política de segurança pode variar muito de uma organização para outra, portanto uma política de segurança para uma organização deve ser bem específica.

As políticas de segurança podem ser classificadas em: segurança física, segurança gerencial e segurança lógica (Mello, 2003). A política de segurança física preocupa-se na proteção do meio físico, isto é, com o uso de trancas e cofres para proteção da informação. A política de segurança gerencial se preocupa com o ponto de vista organizacional, ou seja, definir os processos de segurança que são importantes durante o funcionamento do sistema. A política de segurança lógica define a distribuição de direitos de uso aos usuários, isto é, faz o controle de acesso ao sistema junto com a autenticação de usuários.

2.4 Propriedades e Princípios

Embora as políticas de segurança possam diferir muito umas com relação às outras, todas elas devem seguir certas propriedades básicas para a garantia de segurança. Segundo (Landwehr, 2001), para manter um sistema seguro é preciso obedecer três propriedades básicas: confidencialidade, integridade e disponibilidade (seção 2.2). Em sistemas distribuídos pode-se citar além destas propriedades outras duas: autenticação e não-repúdio.

Autenticação: A autenticação é importante pois pode ser considerada o ponto inicial de entrada de um sujeito em um sistema. A autenticação consiste da identificação válida de parceiros na comunicação e/ou da origem de dados. Além disso, a autenticação serve como base para auditoria e organização de contas de usuários e é um pré-requisito para controle de acesso em sistemas baseados em identidades de usuários do sistema (Moffett, 1995).

Não-repudiação: Não-repudiação, ou não-repúdio, é a prevenção contra a negação do recebimento ou do envio de uma mensagem, por parte de um usuário, conhecido como prova de entrega e prova de origem. É importante em qualquer situação em que os interesses da origem e destino dos dados podem entrar em conflito, como por exemplo, em aplicações de comércio eletrônico ou aplicações bancárias.

Além das propriedades, alguns princípios básicos de segurança são importantes para melhorar a segurança em um sistema. A seguir são apresentados alguns desses princípios.

Responsabilidade: As pessoas tendem a se comportar melhor se sabem que podem responder por seus atos. Este princípio baseia-se na idéia de armazenar todas as operações realizadas por usuários no sistema. Para que funcione, é preciso um sistema de autenticação eficaz, já que de nada serve manter os rastros deixados por um invasor se ele não pode ser identificado.

Menor privilégio: O principio do menor privilégio afirma que a cada entidade em um sistema computacional deve ser garantido apenas o privilégio necessário para desempenhar sua respectiva função (Landwehr, 2001). Se uma entidade possui um privilégio maior que o necessário esta pode querer se aproveitar desse fato para realizar algo fora de sua função e, portanto, ilegal.

Defesa em profundidade: Este princípio reconhece que mecanismos de segurança podem falhar devido principalmente a vulnerabilidades no sistema de segurança. Conseqüentemente, projetos de segurança devem ser implementados usando mecanismos em níveis múltiplos e complementares, tal que, mesmo que um nível de segurança seja atacado, outro nível subseqüente proteja a informação.

2.5 Mecanismos de Segurança

Para implementar uma política de segurança, é necessário garantir as propriedades de segurança através de determinados mecanismos de segurança, ou seja, mecanismos de segurança são as formas pelas quais a política de segurança pode ser implementada. Esta seção descreve alguns mecanismos mais conhecidos.

Mecanismos de segurança física: Mecanismos de segurança física são usados para proteção de equipamentos e para controle de acesso fora do escopo de controle de acesso lógico ou criptografia. São necessários para riscos tais como fogo, ataques terroristas e danos acidentais causados por usuários e técnicos. Mecanismos de segurança podem ser: preventivos (cadeados), de detecção (circuito de TV fechado) ou recuperação (disco de *backup*).

Autenticação pessoal: O foco da autenticação em sistemas computacionais é verificar a identidade de um usuário. O modo mais comum é o uso de *login* e senha, mas para a autenticação pessoal pode ser usado: algo que a pessoa é (impressão digital), algo que a pessoa tem (*smart card*) ou algo que a pessoa sabe (senha) (Landwehr, 2001). Normalmente, quanto mais fatores providos na autenticação mais forte a autenticação pode ser considerada. É importante aqui separar a propriedade de autenticação com o mecanismo de autenticação pessoal. A propriedade é algo mais genérico que pode ser aplicado à autenticação de códigos, indo além da autenticação de usuários.

Autorização: Uma vez que a autenticação é feita com sucesso, um sistema deve verificar quais as operações permitidas pelo usuário autenticado. O processo de autorização tem dois modelos mais conhecidos: *Discretionary Access Control* (DAC) e *Mandatory Access Control* (MAC). O modelo DAC permite associar usuários a permissões. O tipo MAC, associa rótulos aos usuários e objetos ou recursos do sistema; os rótulos dos objetos seguem uma classificação específica enquanto os usuários ou os sujeitos do acesso possuem níveis de habilitação. Controles que determinam as autorizações de acesso são baseados numa comparação da habilitação do usuário com a classificação do objeto. Outro modelo que também está sendo bastante utilizado é o

modelo *Role-Based Access Control* (RBAC), um modelo de controle de acesso que diferencia os usuários por seus papéis representados em um sistema (Bertino, 2003).

Operações de auditoria: Operações de auditoria procuram manter registro de todas as operações de cada usuário do sistema, a fim de buscar possíveis culpados em casos de falhas do sistema. Na prática, funciona como um sistema interno de câmeras de TV, pois, apesar de não prevenir um ato ilegal, registra as ações do atacante. Para que a auditoria funcione bem, é preciso um mecanismo de autenticação eficaz. Arquivos nos quais as operações de auditorias são executadas são conhecidos como *logs*.

Criptografia: Um algoritmo criptográfico transforma uma mensagem original em um texto cifrado incompreensível através do uso de uma chave criptográfica. (Tanembaum, 1995). O texto cifrado só se torna compreensível através da aplicação da função inversa do algoritmo criptográfico com a mesma ou outra chave. A criptografia simétrica usa a mesma chave criptográfica para cifrar e decifrar uma informação, enquanto que a criptografia assimétrica utiliza chaves diferentes para cifrar e decifrar uma mensagem. O algoritmo de criptografia simétrico mais usado atualmente é conhecido como *Data Encryption Standard* (DES), apesar do novo padrão *Advanced Encryption Standard* (AES) estar tomando seu lugar, já que o algoritmo DES é computacionalmente inseguro atualmente. Já na criptografia assimétrica o algoritmo mais usado é o RSA (Rivest-Shamir-Adelman), que apesar de ser ordens de grandeza mais lento que os algoritmos simétricos, é utilizado principalmente para troca de chaves simétricas e assinaturas digitais.

2.6 Confiança

Para que duas entidades troquem informações em um sistema distribuído, é necessário que uma confie na outra. O primeiro passo para a confiança é, portanto, a autenticação de entidades. Existem diversos mecanismos de autenticação de entidades em sistemas distribuídos. Os mais conhecidos são: Kerberos, X.509, SPKI/SDSI e SESAME. Esses mecanismos geram certificados digitais, que são garantias de

identidade das entidades ou assinaturas de conteúdos por parte dos participantes do sistema.

Os quatro mecanismos diferem principalmente no modo como é feito o espaço de nomes e no modo como é feita a autenticação. No X.509 e no SESAME, o espaço de nomes é global, enquanto que no Kerberos e no SPKI/SDSI o espaço de nomes é local. Já quanto ao modo de autenticação, o SPKI/SDSI é o único que usa o modo descentralizado. Devido a esse fato, o SPKI/SDSI é o que mais se enquadra nos propósitos desse trabalho.

O fato do modo de autenticação ser descentralizado traz problemas e benefícios. Um problema é o fato de uma entidade ter que se autenticar em cada ponto do sistema para utilizá-lo. Além disso, a criação de cadeias de confiança é mais complexa. Entretanto, o sistema se torna mais tolerante a faltas, pois um problema em uma autoridade certificadora não desvirtua todo o sistema, ao contrário do que ocorre em mecanismos centralizados.

Todos esses mecanismos têm em comum a sua utilização. Esses mecanismos são usados para implementar uma identificação e um modelo de autorização em sistemas computacionais. Mas o que realmente os diferencia é o modo de autenticação, já que o modelo de autorização é sempre local em todos os mecanismos citados. Apesar disso, a funcionalidade que interessa a este trabalho é o modelo de autenticação. O modelo de controle de acesso e autorização é deixado de lado por não ser do escopo deste trabalho.

2.7 SPKI/SDSI (Simple Public Key Infrastructure / Simple Distributed Security Infrastructure)

O SPKI/SDSI é um mecanismo de infra-estrutura de chaves públicas. Um mecanismo de infra-estrutura de chaves públicas é um mecanismo que garante, através do uso de criptografia assimétrica, a autenticação de entidades em um sistema (Ellinson, et. al., 1999).

O grande benefício que traz o SPKI/SDSI é a ausência de uma estrutura hierárquica, como existe no X.509, ou seja, o SPKI/SDSI é um modelo cuja principal característica é que cada entidade do sistema é uma autoridade certificadora. No SPKI/SDSI a estrutura de nomes é local, diferente também de outros modelos como o X.509.

No SPKI/SDSI cada par de chaves privada/pública representa um principal. Um principal, segundo (Mello, 2003), é um termo utilizado para identificar usuários, processos, máquinas atuando em nome de usuários do sistema, que são considerados aptos pela política de segurança lógica estabelecida em suas ações no sistema. Para o sistema como um todo, o principal é referenciado apenas por sua chave pública, já que a chave privada é mantida em segredo pelo principal.

Em outras abordagens, os principais são referenciados por um nome, não por uma chave pública diretamente. Esse tipo de abordagem é menos flexível, já que um nome é mais propenso a alterações. Além disso, a possibilidade de existirem nomes iguais é bem maior que a existência de um par de chaves igual.

Como cada principal gera seu próprio par de chaves, deve-se considerar a possibilidade de duas entidades gerarem o mesmo par de chaves. Entretanto, na opinião de (Elien, 1998), como uma chave pública possui aproximadamente 305 dígitos decimais, essa possibilidade pode ser ignorada.

Em abordagens com espaço de nomes global, um nome é referenciado globalmente por todo o sistema e é identificado unicamente por todo o espaço que o sistema abrange. No SPKI/SDSI o espaço de nomes é local, ou seja, um nome só é válido localmente, não possuindo um significado global. Porém, esse fato é solucionado pela construção de cadeias de certificação. As cadeias de certificação são assinaturas ligando as chaves umas às outras (Molva, 1999).

Um espaço de nomes local é pertencente a um principal e referencia outros principais ou outros espaços de nomes. A escolha dos nomes de espaço é totalmente arbitrária, e está normalmente associada a: localização, funções, parentesco, etc. Assim é possível a um principal organizar grupos e referenciar a todos por um nome em comum.

```
(public-key
 (rsa-pkcs1-md5
  (e #03#)
  (n
   |ANHCG85jXFGmicr3MGPj53FYYSY1aWAue6PKnpFErHhKMJa4HrK4WSKTO
   YTTlapRznnELD2D7lWd3Q8PD0lyi1NJpNzMkxQVHrrAnIQoczeOZuiz/yY
   VDzJ1DdiImixyb/Jyme3D0UiUXhd6VGAz0x0cgrKefKnmjy410Kro3uW1| )))
```

Figura 2.1: Exemplo de uma S-Expression que define uma chave pública RSA.

O formato dos objetos SPKI/SDSI está registrado em (Ellinson et al., 1999) e é da forma de S-Expressions. Uma S-Expression é uma lista de elementos fechada por parênteses, sendo que esses elementos podem ser cadeias de caracteres ou outras S-Expressions. As S-Expressions são derivações da linguagem LISP (Elien, 1998). A Figura 2.2 apresenta um exemplo de uma chave RSA no formato de uma S-Expression.

2.8 Nomes SDSI

Assim como o SPKI define um principal como sendo uma chave, o SDSI define como deve ser a referência a tais chaves através de um espaço de nomes. Segundo (Mello, 2003), os nomes SDSI ligam principais a identificadores locais, ou seja, somente poderão ser referenciados dentro do espaço de nomes do principal que os definiu. Pode-se ver, na Figura 2.1, que a chave mostrada não está relacionada com nenhum nome, pois para que isso acontecesse deveria aparecer um nome associado após a chave.

```
(name
 (public-key
  (rsa-pkcs1-md5
   (e #03#)
   (n
    |ANHCG85jXFGmicr3MGPj53FYYSY1aWAue6PKnpFErHhKMJa4HrK4WSKTO
    YTTlapRznnELD2D7lWd3Q8PD0lyi1NJpNzMkxQVHrrAnIQoczeOZuiz/yY
    VDzJ1DdiImixyb/Jyme3D0UiUXhd6VGAz0x0cgrKefKnmjy410Kro3uW1| )))
```

```
paulo)
```

Figura 2.2: Exemplo de uma S-Expression que define uma relação de um nome com uma chave pública.

O método de referência de nomes com chaves é simples. A Figura 2.2 ilustra um exemplo. Nela, suponhamos que um usuário do sistema encontre a chave da Figura 2.1 e deseje associá-la com o nome paulo. Basta apenas acrescentar uma identificação que a chave é relacionada com o nome.

Do mesmo modo é possível fazer uma relação de um nome com uma S-Expression. Digamos que outro usuário deseja relacionar a chave da Figura 2.1 com dois nomes: pai e carlos, ou seja, deseja informar que a chave é relativa ao pai de Carlos. A S-Expression ficaria como mostra a Figura 2.3.

```
(name
 (public-key
  (rsa-pkcs1-md5
   (e #03#)
   (n
    |ANHCG85jXFGmicr3MGPj53FYYSY1aWAue6PKnpFErHhKMJa4HrK4WSKTO
    YTTlapRznnELD2D7lWd3Q8PD0lyi1NJpNzMkxQVHrrAnIQoczeOZuiz/yY
    VDzJ1DdiImixyb/Jyme3D0UiUXhd6VGAz0x0cgrKefKnmjy410Kro3uW1| )))
carlos
pai)
```

Figura 2.3: Exemplo de uma S-Expression que define uma relação de nomes.

É importante notar que essa relação tem valor local, ou seja, só é válida para o detentor de tal informação (neste caso quem recebe a S-Expression). Para um usuário informar outros de tais relações que ele cria, ele deve gerar um certificado de definição de nomes.

2.9 Certificados SPKI/SDSI

Os certificados SPKI/SDSI têm duas finalidades. A primeira delas é informar aos outros principais do sistema sobre as relações de nomes que realizam. A segunda finalidade é emitir certificados de autorização de acesso a um recurso. Como não faz parte do escopo do trabalho, essa segunda finalidade não é considerada aqui.

```
(cert (issuer (name K N)) (subject P) {valid})
```

Figura 2.4a: Formato de um certificado de definição de nomes.

```
(cert
 (issuer
  (name
   (public-key
    (rsa-pkcs1-md5
     (e #03#)
      (n
       |ANHCG85jXFGmicr3MGPj53FYYSY1aWAue6PKnpFErHhKMJa4HrK4WSKTO
       YTTlapRznnELD2D7lWd3Q8PD0lyi1NJpNzMkxQVHrrAnIQoczeOZuiz/yY
       VDzJ1DdiImixyb/Jyme3D0UiUXhd6VGAz0x0cgrKefKnmjy410Kro3uW1
| )))
    amigos))
  (subject
   (name
    (public-key
     (rsa-pkcs1-md5
      (e #03#)
       (n
        |
KDQ6Y2VydCg2Omlzc3Vlcig0Om5hbWUoNDpoYXNoMzptZDUxNjpPGjPUbEr
        +4G8lvHemsiETKTQ6ZnJlZCkpKDc6c3ViamVjdCg0Omhhc2gzOm1kNTE2Om
e
        acQg+uGMIEtSGOEYetaApKSg5Om5vdC1hZnRlcjE5OjIwMDEtMDEtMDFf|
)))
       B))
```

Figura 2.4b: Exemplo de um certificado de definição de nomes.

Certificados de definição de nomes são utilizados para publicar nomes locais a fim de que os outros principais possam chegar à chave pública de um nome definido localmente (Mello, 2003). Um certificado de definição de nomes é formado pelos campos (K, N, P, VS). K é a chave do emissor, ou seja, quem assina o certificado. O valor N é o identificador, que juntamente com a chave do emissor forma o nome local que identifica o certificado. P é o sujeito do certificado. O sujeito pode ser uma chave ou um nome, que é formado por uma chave pública seguida de um ou mais identificadores. Finalmente, VS é uma identificação de validade do certificado, sendo o único campo opcional. VS normalmente adota a forma (T1, T2), sendo que T1 é o tempo inicial da validade e T2 o tempo final. A Figura 2.4a mostra a forma de um certificado em S-Expression.

Em um sistema distribuído, assume-se que cada entidade possua um par de chaves e cada nome deve estar associado à zero ou mais chaves para gerar uma cadeia de nomes. Por exemplo, o nome “paulo” da Figura 2.3 está relacionado apenas com uma

chave. No entanto, o nome pai pode estar relacionado com diversas chaves ou outros nomes.

Um exemplo está ilustrado na Figura 2.5. Nela, pode-se ver uma rede peer-to-peer composta de diversos nós. O nó A possui boas relações com os nós B, D e E e deseja criar um grupo composto por tais nós e chamar o grupo de “amigos”. Para tanto, basta ele emitir um certificado de nomes para cada um associando o nome “amigo” com a chave de cada um dos nós que deseja incluir no grupo. A Figura 2.4b mostra um exemplo de um certificado emitido pelo nó A (chave ANHC...) relacionando o nó B (chave KDQ6...) ao nome “amigos”.

Esses grupos são normalmente usados para emissão de certificados de autorização a um grupo de usuários. Por exemplo, se o nó A da Figura 2.5 deseja emitir um certificado de autorização de acesso a um determinado recurso a todos seus nós amigos, basta emitir um único certificado relacionado com o nome “amigos”.

Além dos pontos citados, o nó A do exemplo anterior é capaz de informar a outros pontos do sistema quem faz parte de seus nós amigos. Usando essa funcionalidade o sistema acaba por gerar uma cadeia de certificados com os nós próximos criando relacionamentos entre eles. O nó B, por exemplo, poderia perguntar ao nó A quais são seus amigos e adicionar esses nós no seu próprio grupo de nós amigos, ou então, criar um grupo “amigos de amigos”.

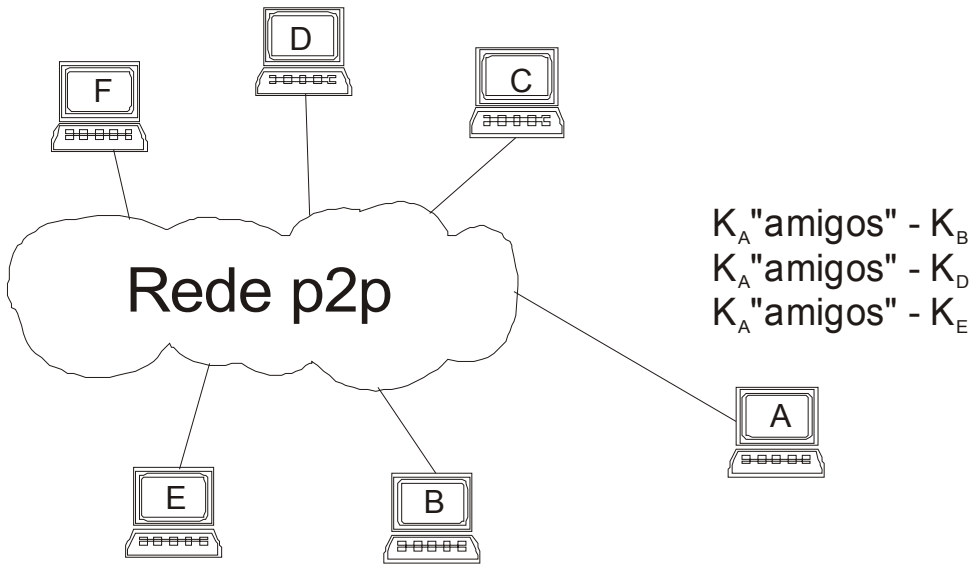

Figura 2.5: Exemplo de um grupo formado por certificados de nomes.

2.10 Conclusão

Este capítulo apresentou as definições básicas sobre segurança computacional, dando ênfase à parte de autenticação de entidades e SPKI/SDSI. Existe um grande interesse de profissionais para que se contenha a quantidade de ataques aos sistemas distribuídos, principalmente a Internet.

Com relação aos conceitos de segurança, nota-se que todo trabalho envolvendo segurança computacional deve seguir certos conceitos e princípios a fim de obter resultados satisfatórios. Os mecanismos de segurança auxiliam na realização da política de segurança, que deve ser específica para atender as necessidades de um sistema.

A autenticação é bastante importante, já que sem um mecanismo eficiente pode-se ter problemas em outras propriedades, como integridade dos dados se estes forem alterados por um atacante erroneamente identificado como um usuário legítimo em um sistema. Devido a esse fato, o capitulo procurou abordar os mecanismos de autenticação existentes e justificar a decisão de buscar um que mais se adequasse às necessidades de sistemas descentralizados como redes peer-to-peer.

Com o uso de SPKI/SDSI é possível criar cadeias de nomes e assim interligar os pontos de um sistema distribuído. Essa interligação cria uma rede de confiança dentro do sistema, importante para o controle de acesso e identificação de nós confiáveis no sistema. A identificação de nós confiáveis é algo extremamente importante em sistemas distribuídos de larga escala que possuem a propriedade de anonimidade, como as redes peer-to-peer.

Capítulo 3 Redes Colaborativas Peer-to-Peer

3.1 Introdução

No final da década de 90, houve uma mudança significativa no poder computacional dos computadores pessoais. Estes, que antes tinham muito pouco poder computacional se comparado com os grandes servidores que abasteciam a internet com os recursos que ela utilizava, começaram a adquirir um poder computacional suficiente para oferecer algo à rede.

Quando uma rede de computadores apresenta um nível razoável de computação nas bordas, isto é, apresenta um grau de computação considerável fora dos servidores, a rede pode ser chamada de rede peer-to-peer, ou simplesmente p2p (Rocha et al., 2004). As redes p2p surgiram no final da década de 90 e desde então transformaram a internet, antes basicamente dotada de provedores de serviço no centro e de clientes requisitando seus recursos nas bordas. Agora é capaz de grandes transações de dados entre as bordas graças ao surgimento das redes p2p, que aumentaram a participação computacional dos antigos clientes que utilizavam recursos sem nada a oferecer em troca.

Os primeiros sistemas p2p que surgiram foram o Napster (Napster, 2004) e o Gnutella (Gnutella, 2004). Com o surgimento dessas aplicações alguns computadores pessoais localizados nas margens da internet se tornaram grandes provedores de recursos. Isso se tornou possível porque, embora os servidores tradicionais sempre tenham tido mais poder computacional e capacidade de armazenamento que os computadores pessoais, os sistemas p2p aproveitam a capacidade dos computadores pessoais de modo que um servidor não é capaz de atingir o poder de computação do sistema sozinho.

Este capítulo faz uma abordagem geral sobre as redes p2p. O capítulo mostra as definições mais usuais sobre redes p2p, mostra os conceitos sobre o tema e quais os requisitos necessários para uma rede ser considerada p2p. Além disso, o capítulo descreve como exemplo as redes Napster e Gnutella e o impacto que elas causaram na internet.

3.2 Conceitos

Um dos principais desafios que apareceram quando se começou a discutir p2p é como identificar e definir uma aplicação p2p. Como já mencionado, as redes p2p começaram a surgir no final dos anos 90 e proporcionaram uma nova funcionalidade à internet. Essa funcionalidade está diretamente associada ao poder computacional das aplicações p2p.

As aplicações tradicionais da internet são baseadas em um modelo cliente/servidor. Nesse modelo, um cliente faz uma requisição de um recurso e o servidor retorna, se possível, o recurso desejado. É importante notar que nesse modelo o servidor é uma máquina de grande poder computacional que atende não a um, mas a diversos clientes ao mesmo tempo.

A Figura 3.1a ilustra esse cenário descrito. Pode-se observar que os servidores estão concentrados no centro da rede e o poder computacional se concentra junto deles. Os clientes apenas utilizam-se dos recursos sem oferecer nada em troca. Para diversos tipos de aplicações, esse modelo é satisfatório.

No modelo p2p a computação existente é mais distribuída ao longo da rede, como mostra a Figura 3.1b. Pode-se observar, nesse modelo, que os servidores, apesar de terem um poder computacional maior que os pontos das margens do sistema, contribuem de forma igual ao funcionamento da aplicação. Esse modelo aproveita melhor os recursos do sistema, otimizando o poder da rede. Também se vê que no modelo p2p existem muitas comunicações entre os nós marginais da rede, o que praticamente não ocorre em aplicações do modelo cliente/servidor.

(Oram, 2001) define dois pontos cruciais para identificar uma rede como sendo p2p:

1. Permite conectividade variável e endereços de rede temporários?

2. Fornece aos nós nas margens uma autonomia significativa com relaçãop à rede?

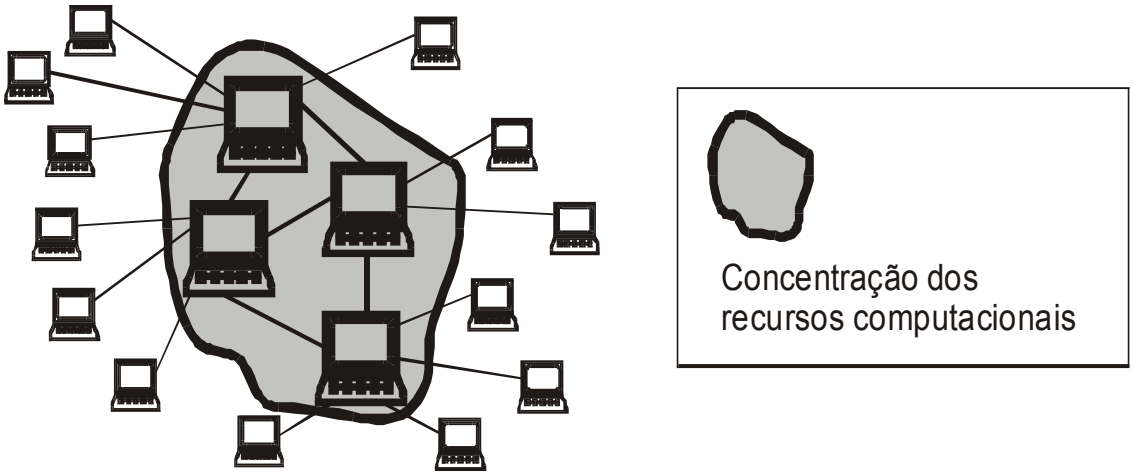

Figura 3.1a: Modelo Cliente/Servidor

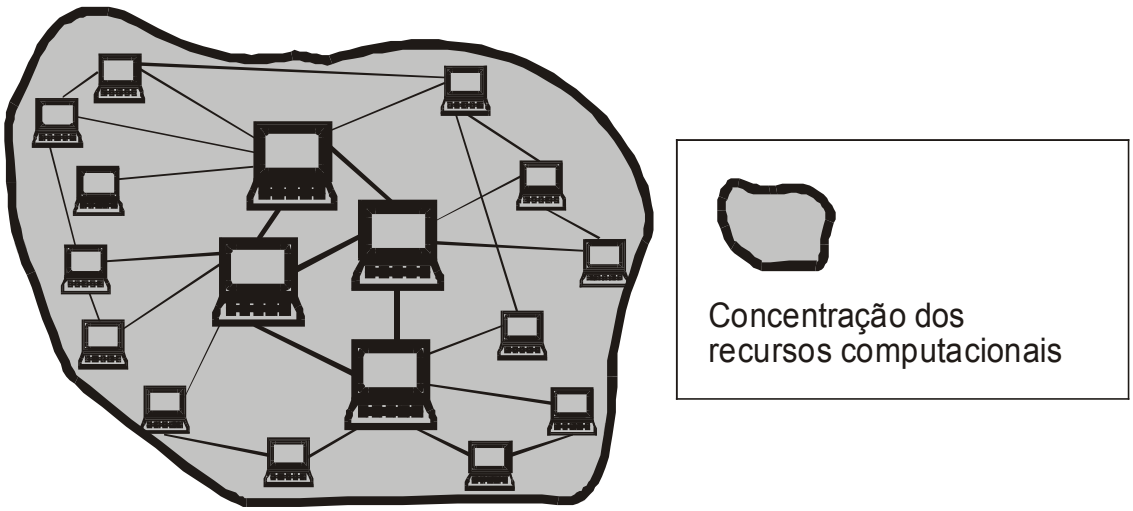

Figura 3.1b: Modelo p2p

Se a resposta a ambas essas perguntas for afirmativa, a aplicação é p2p, caso contrário não é. Um ponto citado nessa definição é o uso de endereços temporários. Segundo (Oram, 2001), os pontos das redes p2p não devem estar diretamente

relacionados a um endereço IP fixo, como um servidor de páginas HTTP. Esse fato permite ao sistema uma maior dinâmica no sentido que permite que os nós entrem e saiam da rede com maior rapidez. O segundo ponto é a autonomia das margens. Entretanto, (Oram, 2001) afirma que o sistema deve ter certa autonomia para funcionar sem a centralização existente no modelo cliente/servidor.

O termo redes colaborativas peer-to-peer é freqüentemente usado para reforçar a idéia que a principal função das redes p2p é a troca de recursos entre os participantes. Pode-se citar então, que outro ponto importante na definição de uma rede p2p é sua funcionalidade: as redes p2p são usadas principalmente para troca de recursos entre os nós marginais da rede, também chamados de *servents*. Denomina-se *servent* o participante das redes peer-to-peer que têm, ao mesmo tempo, funções de servidor (quando oferecem recursos, respondem a consultas ou delegam serviços) e cliente (quando requisitam recursos ou consultas).

No modelo cliente/servidor, os clientes requisitam recursos apenas dos servidores. Se um cliente desejar oferecer um recurso, ele deve convencer um servidor a disponibilizar esse recurso aos outros participantes do sistema. Em redes colaborativas p2p esse processo é simplificado porque os recursos não precisam passar por um intermediário, no caso do modelo cliente/servidor o servidor, para serem disponibilizados para os nós participantes do sistema.

Existem outras definições para redes p2p. (Rocha et al., 2004) define as seguintes propriedades:

- Nós podem estar localizados nas bordas da rede;
- Nós com conectividade variável ou temporária e endereços também temporários;
- A capacidade de lidar com diferentes taxas de transmissão entre nós;
- Nós com autonomia parcial ou total com relação a um servidor centralizado;
- Assegurar que os nós possuem capacidades iguais de fornecer e consumir recursos de seus *peers*;

- A rede deve ser escalável;
- A capacidade dos nós se comunicarem diretamente uns com os outros.

Todas essas características das redes p2p apresentadas conduzem ao questionamento: existe alguma diferença entre uma rede p2p e um sistema descentralizado? Em muitos pontos os conceitos são iguais, porém os sistemas descentralizados não são necessariamente independentes de um protocolo inferior, como no caso do IP. Já as redes p2p devem ter uma independência relativa do protocolo IP. Diversos autores chamam as redes que independem das camadas inferiores de redes *overlays* (Crespo e Garcia-Molina, 2002). As redes *overlays*, no entanto, não necessariamente possuem a descentralização presente nas redes p2p.

3.3 Arquiteturas p2p

As redes p2p podem ser organizadas sob diversos tipos de aspectos. Embora todas elas possuam problemas de segurança em comum, cada arquitetura possui seus próprios problemas. Nessa subseção são apresentados os modelos arquiteturais comuns em redes p2p, mostrando as vantagens e os problemas normalmente encontrados.

As redes p2p podem ser divididas de duas formas: com relação à centralização ou não da rede e com relação à estruturação da busca de recursos. Redes p2p, com relação à sua distribuição, podem ser organizadas em três tipos: centralizadas, descentralizadas ou descentralizadas com uso de supernós. As redes centralizadas possuem um nó central que realiza alguma função especial, como controle de acesso ou busca de recursos. Redes descentralizadas são as chamadas redes p2p puras, nas quais cada nó tem exatamente as mesmas funcionalidades que os outros. As redes com supernós possuem nós com maior capacidade de armazenamento e processamento que se elegem para realizar funções especiais como busca de recursos. A diferença com redes centralizadas é que esses nós podem se tornar supernós somente por um determinado período de tempo e a rede deve ser capaz de funcionar mesmo que nenhum supernó esteja presente.

Já a busca de recursos nas redes p2p pode ser feita de três modos segundo (Gupta, 2003). O primeiro, a busca centralizada, acontece quando um nó central realiza as buscas e armazena os dados necessários. Este modelo não existe em redes p2p puras. O segundo é a desestruturada, a qual um nó que busca determinado recurso faz uma pergunta aos nós mais próximos e estes ou respondem afirmativamente ou repassam a pergunta a outros nós. Nesse modelo a busca é mais simples de ser implementada, porém a busca pode se tornar menos efetiva ou mais lenta, já que nós muito distantes têm dificuldades de se comunicarem. O outro modelo, chamado de estruturado, é quando uma rede p2p como um todo mantém uma tabela *hash* distribuída, ou DHT (*Distributed Hash Table*), com a localidade dos recursos. Nessa arquitetura, cada nó armazena uma parte da DHT e eles fazem perguntas à tabela. Os nós também são responsáveis por manter a DHT atualizada. Essa arquitetura é mais elaborada e exige maiores cuidados com segurança, apesar disso, se for bem utilizada, leva a uma melhor utilização dos recursos de uma rede p2p caso não exceda na utilização da largura de banda para operações de manutenção da tabela.

Uma outra classificação para redes p2p que existe é quanto ao conteúdo das informações transitadas pela rede. Segundo (Cornelli et al., 2002), as redes podem ser do tipo: (1) Troca de mensagens, (2) Computação distribuída e (3) Troca de recursos. Do tipo troca de mensagens pode-se incluir aplicações como ICQ, MSN Messenger e Yahoo! Messenger. A aplicação SETI@Home é um tipo de aplicação de computação distribuída. Já as aplicações de troca de recursos são as mais conhecidas e entre elas estão o Napster, O KaZaA e Gnutella.

3.3.1 Redes p2p Centralizadas

Do ponto de vista de segurança, as redes p2p centralizadas são as que possuem melhores condições de se tornarem seguras. Isto porque a entidade centralizadora se responsabiliza pela maior parte da segurança da rede. Embora possa parecer que redes p2p centralizadas não possuam riscos grandes à segurança, alguns pontos devem ser observados para que se obtenha uma rede p2p segura.

Normalmente, uma rede p2p centralizada pressupõe um nó central que é responsável por algumas funcionalidades ausentes nos outros nós, sendo assim, redes centralizadas possuem uma hierarquia com relação aos nós. Por exemplo, no caso do Napster, a rede possui um nó central que realiza as buscas de conteúdo e conexão à rede. Sem esse nó, a rede não funciona. O nó central é o ponto alto da hierarquia do sistema. Em redes p2p, esse nó central pode ser responsável pela distribuição de identificadores únicos, distribuição de chaves públicas, cálculo de reputação de nós (explicado no capítulo 4), entre outros serviços que ajudam a tornar a rede segura.

Entretanto, um nó central não pode assegurar algumas questões, como garantir a inexistência de conteúdo malicioso oculto dentro de informações supostamente confiáveis. Como as informações não trafegam pelo servidor, ele não é capaz de garantir a qualidade dos dados. Mecanismos como reputação dos nós pode ajudar nesse ponto, mas mesmo assim não resolve completamente o problema, uma vez que, se um nó é confiável para a maioria dos outros, não significa necessariamente que esse nó irá se comportar bem com todos os outros que não o conhecem.

Mas, apesar da maior segurança em redes p2p centralizadas, o fator que mais dificulta o crescimento dessa arquitetura é sua escalabilidade, já que a rede fica restrita à capacidade do nó central. Além disso, a rede fica totalmente dependente da operabilidade desse nó (ponto único de falha). Mesmo que nós auxiliares do nó central sejam implantados, a rede ainda terá problemas sérios de escalabilidade, já que a capacidade de provimento de recursos de um nó é limitada. Com vários nós auxiliares a rede se descaracteriza e deixa de ser considerada uma rede centralizada.

3.3.2 Redes p2p Descentralizadas

Redes p2p descentralizadas não devem possuir nenhum nó com qualquer diferença de funcionalidade dos demais, ou seja, cada nó é igual ao outro do ponto de vista de suas obrigações com o sistema. Esse fato ajuda na escalabilidade do sistema, que não necessita de pontos realizando funções especiais como controle de acesso ou busca de recursos. Outro ponto positivo é a tolerância a faltas que esse modelo possui, já que nenhum nó é essencial ao sistema. Pode-se notar nesse esquema que não há

qualquer forma de hierarquia entre os nós, já que um modelo hierárquico pressupõe controle dos nós inferiores pelos nós superiores na hierarquia.

Modelos hierárquicos normalmente possuem facilidades de segurança. Em modelos hierárquicos os pontos superiores realizam certas funcionalidades de segurança responsáveis por garantir a segurança de toda a rede. Além disso, os nós superiores podem monitorar as ações dos nós inferiores para que esses pontos inferiores se comportem de maneira aceitável dentro do sistema. As redes p2p descentralizadas têm sua segurança muito mais complexa que redes hierárquicas, já que, em redes p2p descentralizadas nenhum nó pode ser capaz de garantir a segurança de toda a rede, já que, caso isso ocorra, a rede p2p descentralizada se descaracteriza.

Podem-se destacar dois modelos de segurança para redes p2p descentralizadas. No primeiro, a rede como um todo tenta se manter segura com a ajuda de todos os nós. Esse é um bom modelo, desde que a rede consiga minimizar ações de nós que porventura tentem corromper ou desvirtuar esse trabalho, ou seja, esse modelo normalmente é vulnerável a ataques internos. A rede, para que o modelo de segurança funcione, deve ser capaz de identificar e ignorar nós internos maliciosos. Já contra ataques externos à rede, esse modelo funciona bem, já que ataques à segurança de um único nó na rede deve notificar toda a rede contra a ameaça.

Outro modelo para segurança de redes p2p descentralizadas é quando cada nó tem sua própria política de segurança. Um bom exemplo disso é o trabalho (Kamvar et al., 2003). Nesse trabalho, cada nó define o quanto pode e deseja confiar nos outros nós. Esse nó só será afetado por um problema de segurança se confiar em um nó malicioso. O problema desse modelo com relação ao anterior é que ele é menos eficiente contra ataques externos, embora seja bem mais eficiente contra ataques internos.

3.3.3 Redes p2p com Supernós

Supernós podem ser uma solução ao problema da escalabilidade das redes p2p centralizadas. Se forem usados diversos supernós em pontos estratégicos, uma rede p2p

pode crescer sem maiores problemas de escalabilidade. Apesar de esse ser um ponto positivo, existem fatores que comprometem esse modelo.

Um fator importante com relação ao uso de supernós é o controle desse uso. Na rede KaZaA, por exemplo, qualquer nó pode se tornar um supernó. Essa liberdade dada aos pontos do sistema tem diversas conseqüências, tanto positivas quanto negativas.

Um ponto positivo é que os nós se tornam mais autônomos para decidir sobre suas obrigações com a rede. Outra vantagem é um ganho de desempenho no sentido que se existe controle, necessariamente deve existir também uma troca de mensagens para que o controle funcione. Já um ponto negativo é que alguns nós pouco capazes podem encarregar-se de obrigações e diminuir o desempenho da rede, pois nós com pouco poder de processamento não conseguiriam atender às obrigações em um tempo aceitável.

Quando existe um controle de acesso ao grupo de supernós, uma série de ações deve ser tomada. Uma delas é estabelecer as condições que são submetidas aos nós para se tornar um supernó. Isso demanda troca de mensagens entre os nós, conseqüentemente aumentando o tráfego da rede. Outro ponto que pode ser levado em conta é a localização geográfica dos nós. Se o controle levar em conta esse fator, já estará em vantagem para o modelo anterior, pois se os nós se tornam supernós por uma simples autoproclamação, podem ocorrer regiões da rede que estejam superlotadas de supernós e outras sem nenhum supernó ao redor.

3.3.4 Busca Centralizada em Redes p2p

A busca centralizada caracteriza-se pela existência de um nó responsável pelas buscas de todo o sistema. Quando algum nó da rede deseja procurar por um recurso ele encaminha um pedido de busca ao nó central e este pesquisa se o recurso existe e onde se encontra o recurso. O modelo do Napster segue esse método de busca.

É importante diferenciar esse modelo de busca com a arquitetura centralizada. Na arquitetura centralizada o nó central não é responsável apenas por realizar as buscas do

sistema. Por exemplo, em uma rede p2p centralizada o nó central presente pode realizar o controle de acesso ao sistema e a rede pode realizar uma busca desestruturada. Nesse exemplo, a arquitetura é centralizada, mas a busca não.

3.3.5 Busca Desestruturada em Redes p2p

Podem-se ter dois tipos de busca de conteúdo em redes p2p: busca estruturada e desestruturada. A busca desestruturada se caracterizada por, quando algum ponto do sistema deseja procurar algum dado na rede, o nó fazer uma pergunta a seus nós vizinhos. Se algum deles tiver o dado, responde ao requisitante. Se não tiverem, os nós repassam a requisição a outros nós, sendo que esse pedido tem um valor finito de nós que deve percorrer. Esse valor finito serve para não carregar indefinidamente pedidos pela rede. Como uma rede p2p descentralizada normalmente não é capaz de saber seu tamanho e organização estrutural, esse valor finito é necessário. Porém, isto acarreta numa perda de funcionalidade, pois dois nós podem estar separados por um valor muito alto de saltos que se tornam inalcançáveis um ao outro.

As questões de segurança nesse tipo de busca são menores que a busca estruturada. Um problema grave, que pode ocorrer em ambos os casos, é no caso de um nó responder a qualquer pergunta afirmativamente. Pode-se citar, por exemplo, o caso do *worm* Mandragore (Gupta, 2003) que atacou a rede Gnutella. Esse *worm* respondia a qualquer pergunta que encontrasse afirmativamente e passava seu *link* como resposta. Se alguém o copiasse, ele alterava o funcionamento da aplicação para que pudesse responder afirmativamente a todas as perguntas no novo nó infectado.

3.3.6 Busca Estruturada em Redes p2p

O outro tipo de busca, a busca estruturada, é feito pelo uso de uma tabela *hash* distribuída. Essa tabela *hash* é dividida pela rede de forma que, cada nó, dependendo de sua colocação no sistema e de seu identificador, fica responsável por uma parte dessa tabela. Quando um determinado recurso deve ser colocado na tabela, o nó possuidor do recurso deve notificar o nó responsável por armazenar a posição em que esse recurso

deve ser indexado. Se algum outro nó deseja requisitar o recurso, ele faz uma pesquisa na DHT para saber quem possui o recurso. Como o algoritmo de busca na DHT é conhecido por todos os nós, é mais rápido para o requisitante encontrar o nó que armazena a porção da DHT na qual se deve encontrar as informações do possuidor do recurso, já que a busca é direcionada (Sit e Morris, 2002). Assim, o nó requisitante descobre onde está localizada a informação que deseja dentro da DHT.

Este modelo tem a vantagem de proporcionar uma busca direcionada aos recursos de uma rede p2p. Apesar disso, deve haver uma colaboração total de cada nó da rede para o bom funcionamento do sistema. Se, por exemplo, um nó se recusa a responder as requisições sobre sua parte cabível da DHT, os recursos nela listados estarão inalcançáveis mesmo para os nós vizinhos ao detentor da informação. Portanto, esse modelo deve contar ou com a boa fé dos usuários ou com mecanismos que detectem o comportamento irresponsável de usuários maliciosos, como afirma (Sit e Morris, 2002).

O uso de DHT pode ser também encontrado em redes p2p desestruturadas com supernós. Os supernós seriam responsáveis pela DHT, enquanto que os outros nós apenas se encarregam de enviar as informações à DHT e a utilizarem para buscas. Embora o uso de supernós diminua a quantidade de nós em que a DHT está presente, isso não acarreta na mudança da política de segurança da DHT.

3.4 Metadados

Numa fase de busca em uma rede colaborativa p2p, os usuários devem informar algumas características do recurso que desejam para tentar encontrá-los dentro do sistema. Essas informações enviadas aos outros pontos da rede são chamadas metadados. Os metadados são informações que referenciam um tipo de dado específico que se deseja encontrar.

(Oram, 2001) compara os metadados com um conteúdo de um catálogo ou um guia de TV. Os metadados são rótulos, como “título”, “autor”, “tipo” ou “idioma” usados para descrever um livro, pessoa ou programa de TV. Um metadado pode ser considerado um dado informativo sobre outro dado.

Um exemplo mais simples de metadado é o tipo de um arquivo que se deseja encontrar. Nas aplicações p2p mais usadas, como o KaZaA ou o Emule (Emule, 2004), é permitido a um usuário baixar desde arquivos de música até arquivos executáveis. Um metadado nesse caso seria a busca por arquivos-texto. Então o nó que deseja requisitar um arquivo-texto deve informar na sua busca que deseja arquivos do tipo "*.txt".

O uso de metadados tem um papel fundamental no funcionamento das redes p2p. Sem os metadados, os usuários seriam obrigados a baixar todo o conteúdo de um determinado recurso para depois descobrirem do que se trata o conteúdo. Além disso, o metadado serve para filtrar as buscas por conteúdo. Sem os metadados, uma busca por recursos no sistema deveria varrer todo o conteúdo da rede de nó por nó para se encontrar o que se deseja.

Existem dois tipos de metadados, os explícitos e os implícitos. Os explícitos são os metadados informados dentro de uma busca por recursos na rede. Pode-se incluir dentro desse tipo o tamanho mínimo e máximo do arquivo que se procura, o nome do arquivo ou a largura de banda do nó que possui o recurso. Os metadados implícitos são os metadados que não são necessários informar nas buscas. No caso do Napster, por exemplo, não precisa informar numa busca que se deseja um arquivo de música, já que todo o conteúdo da rede é de músicas do formato MP3.

Existem metadados que podem não ser requisitados, mas devem ser informados em uma busca por recursos em redes p2p. Um exemplo é o valor *hash* de um arquivo. O valor *hash* de um arquivo é o resultado do conteúdo do arquivo aplicado a uma função *hash*, como a SHA1 ou MD5. A finalidade de se obter um valor *hash* de um arquivo é reconhecer dois arquivos de fontes diferentes como sendo iguais (garantir a integridade do arquivo). Se os valores *hash* de dois arquivos com nomes quase iguais, como por exemplo "javaprog.doc" e "javaprg.doc", forem iguais, os arquivos devem ser iguais.

Os valores *hash* podem ser usados como identificadores únicos de arquivos. Embora dois arquivos diferentes possam gerar o mesmo valor *hash*, essa probabilidade é pequena (Stallings, 2003). A vantagem disso é que um usuário pode renomear um arquivo que acaba de receber e esse arquivo continua sendo uma cópia de arquivo original. Já quando se trata de redes p2p seguras, outros tipos de metadados podem ser

requisitados. Esses metadados normalmente estão relacionados com certificados digitais, valores de reputação, micropagamentos ou assinatura de conteúdo.

3.5 Gnutella

Embora o Napster tenha sido a primeira aplicação peer-to-peer colaborativa a surgir em grande escala na internet, o Napster possui algumas características que o distanciam do modelo de rede p2p puro e totalmente descentralizado. A primeira tecnologia p2p totalmente descentralizada que surgiu comercialmente em grande escala na internet é o sistema Gnutella. O Gnutella é um protocolo de distribuição de conteúdo descentralizado usado por diversas aplicações. Tais aplicações, por usarem o mesmo protocolo, podem trocar conteúdo entre si sem problemas de compatibilidade.

Segundo (Oram, 2001), a gnutella está entre as primeiras tecnologias descentralizadas que devem reformular tanto a internet quanto a forma de pensar das pessoas sobre aplicações em rede. Essa forma de pensar é a reação tradicional instintiva de se criar um sistema hierárquico cliente/servidor para qualquer tipo de aplicação.

A rede Gnutella é a prova de que tecnologias descentralizadas em larga escala são viáveis. Segundo (Kan, 2001), a rede gnutella é, de maneira subjetiva, uma festa entre amigos na internet, daquelas em que cada um deve levar um prato surpresa. Mas, ao invés de bolos ou sanduíches, arquivos MP3 e MPEG.

Seguindo as classificações de arquiteturas apresentadas neste capítulo, a rede gnutella encaixa-se totalmente no perfil de uma rede descentralizada com busca não-estruturada. Diferentemente do Napster, a rede Gnutella funciona sem um servidor central para controle de acesso e busca de conteúdo.

3.5.1 Gnutella X Napster

Existem diversas diferenças no modelo do sistema Gnutella e no modelo da aplicação Napster. Basicamente, a rede Gnutella é um exemplo de uma rede p2p pura e

a rede Napster uma rede centralizada. Apesar da funcionalidade básica das duas ser a mesma - a troca de arquivos via TCP/IP - as diferenças vão desde a conexão com a rede até a busca de conteúdo.

A primeira diferença é como ocorre a conexão de um novo nó na rede. Na rede Gnutella, quando um nó deseja ingressar na rede ele envia uma mensagem de PING a um nó participante da rede que esteja mais próximo a ele ou que ele conheça. Depois, esse nó que recebe a mensagem PING retransmite a uma vizinhança próxima e estes respondem à mensagem com uma mensagem PONG. Desta forma, o novo nó informa a alguns nós que deseja participar da rede e adquire conhecimento dos seus nós vizinhos. Um ponto interessante é que no modelo do Gnutella os nós não precisam informar os dados que compartilham para ninguém no momento da conexão ao sistema.

No caso do Napster, quando um nó deseja ingressar no sistema, ele deve contatar o nó central da rede. Esse nó central requisita algumas informações sobre o conteúdo que o nó deseja compartilhar na rede. Pode-se notar que no modelo do Napster, a indexação de conteúdo é feita já no ingresso do nó na rede. Entretanto, como todos os nós do sistema estão conectados diretamente no nó central não existem grandes distâncias entre dois pontos quaisquer.

Outra diferença está na busca por conteúdo. O protocolo Gnutella realiza uma busca descentralizada não-estruturada, enquanto que a aplicação Napster realiza uma busca centralizada. Na busca descentralizada não-estruturada da rede Gnutella, quando um usuário deseja obter um conteúdo sobre determinado assunto, ele faz uma requisição aos seus vizinhos perguntando se estes possuem algum que combine com as definições da requisição. Estes vizinhos podem ou responder afirmativamente a requisição e/ou repassá-la a outros nós. Desta forma, uma requisição pode perambular por grande parte do sistema até encontrar o conteúdo desejado.

No modelo centralizado do Napster, toda requisição deve ser feita automaticamente para o nó central. O nó central possui uma base contendo informações sobre o conteúdo disponibilizado por todos os usuários. O nó central então busca nessa base e retorna ao usuário as informações que possui e se enquadram nas definições.

3.5.2 Metadados Gnutella

As buscas de conteúdo na rede Gnutella podem variar de acordo com a aplicação que cada usuário do sistema usa. Isto porque cada usuário tem a liberdade de interpretar os dados que recebe de uma requisição da forma que quiser. Por exemplo, quando uma aplicação recebe uma consulta contendo o metadado "listas encadeadas", pode: interpretar a conjunção ou a disjunção das palavras, retornar somente arquivos com esse nome de arquivo, retornar qualquer arquivo-texto que contenha as palavras, ou seja, a interpretação do metadado fica a cargo do nó que o recebe.

Essa liberdade é necessária pelo fato dos nós possuírem diferentes conteúdos e trabalharem em diferentes áreas de conhecimento. A palavra "carro", por exemplo, usada como metadado, retorna conteúdos diferentes se enviadas a um catálogo de brinquedos e a um *site* de venda de carros. Embora o conteúdo retornado pelos dois seja diferente, nenhum dos dois está errado, pois cada um trata o metadado de acordo com a base de dados que possui, e nenhum deles pode ter certeza sobre qual o conteúdo exato desejado pelo requisitante.

3.5.3 Anonimato

O protocolo Gnutella é um protocolo independente das camadas inferiores no modelo TCP/IP. Isto significa que a rede Gnutella pode ser considerada uma rede *overlay*. Redes *overlays* são sistemas que funcionam sobre uma outra rede existente. Portanto, os usuários da rede Gnutella não são associados diretamente aos seus endereços IP.

Essa dissociação é importante porque muitos usuários usam, por exemplo, acesso à internet via *dial-up*, ou acesso discado. O acesso discado gera um endereço IP para cada conexão na rede. Como os usuários não passam o tempo todo conectado à internet em computadores *dial-up*, cada conexão gera um endereço IP diferente. Portanto os usuários não devem ser associados a um valor de IP fixo. Outras aplicações peer-to-peer mais conhecidas na atualidade, como ICQ, KaZaA e Napster, também ignoram o modelo DNS, vigente na internet.

Essa liberdade gera um pseudo-anonimato na rede Gnutella. Embora numa conexão entre dois pontos da rede os nós tomem conhecimento sobre seus endereços IP atuais, esse valor de IP não é fixo, portanto não possui grande valor. Esse ponto é um grande problema de segurança, pois usuários maliciosos podem trocar de identificador a todo instante sem se preocupar com seus endereços IP.

A propriedade de anonimidade é comum na maioria das aplicações p2p. A maioria das aplicações não faz referência ao endereço IP do usuário. Entretanto, alguns sistemas mais protegidos restringem essa propriedade utilizando maior controle de acesso ao sistema e associando mais fortemente o usuário a um endereço IP. A falta de anonimidade não significa necessariamente que a rede deixa de ser p2p, apesar da importância dessa propriedade nesse tipo de sistema.

3.5.4 Arquitetura

O protocolo Gnutella é composto de cinco tipos de mensagens. (Clip2, 2004) apresenta a versão 0.4 do protocolo Gnutella com as mensagens listadas na tabela 3.1. Um nó que deseje se conectar na rede Gnutella deve estabelecer uma conexão com outro nó presente no sistema. A aquisição da informação sobre o endereço desse nó presente no sistema não faz parte da arquitetura, entretanto alguns servidores mantêm disponível para consulta listas de nós que normalmente estão presentes na rede.

Depois que o nó que deseja ingressar no sistema obteve as informações de conexão de um nó presente no sistema, ele deve enviar a seguinte mensagem para realizar uma conexão:

GNUTELLA CONNECT/<protocol version string>/n/n

No caso do Gnutella versão 0.4 descrita em (Clip2, 2004), a <protocol version string> é a string “0.4”, ou “\x30\x2e\x34”.

Tabela 3.1: Tipo de mensagens Gnutella

Descritor	**Descrição**
Ping	Usado para descobrir nós na rede.
Pong	Resposta para uma mensagem Ping.
Query	O mecanismo de consulta pela rede.
QueryHit	A resposta de uma consulta Query.
Push	Mecanismo usado para conectar nós protegidos por *firewalls*.

O *servent* que recebe um pedido de conexão deve responder com a mensagem:

GNUTELLA OK/n/n

Entretanto, caso o *servent* não seja capaz de aceitar o pedido de conexão ele pode re-encaminhar o pedido de conexão ou simplesmente descartá-lo.

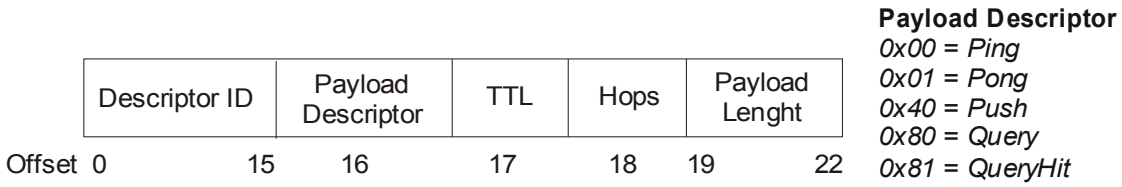

Figura 3.2: Cabeçalhos de mensagens do Gnutella.

Os cabeçalhos de mensagens do protocolo seguem como na Figura 3.2. Cada cabeçalho possui 5 campos. O mais importante é o *Payload Descriptor*, que informa o tipo de mensagem a ser enviada. O primeiro campo é o *Descriptor ID*, que identifica seqüencialmente as mensagens. O campo *TTL* é usado para indicar a quantidade de saltos que uma mensagem deve ainda permanecer na rede. O campo *Hops* indica quantos saltos a mensagem já caminhou. Finalmente, o campo *Payload Lenght* indica o tamanho da carga útil.

3.6 Controle de Acesso

Um ponto importante ainda não mencionado é o controle de acesso a um sistema peer-to-peer. A maioria das aplicações p2p atual não faz qualquer restrição aos nós que desejam ingressar no sistema. Mas, para que redes reforcem seus mecanismos de segurança, seria necessário um controle mais rigoroso ao sistema, algo que pudesse banir usuários maliciosos da rede.

Esse tipo de controle é de difícil implementação pelo fato da grande parte dos usuários não usar endereços IP fixos. Entretanto, é possível criar um outro tipo de controle, feito localmente por um usuário, como mostra o trabalho de (Righi, et al., 2004a), o qual define através de um modelo RBAC os níveis de permissões para os recursos de um nó em uma rede p2p. Nesse caso, um usuário deve provar que tem boas intenções para ser aceito como cliente do *servent*. Embora esse controle não afete todo o sistema, o sistema garante que usuários maliciosos não prejudiquem outros usuários sem que estes tomem conhecimento.

3.7 Redes p2p e Direitos Autorais

As redes peer-to-peer, pelo fato de serem descentralizadas e possuírem uma anonimidade relativa, são capazes de distribuir com sucesso materiais privados, protegidos ou proibidos. O Napster, a primeira aplicação p2p a atingir larga-escala, só obteve sucesso porque distribuía músicas no formato MP3 de graça. Outras redes p2p, como a Gnutella, seguem a mesma direção.

Por causa disso, muitas pessoas associam as redes p2p com o fim da censura e do direito autoral. O Napster enfrentou uma batalha judicial em 2001 por causa do mau uso de seu sistema reclamado pelas gravadoras de discos. Como os usuários do Napster eram capazes de obter qualquer música que desejassem, deixavam de comprar CDs. Depois de perder essa batalha judicial o Napster teve que se tornar um sistema pago, para restringir esse processo de distribuição de conteúdo protegido.

Já no caso da Gnutella, pelo fato de ser mais um protocolo que uma aplicação e ser um sistema totalmente descentralizado, o sistema consegue sobreviver apesar da crítica por parte da indústria fonográfica. É impossível definir um responsável pelo funcionamento atual do gnutella e muito menos tentar parar a rede. Para parar o Gnutella é preciso parar a internet.

Entretanto, existem tentativas de tentar inibir a distribuição de conteúdo ilegal pelo sistema. Um exemplo é a aplicação Wall of Shame (Oram, 2001) desenvolvida para o protocolo Gnutella. Essa aplicação, um "cavalo de tróia", foi a primeira tentativa de identificar usuários e traficantes de pornografia infantil pela rede. Essa aplicação, funcionando sobre a rede Gnutella, recebia todas as requisições que tratassem do assunto e respondia com um falso conteúdo. Caso esse conteúdo fosse requisitado, o endereço IP do requisitante era colocado em uma página Web. Assim, quem trafegasse pornografia infantil pelo Gnutella acabava aparecendo na página se requisitasse algo da aplicação Wall of Shame.

Apesar da maior parte das pessoas associarem redes p2p com essa funcionalidade, as redes p2p não servem apenas para troca de conteúdos digitais. Um bom exemplo é o projeto SETI@Home. O SETI@Home é um projeto para busca de vida extraterrestre usando uma rede p2p centralizada. Nesse projeto, os computadores pessoais realizam cálculos matemáticos indicados pelo servidor central e retornam a ele ou a outros pontos o resultado dos cálculos que esses computadores realizaram. O servidor central é quem coordena os cálculos a serem realizados.

3.8 Conclusão

Este capítulo apresentou os conceitos básicos em torno de sistemas peer-to-peer. Peer-to-peer é um termo novo que surgiu no mercado antes mesmo que pesquisas em torno do tema fossem realizadas, diferente do que ocorre normalmente na ciência. A tecnologia peer-to-peer alterou drasticamente o modelo da internet vigente na época que surgiu, por volta do final da década de 90.

As redes p2p podem ser classificadas de acordo com a estrutura da rede ou de acordo com o conteúdo das informações trafegadas. Como o conteúdo trafegado não é ponto preponderante neste trabalho, foi detalhada a classificação segundo a estrutura da rede. Essa classificação varia de acordo com dois pontos: a centralização ou não da rede e com relação a busca de conteúdo pelo sistema.

Outro ponto importante no conceito de redes p2p é o metadado usado nas buscas de conteúdo. Os metadados expressam de alguma forma o conteúdo que um usuário do sistema deseja encontrar. Da mesma forma, o resultado da busca expressa a interpretação de um nó que recebeu uma requisição de busca. Essa interpretação varia de acordo com a exatidão do metadado e de acordo com a base de conhecimento do nó que recebe o metadado.

O protocolo Gnutella é o melhor exemplo de rede p2p pura. Ele reúne todas as características presentes em redes p2p puras, além disso, foi o primeiro protocolo p2p puro a surgir. Sua arquitetura possui diversos pontos que diferem da aplicação Napster. A principal diferença é a centralização do Napster, o qual (Rocha et al., 2004) chama de um “ponto de falha” na rede p2p, pois o nó central possui funcionalidades que os outros pontos do sistema não têm. Devido a essa centralização, a escalabilidade de uma rede p2p fica comprometida.

Uma propriedade comum em redes p2p é a anonimidade. Essa propriedade garante uma independência da aplicação p2p dos endereços IP presentes nas máquinas. Essa propriedade é necessária principalmente pelo fato de que muitos usuários dessas redes usam conexões *dial-up* para acessar a internet, e nesse tipo de conexão o endereço IP é temporário. Embora necessária, essa propriedade traz alguns problemas de segurança em sistemas p2p, pois com identificadores temporários os sistemas encontram dificuldades para localizar usuários mal-intencionados ou atacantes.

Normalmente, o uso de redes p2p é associado com a quebra de direitos autorais e o fim da propriedade intelectual. Embora o uso de redes p2p tenha contribuído para esse fato, as redes p2p possuem outras funcionalidades que são válidas para pesquisa. Um bom exemplo de um bom uso de redes p2p é o projeto SETI@Home.

Capítulo 4 Trabalhos Relacionados

4.1 Introdução

As redes peer-to-peer são redes de computadores nas quais seus participantes são *servents*, isto é, têm funções de servidor e cliente ao mesmo tempo. Para *servents* trocarem informações e recursos, são necessários alguns mecanismos de segurança para garantir a funcionalidade e proteção do sistema contra o mau uso por parte de participantes ou entidades externas. Dois pontos importantes nas redes p2p são a segurança na troca de recursos e o estabelecimento de níveis de confiança entre os *servents*.

Existem diversos trabalhos que sugerem soluções para essas questões em redes p2p. Esses trabalhos tentam resolver alguns problemas decorrentes desse tipo de arquitetura, onde todos os participantes têm funcionalidades basicamente iguais e não existe no sistema um nó central, ou seja, uma entidade raiz responsável pelos demais nós que possa garantir o bom uso da rede.

Este capítulo busca identificar os artigos atuais na área em questão, descrever os pontos principais de cada um, fazer uma análise crítica e identificar problemas existentes nesses trabalhos. A partir desse estudo é possível descrever uma proposta que minimize os problemas encontrados em tais artigos.

O capítulo está dividido em seções. Na seção 2 são apresentados os trabalhos sobre redes p2p baseadas em micropagamento. A seção apresenta três trabalhos sobre redes p2p baseadas em reputação, exaltando as peculiaridades de cada trabalho. Já a seção 4 mostra alguns outros artigos referentes à segurança e redes p2p. Na seção 5 são apresentados os problemas encontrados a respeito do tema e sugere algumas soluções propostas. Por último, na seção 6, uma breve conclusão é apresentada.

4.2 Redes p2p baseadas em Micropagamento

Dois tipos básicos de confiança podem ser estabelecidos em redes p2p. O primeiro deles, e menos usado, é o *micropayment,* ou micropagamento. Nesse tipo de confiança, os participantes têm que oferecer uma garantia de sua confiabilidade ao sistema. Devido a esse fato o modelo restringe muito o seu uso. Um trabalho sobre micropagamento é (Yang e Garcia-Molina, 2003), o qual sugere o uso de um sistema de pagamento para o uso de uma rede p2p, como o exemplo do Napster, que, de uma rede p2p livre passou a cobrar pelos seus serviços.

4.2.1 PPay

No trabalho de (Yang e Garcia-Molina, 2003), chamado de PPay, usuários que queiram participar da rede, devem obter uma quantidade de "moeda digital" de um usuário participante que tenha esse serviço, pagando com dinheiro real por essa moeda. Com essa "moeda digital" o usuário pode pagar pelo uso dos recursos da rede p2p, assim como vender seus recursos disponíveis. Embora a segurança desse modelo tente garantir que violações sejam não-rentáveis, o uso do modelo é restrito a sistemas que exijam ou suportem essa idéia de compra de participação.

Para deixar mais claro como uma violação pode se tornar não-rentável, pode-se imaginar que para cada conexão o usuário deve pagar R$ 0,01. Para quem deseja obter uma música, 1 centavo é muito pouco e esse custo se torna irrisório. Porém, se algum usuário mal-intencionado resolver causar um ataque *DoS* ele deve pagar 1 centavo para cada conexão que realize e, desta forma, o usuário acaba pagando muito pelas diversas conexões que tem que abrir para realizar o ataque.

4.2.2 Escambo

Um outro tipo de mecanismo de micropagamento é o caso do protocolo Escambo (Righi et al., 2004b). Nesse trabalho, os usuários em vez de pagar por recursos, devem

oferecer seus recursos para obter novos recursos. Esse protocolo também minimiza o problema dos nós caronas. Segundo (Napster, 2004), o Napster já chegou a possuir cerca de 80% de seus usuários como nós caronas. Um nó carona é um nó que apenas requisita recursos, mas não oferece nenhum ao sistema. No Escambo, como os nós devem oferecer recursos para conseguir recursos, os nós caronas não conseguem requisitar recursos.

No sistema Escambo, cada usuário deve estabelecer uma política de troca de recursos, obedecendo a patamares máximos e mínimos. Por exemplo, um usuário pode definir que para alguém acessar seus recursos deve oferecer no mínimo 10Mb de dados para a rede. Então, somente usuários que têm essa quantidade de recursos para oferecer estão aptos a acessar os recursos do nó que definiu essa política.

4.3 Redes p2p baseadas em Reputação

O outro tipo de confiança obtida em redes p2p é o tipo reputação. Vários trabalhos exploram esse método de segurança e todos eles têm em comum o fato de utilizarem experiências anteriores dos nós para atribuir níveis de confiança a recursos ou a outros nós (Kamvar et al., 2003, Singh and Liu, 2003, Damiani et al., 2002, Cornelli et al., 2002, Gupta et al., 2003). Para que isso funcione, os nós precisam ser persistentes no sentido de manterem seus identificadores ou pseudônimos[1] durante todo o período de tempo que estiverem dentro do sistema. Caso um nó deixe a rede e retorne com outro identificador, as informações anteriores sobre esse nó não são utilizadas.

Segundo (Kamvar et al., 2003), existem cinco questões importantes que são tratadas por qualquer sistema p2p baseado em reputação:

1) O sistema deve ser automonitorável, isto é, as éticas compartilhadas pela população de usuários são definidas e sustentadas pelos próprios nós e não por uma autoridade central.

[1] Já que, normalmente, redes p2p devem possuir a propriedade de anonimidade dentro da rede, o uso de pseudônimos é freqüente.

2) O sistema deve manter a anonimidade, isto é, a reputação de um nó deve ser associada com um identificador opaco ao invés de uma identidade externamente associada (como o endereço IP).

3) O sistema não deve designar qualquer lucro a recém-chegados, isto é, uma reputação deve ser obtida com um bom comportamento consistente através de várias transações e não deve ser vantajoso para nós maliciosos com reputações baixas continuamente trocar seus identificadores para obterem status de recém-chegados.

4) O sistema deve manter um *overhead* mínimo em termos de computação, infra-estrutura, armazenamento e complexidade de mensagens.

5) O sistema deve ser robusto a um conjunto de nós maliciosos que conheçam um ao outro e tentem coletivamente subverter o sistema.

Portanto, os trabalhos sobre redes p2p baseadas em reputação devem seguir essas premissas a fim de conseguirem satisfazer as exigências de bom funcionamento da arquitetura. Diversos trabalhos já foram realizados nessa área e os principais cumprem, ao menos em parte, essas questões.

4.3.1 XRep e P2PRep

Os primeiros trabalhos nesse tema são os do grupo italiano *Security Group* da Università di Milano (Unimi, 2004), dentre eles os seguintes: (Damiani et al., 2002), chamado de XRep, e (Cornelli et al., 2002), chamado de P2PRep. A principal diferença entre os dois protocolos é que P2PRep se preocupa em guardar informações apenas sobre os *servents* que fazem parte da rede p2p, enquanto que o XRep armazena, além disso, informações sobre os recursos disponíveis pelo sistema.

A idéia desses trabalhos começa com os nós guardando informações sobre transações feitas por eles em uma base de reputação. Quando algum nó deseja receber um recurso desconhecido oferecido por um nó também desconhecido, ele requisita a opinião de alguns nós para verificar a reputação do detentor do recurso, além da

reputação que o recurso desejado possui entre eles (somente no caso do XRep). Se o nível de reputação de, tanto o recurso quanto o nó que o oferece forem satisfatórios, a operação é realizada. Caso contrário, o nó pode cancelar o *download*. Esses trabalhos possuem mecanismos de segurança eficientes, porém trazem algumas limitações, como a baixa utilização de nós e recursos com pouca reputação.

Um exemplo é de um *servent* que entra no sistema após um bom tempo de funcionamento da rede. Esse nó possui apenas recursos comuns a vários outros nós e não possui nenhuma reputação de início. É bem provável que existam outros nós com os mesmos recursos que os seus e possuam melhor reputação no sistema. Como seus recursos são comuns a outros nós, dificilmente um nó que precise desses recursos vai recorrer a esse novo nó, já que ele não possui boa reputação. Portanto, esse nó dificilmente vai aumentar seu valor de reputação e assim seus recursos praticamente não serão utilizados, enquanto que podem existir nós que estejam com problemas sérios de tráfego, já que possuem boa reputação e são muito requisitados.

4.3.2 TrustMe

Outro trabalho importante é o de (Singh e Liu, 2003). Esse artigo também descreve um modelo de segurança em redes p2p baseado em reputação. Nesse modelo, chamado de TrustMe, quando um nó entra na rede, o nó *bootstrap*[2] seleciona um conjunto de nós (chamados de THA[3] *peers*) para atuarem como controladores da reputação desse novo nó. Embora seja definido que o modelo necessite que os THA *peers* sejam um conjunto com diversos nós, o trabalho não especifica como deve ser feito esse gerenciamento dos vários THA *peers*.

Quando um nó requisita e utiliza-se de um recurso de um outro nó, ele deve avaliar a atuação desse nó que ofereceu o recurso. Ele deve enviar um relatório de comportamento aos THA *peers* desse nó. Se o nó que oferece o recurso se comportou bem, provavelmente ele receberá uma boa avaliação e seu valor de reputação aumentará.

[2] Um nó *bootstrap* é o nó responsável pela conexão com um nó recém-chegado ao sistema.

[3] *Trust Holding Agents.*

4.3.3 CORC e DCRC

Um outro trabalho relacionado com o tema abordado neste documento é o artigo (Gupta et al., 2003). Este define dois tipos de cálculo de reputação chamados de CORC (Credit Only Reputation Control) e DCRC (Debit-Credit Reputation Control). Nos dois exemplos, um ou mais nós são chamados de nós RCA (Reputation Control Agent) e são responsáveis pelo cálculo da reputação de um outro nó qualquer. Todas as transações devem, envolvendo esse nó, ser reportadas ao nó RCA pertinente a ele.

A diferença dos modelos é que o CORC calcula somente transações válidas para cálculo da reputação, ou seja, o valor da reputação somente cresce. No modelo DCRC, transações ruins podem piorar a reputação do nó no sistema. A vantagem desse trabalho para o de (Singh and Liu, 2003) é que o valor de reputação é mantido pelo próprio nó e, caso o nó RCA saia da rede sem aviso, o valor de reputação ainda é guardado e pode ser delegado outro nó para atuar como RCA.

4.3.4 EigenTrust

Todos os trabalhos citados até o momento têm algo em comum: quando um nó deseja saber a reputação de um outro nó desconhecido, esse nó pergunta aos seus nós vizinhos sobre a reputação de tal nó. A resposta tem relação com a opinião dos nós que trocaram recursos com o nó desconhecido, e nada tem com as próprias experiências do nó que requisita a informação.

Contra esse enfoque, existe o trabalho de (Kamvar et al., 2003), que apresenta o protocolo EigenTrust. O protocolo EigenTrust utiliza as notas pessoais de reputação dadas pelo nó requisitante aos nós que ele conhece para calcular a reputação de um nó desconhecido, ou seja, a reputação de um nó desconhecido vai variar de acordo com as experiências anteriores do nó requisitante.

Por exemplo, se o nó *x* pede a opinião de um nó conhecido *y* sobre a reputação de *z*. O nó *x* não conhece *z*, mas conhece *y* que conhece ambos. O valor de reputação que *x* tem sobre *y* é 0.5, isto é, *x* confia em *y* no valor 0.5. O nó *y* responde a *x* que confia em *z*

no valor de 0.8. Então, o nó *x* calcula os valores de confiança e chega a conclusão que seu valor de confiança em *z* é 0.5 x 0.8 = 0.4. É importante notar que a confiança de *y* em *x* pode ser diferente de 0.5 e que *x* pode valer-se de outras opiniões para calcular um valor de reputação para *z*, e este valor de reputação somente é valido para *x*.

4.4 Trabalhos relacionados com segurança e redes p2p

Existem diversos trabalhos que abrangem esses dois tópicos sob um ponto de vista diferente dos citados até aqui neste documento. Um trabalho interessante que pode se encaixar nesse grupo é (Sit e Morris, 2002). O trabalho trata de considerações quando se tenta prover segurança a uma tabela *hash* distribuída (DHT) de uma rede p2p. Redes p2p podem armazenar informações de busca de duas formas: ou os nós somente armazenam suas próprias informações e respondem apenas a consultas destinadas a recursos que eles possuem, ou a rede p2p pode formar uma DHT, e cada nó deve ser responsável por armazenar uma parte dessa tabela *hash* e consultas são feitas a essa tabela, não ao conteúdo dos nós diretamente. Apesar do uso de DHTs para otimizar a busca de conteúdo em redes p2p ser interessante, redes p2p baseadas em reputação normalmente não se utilizam dessa técnica.

Outro trabalho importante é (Vlachos et al., 2004), que desenvolve uma aplicação p2p para gerar a segurança em uma rede de computadores. A idéia é que as máquinas informem umas às outras sobre ocorrências de ataques e vírus para que a rede como um todo aumente ou diminua a segurança em torno dela. O problema neste trabalho é que ele não resolve o problema de um nó malicioso começar a distribuir mensagens falsas para que outros nós diminuam a segurança da rede com o objetivo de futuramente atacar algum nó.

4.5 Problemas Encontrados

Embora alguns dos trabalhos relacionados sejam muito diferentes uns dos outros, podem-se encontrar alguns problemas em comum a todos. Em (Gupta, 2003), são

descritos alguns tipos de ataques freqüentes a redes p2p. Entre esses ataques está listado o ataque *man-in-the-middle*. Nesse tipo de ataque, comum em redes ponto-a-ponto, o atacante se encontra entre um ponto que realiza a comunicação entre dois nós. Esse nó malicioso pode espionar, ou até mesmo forjar mensagens trocadas entre os nós. É um tipo de problema de difícil solução, uma vez que os nós comunicantes normalmente sequer tomam conhecimento da existência de um nó malicioso entre a comunicação.

Outro tipo de problema é o *denial-of-service*. Nesse problema, um ou mais nós maliciosos congestionam a rede com mensagens inúteis para comprometer o funcionamento da rede. Também é um grave problema de difícil solução, pois nem sempre se pode detectar o emissor da sobrecarga de mensagens.

Os dois problemas citados ainda são agravados em redes p2p pela propriedade de anonimidade normalmente existente nesse tipo de rede. Como os nós não devem ser relacionados diretamente com um valor externo, como endereço IP, é muito difícil isolar nós maliciosos. Uma solução encontrada é diminuir ao máximo os benefícios de recém-chegados à rede, para que os nós não troquem de identificadores a cada mau-uso do sistema. Embora essa seja uma boa solução, gera um outro problema. Nós recém-chegados encontram dificuldades durante algum tempo na rede, mesmo que esses nós sejam usuários bem intencionados.

Essa mesma situação gera um outro problema, a criação de gargalos de tráfego de informações por nós com maior reputação. Se os nós são identificados com um valor de reputação, estes preferem utilizar recursos dos nós com maiores reputações, gerando assim um congestionamento sobre esses nós. Embora congestionamento de tráfego não seja um problema de segurança exatamente, esse problema é gerado por um mecanismo de segurança, sendo assim deve ser tratado com a mesma atenção que outros problemas de segurança. Uma solução para esse problema seria gerar valores de reputação diferentes para um mesmo nó, ou seja, um nó *x* pode ter um valor alto de reputação para um nó *y* e um valor baixo para um nó *z*. Nesse caso o nó x receberia tráfego apenas advindo do nó *y*, minimizando esse problema de congestionamento sem interferir na política baseada em reputação de um sistema.

Se os nós possuem reputação diferente para pontos diferentes da rede, como citado no parágrafo anterior, é resolvido outro problema encontrado em redes p2p. Pode existir um caso em que uma grande parte da rede é maliciosa com relação a certo nó, e essa parte pode levar a acreditar na boa fé de outro nó, mesmo que seus nós "amigos" não confiem nessa transação. Por exemplo, se um nó *x* deseja saber a reputação de *y*, e a maior parte do sistema responde ao seu pedido de questionamento de reputação positivamente, o nó *x* deverá confiar em *y*, mesmo que *y* possa fazer parte de um conjunto de nós maliciosos que dominam parte da rede. Contra esse ponto, o trabalho de (Kamvar et al., 2003) é eficaz, já que a reputação do nó *y* dependeria da reputação que *x* possui de quem o informa que *y* é ou não confiável. Se os nós respondem que *y* é confiável a *x*, mas este não confia neles, mesmo que sejam muitos, a reputação de *y* será baixa e *x* se negará a realizar a transação.

O fato das redes p2p possuírem a propriedade da anonimidade também traz um fator benéfico à disseminação de vírus e *worms* pelas redes. Como a maioria dos trabalhos, com exceção de (Damiani et al., 2002), trata somente de reputação de nós, códigos maliciosos não são classificados com baixa reputação, e sim quem os distribui. Isso é um fato grave devido à possibilidade de um usuário malicioso poder constantemente trocar sua identificação para não possuir má-reputação pela distribuição de vírus ou *worms* pelo sistema. Já no trabalho citado que armazena reputação dos recursos, um vírus, depois de algumas transações, ganharia uma baixa reputação e deixaria de ser solicitado pelos usuários do sistema. Embora esse seja um ponto positivo, pode-se causar uma sobrecarga de informações aos nós, que, além de armazenar valores de reputação para os diversos nós da rede, devem armazenar valores de reputação para todos os recursos transitados. O problema da sobrecarga é agravado pelo fato de que os nós podem possuir capacidades de processamento e armazenamento bem diferentes, o que pode prejudicar nós com menores capacidades.

O trabalho de (Vlachos et al., 2004) ajuda-nos a obter uma solução para o problema do tráfego de vírus. O modelo de disseminação de alertas explicado nesse trabalho pode auxiliar o esquema de (Damiani et al., 2002), emitindo alertas para nós atualizarem pró-ativamente suas tabelas de reputação de recursos. Embora a solução pareça ideal num primeiro momento, a solução agrega os problemas encontrados em

(Vlachos et al. 2004), como a indevida sinalização de nós maliciosos da existência ou ausência de recursos maliciosos no sistema. Um mecanismo que pode minimizar esse problema é utilizar a política de confiança de (Kamvar et al., 2003), sendo que essa política consideraria somente alertas de nós confiáveis pelos nós que os recebem.

Quando os nós saem do sistema normalmente não avisam a outros nós sobre essa ação. Poucos trabalhos tratam dessa questão que pode gerar alguns problemas de segurança. Por exemplo, no trabalho de (Singh e Liu, 2003), se existe para um nó qualquer um único nó THA, e este nó deixa o sistema sem qualquer aviso por ocasião de uma falha ou um acidente, ele não realiza as operações de delegação de suas obrigações e o valor de confiança do nó pelo qual ele era responsável é perdido e o nó perde o que conquistou através de um bom-uso contínuo do sistema.

Um problema relacionado com esse caso é o exemplo de nós que saem da rede depois de várias transações e retorna com outro identificador sem avisar outros nós. Outros nós não têm como saber que o nó recém-chegado é o mesmo com quem se relacionaram há algum tempo antes. O problema se agrava pois, em algumas arquiteturas propostas como (Cornelli et al., 2002) e (Kamvar et al., 2003), não existe um tempo definido para abandonar as informações armazenadas pertinentes a identificadores não-presentes na rede. Se esses nós abandonarem os dados no momento em que percebem que o identificador está ativo, correm o risco de o nó apenas ter saído por algum tempo do sistema. Então quando este volta os outros nós apagaram as informações que possuíam sobre ele. Mas se os nós nunca descartarem esse tipo de informação, as bases se tornarão repletas de informações sobre identificadores abandonados.

Redes que usam uma DHT também sobrem de falhas de segurança. Quando algum nó age indevidamente e responde incorretamente as consultas à parte da DHT a qual ele é responsável, os dados indexados nessa porção da tabela podem tornar-se inalcançáveis. O trabalho de (Sit e Morris, 2002) sugere algumas precauções e algumas idéias para contornar esse problema.

4.6 Conclusão

Neste capítulo foram apresentados os trabalhos recentes envolvendo os problemas existentes sobre segurança em redes p2p, com enfoque a trabalhos sobre níveis de confiança em redes p2p. Todos os trabalhos citados apresentam problemas em comum, entretanto alguns problemas são específicos de poucos. Um dos pontos mais polêmicos em relação a redes p2p é a questão de manter a anonimidade com o provimento de segurança. Outro ponto em aberto é como deve ser realizada a política de segurança de uma rede p2p, integrada ou individual. Um terceiro ponto é garantir a segurança do tráfego corrente na rede. Tendo em vista esses aspectos, trabalhos a serem realizados enfocando esses pontos são válidos para pesquisa na área.

Capítulo 5 RBRP: Role-Based Reputation Protocol

5.1 Introdução

Os protocolos de reputação em redes peer-to-peer têm um papel importante no provimento de segurança da rede. Esses protocolos são responsáveis por auxiliar os nós a encontrar a fonte mais confiável de um determinado recurso. Normalmente os protocolos baseiam-se em experiências anteriores para calcular um valor numérico que represente um valor de confiança com relação a um determinado nó possuidor do recurso ou mesmo ao próprio recurso (Pellissari et al., 2004).

Quando a aplicação se encarrega de calcular esse valor numérico e atribuir valores numéricos de confiança de acordo com regras pré-definidas, os protocolos funcionam bem somente se tais regras forem válidas. Porém, fica inviável definir regras pré-definidas globais para atribuição de valores de confiança, já que cada nó de uma rede p2p pode necessitar uma política de segurança diferente.

Para seres humanos é difícil atribuir um valor numérico que represente um valor de confiança com relação a outro ser humano. Seres humanos costumam usar rótulos que indicam confiança como: “amigo”, “colega”, “vizinho” ou “aluno”. No mundo real, quando um ser humano deseja saber se outro é confiável, pergunta a um terceiro conhecido de ambos e recebe uma resposta do tipo “ele é amigo”. Nota-se que esses rótulos reproduzem papéis que outros seres humanos representam. Portanto, os protocolos de reputação que seguissem essa idéia estariam mais próximos da realidade dos seres humanos e teriam melhor aceitação em uma rede peer-to-peer cujos nós representam seres humanos.

Este trabalho define o protocolo Role-Based Reputation Protocol (RBRP), um protocolo baseado em papéis para cálculo de reputação e valores de confiança para nós

de uma rede peer-to-peer. O trabalho apresenta, de maneira detalhada, todo o funcionamento do protocolo, incluindo: a especificação formal, os dados armazenados pelas bases de dados, as mensagens trafegadas, as delegações do usuário e a implementação do protocolo. A principal característica desse protocolo está na forma como reconhece um valor de confiança. Enquanto os demais protocolos pesquisados (Kamvar et al., 2003, Damiani et al., 2002, Cornelli, 2002, Singh e Lui, 2003, Gupta et al., 2003, Righi et al., 2004) procuram quantificar em números valores de confiança, o RBRP atribui nomes relacionados com papéis aos valores de confiança. Isso, além de organizar os valores de confiança em grupos reproduzidos por papéis, auxilia os usuários na atribuição de confiança a outros nós. O RBRP também se preocupa em separar os valores de confiança com relação ao tipo de dado desejado. Isso é importante porque, por exemplo, uma boa fonte de arquivos MP3 não é necessariamente uma boa fonte de arquivos-texto.

Outro detalhe importante com relação ao RBRP e a outros protocolos de reputação é a largura de banda utilizada. Alguns autores se preocupam em comprovar que a carga gerada pelas mensagens desses protocolos não gera um aumento significativo na largura de banda utilizada pelo sistema. Esse trabalho comprova matematicamente que o RBRP é capaz de diminuir a largura de banda utilizada por uma rede p2p, já que com a utilização desse protocolo a quantidade de trocas de recursos inválidas diminui significativamente.

5.2 Seres Humanos e Redes p2p

A aplicação Napster foi, sem dúvida, a principal responsável pela disseminação do conceito de redes p2p pelo mundo de tecnologia da informação. Assim como o Napster, outros sistemas que desempenham a mesma função, a troca de informações ponto-a-ponto, são nomes comumente relacionados com p2p. Aplicações como KaZaA, Gnutella ou Emule são bons exemplos desse fato.

Essas aplicações têm diversos pontos em comum. Um desses pontos é a funcionalidade do sistema: a troca de arquivos pela Internet. Mas o ponto a ser

ressaltado aqui é a presença humana atrás de cada nó do sistema. Em todos esses sistemas, cada nó representa um ser humano buscando recursos na Internet.

Seres humanos possuem diversas diferenças de raciocínio se comparados com máquinas. Essas diferenças alteram o modo como o ser humano encara a confiança. Para máquinas, é simples quantificar em valores numéricos um valor de confiança, mas para seres humanos não. Seres humanos, por outro lado, são mais capazes de associar um valor de confiança a um nome, como “amigo” ou “colega”.

5.3 Nomear Confiança X Quantificar Confiança

Considerando que os nós das principais redes p2 são controlados por seres humanos e estes têm melhor capacidade para associar a confiança a nomes, o uso de nomes de confiança para valores de reputação em redes p2p parece, num primeiro momento, mais correto. Entretanto, é preciso tomar alguns cuidados.

Um ponto em aberto com relação aos nomes relacionados com confiança é o idioma em que são apresentados. Um usuário português pode não entender, por exemplo, os nomes usados por um inglês, e vice-versa. Para solucionar esse problema, podem ser adotadas duas ações. A primeira é regulamentar uma “língua oficial” dentro da rede. Então os nós só podem usar nomes de confiança escritos nessa língua. Outra solução, bem mais complexa, é manter uma base de dados para traduções.

Outro ponto é a forma como os seres humanos encaram os nomes de confiança. É possível que exista uma pessoa que tenha vários amigos, mas não confie muito neles. Outra pessoa pode possuir poucos amigos, mas confiar muito neles. Essas pessoas usam o nome “amigo” para um valor de confiança diferente. Uma solução para isso é acompanhar sempre do nome “amigo” um outro nome que quantificaria a confiança, como “alto” ou “muito alto”. Tais valores podem representar valores numéricos, como, por exemplo, o nome “muito alto” representa valor numérico 10 e valor “baixo” representa valor 3.

5.4 Papéis de Confiança

Os nomes a serem usados em protocolos de reputação não devem ser escolhidos a esmo. Essa escolha deve ser feita de acordo com funções desempenhadas pelos nós no sistema. Embora todos os nós possuam as mesmas funcionalidades na rede, os nós tendem a concentrar suas atividades nas que mais desejam realizar. Por exemplo, em um sistema como o Emule é permitida a troca de arquivos dos tipos: MP3, Vídeo, Executáveis e texto. No Emule, por exemplo, usuários que buscam recursos de leitura certamente vão concentrar suas buscas e seu repositório de dados em arquivos texto e usuários que buscam músicas tendem a possuir mais arquivos MP3 que arquivos-texto.

Essa separação de atividades em redes p2p gera uma separação de funcionalidades realizadas pelos nós. Um determinado nó pode separar, portanto, os outros nós de acordo com as funções que estes representam no sistema. Essas funções podem ser representadas por papéis, como os papéis usados no modelo RBAC. Um exemplo é que um nó de uma rede p2p pode definir, usando o conceito de papéis, um grupo de nós que representam um papel “usuário de arquivos MP3”. Então esse nó atribui maior valor de confiança aos nós que pertencem a esse grupo e representam esse papel.

5.5 Arquitetura RBRP

O protocolo Role-Based Reputation Protocol (RBRP) é um protocolo de reputação baseado em papéis para redes peer-to-peer, sua principal característica. Esse protocolo contribui com alguns pontos significativos no estado-da-arte sobre segurança em redes peer-to-peer.

O ponto principal é distinguir em papéis os nós presentes nas redes, aproximando-se do modelo de controle de acesso RBAC. O segundo ponto é usar nomes ao invés de números para valores de reputação, facilitando a classificação em papéis pelo usuário do sistema. A distinção em papéis auxilia na classificação de reputação dos nós. Nos trabalhos relacionados encontrados, um nó é considerado confiável ou não independente do tipo de recurso que é buscado no sistema. O uso de papéis, portanto, iguala em reputação diversos nós que, por algumas afinidades, se comportam de maneira parecida.

Outra contribuição deste protocolo é, através do uso de papéis, separar os níveis de reputação de um determinado nó da rede de acordo com o tipo de dado desejado numa troca. Nós que usam uma rede p2p para, por exemplo, trocar apenas arquivos-texto serão considerados nós com boa reputação para troca de arquivos-texto somente. Se um usuário buscar a reputação de um destes nós para obter um arquivo texto o valor de reputação não será o mesmo caso opte por um arquivo MP3.

Uma outra vantagem desse protocolo está no uso do certificado de nomes SPKI/SDSI para informar um valor de reputação. Esse uso aproxima o RBRP de um padrão internacional e, além disso, permite que um controle de acesso usando SPKI/SDSI seja anexado ao protocolo sem maiores dificuldades.

Outra característica importante nesse protocolo é atribuir um campo validade ao valor de reputação. Isso é importante porque os nós podem abandonar um identificador, então os dados presentes nas bases de dados relacionados com tal nó também devem expirar, tornando as bases de dados mais atualizadas.

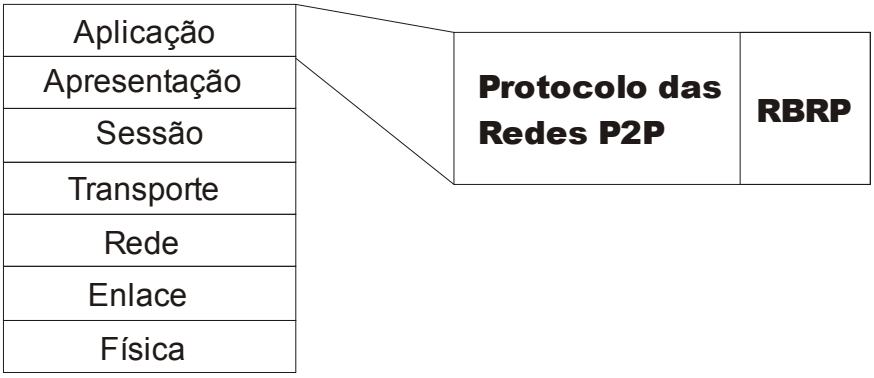

Figura 5.1: Presença do RBRP na pilha de protocolos.

O protocolo RBRP está localizado na pilha de protocolos do modelo referencial OSI como mostra a figura 5.1. Dentro da camada de aplicação, estão localizados os protocolos p2p conhecidos, como o Gnutella ou o Napster. O RBRP é um protocolo localizado juntamente com essa camada, já que pode ser considerado como uma extensão de segurança desses protocolos, porque é dependente de algumas funcionalidades básicas das redes p2p, como a busca por recursos. Entretanto, o RBRP não pode ser considerado uma extensão de um determinado protocolo existente, já que o RBRP é desenvolvido para funcionar juntamente com praticamente qualquer protocolo de redes peer-to-peer.

5.5.1 Bases de dados

Cada nó do protocolo RBRP possui as bases mostradas na Figura 5.2. Pode-se observar nessa figura duas bases. A primeira é a base de nomes que lista os nomes conhecidos pelo nó que a possui. Nessa base existem três campos principais. O primeiro campo, *nome*, lista os nomes – indicando papéis – conhecidos pelo nó. O segundo, *valor de confiança*, mostra quanto o nó confia no papel relacionado. Finalmente, o terceiro campo indica os tipos de dados válidos para a confiança. Inicialmente é proposto separar os dados pelos tipos de dados, mas o protocolo pode ser avançado de modo que suporte outros tipos mais elaborados de distinção de dados.

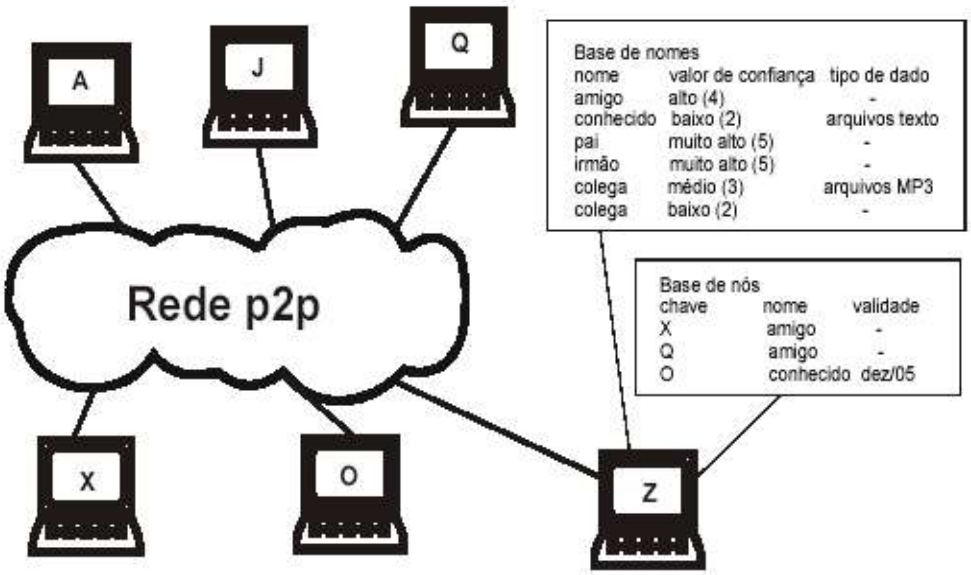

Figura 5.2: Bases dos dados.

A segunda base lista as chaves, que representam outros nós do sistema, relacionadas com os nomes listados na primeira base. Além disso, existe um campo opcional de validade, pois esse relacionamento pode expirar. A partir dessas duas bases os nós são capazes de armazenar suas experiências anteriores e classificar os nós conhecidos por papéis desempenhados no sistema.

5.5.2 Busca por Recursos

O protocolo RBRP não influi, a princípio, no resultado das buscas por conteúdo em uma rede p2p. Essa busca depende exclusivamente do tipo de rede na qual esse protocolo seja utilizado. Se, por exemplo, o RBRP for aplicado à rede Napster, a busca é centralizada. Por outro lado, se for aplicado ao Gnutella a busca é descentralizada e desestruturada.

Apesar da busca não ser alterada pelo RBRP, o protocolo realiza uma ação de classificação dos resultados de acordo com o nível de confiança indicado em suas bases.

Na listagem dos resultados, são mostrados os resultados advindos dos nós que estão presentes na base de nós desse nó requisitante. Assim, a aplicação indica a preferência por usar recursos dos nós com maiores valores de reputação. A Figura 5.3 apresenta um exemplo dessa questão.

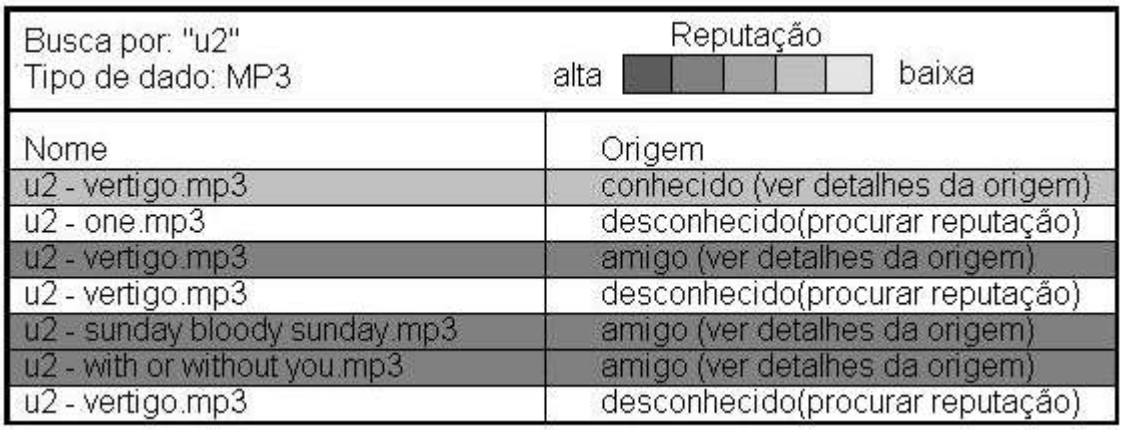

Figura 5.3: Exemplo de resultado de busca com o RBRP

A Figura 5.3 ilustra um exemplo de um resultado de busca numa rede p2p que usa o protocolo RBRP. Pro exemplo, considerando que um determinado usuário resolve buscar uma MP3 da banda “U2”. Ele então faz uma pesquisa por “u2” pelo sistema. O resultado que ele recebe está marcado na Figura 5.3. O protocolo RBRP deve, então, sinalizar através de cores ou outro esquema quais as melhores fontes para a obtenção do recurso. No caso do exemplo, foram encontrados 3 arquivos diferentes com a mesma reputação. O usuário então deve escolher qual dos 3 arquivos lhe interessa mais.

Um detalhe importante é que o RBRP não força o usuário a escolher nenhum arquivo, nem realiza uma troca automática a partir da pesquisa. Essa troca automática não é viável, pois em alguns casos é possível que a busca retorne centenas de recursos com a mesma reputação, todos altos.

Outro detalhe é que a aplicação deve habilitar que o usuário faça uma pesquisa por um valor de reputação de um nó desconhecido. Essa pesquisa é feita caso o usuário deseje conhecer a opinião dos seus nós conhecidos sobre um nó desconhecido.

5.5.3 Cálculo de Reputação de Nós Desconhecidos

Um nó que deseja um recurso de um nó desconhecido por ele pode desejar conhecer sua reputação no sistema, para decidir se requisita ou não um determinado recurso que somente esse nó desconhecido possui, fato concluído através de uma busca pelo recurso no sistema. Essa verificação da reputação é mostrada na Figura 5.4. Primeiro, o nó que deseja conhecer o valor de reputação do outro nó envia uma mensagem aos nós que estão presentes em sua base de nós. Essa mensagem contém uma requisição sobre a reputação do nó que deseja conhecer.

Em seguida, os nós que conhecem esse nó descrito na requisição respondem com um certificado de nomes SPKI. O certificado é usado para associar uma chave a um nome. O nome no certificado corresponde ao papel que o nó da requisição pertence na tabela do nó que recebeu a requisição. Por exemplo, na Figura 5.4, o nó que envia a requisição é o nó "Q", o nó que responde é o nó "Z" e o nó requisitado é o nó "X". Pode-se ver, no exemplo, que o nó "Z" indica que o nó "X" é um nó definido com papel "amigo". Portanto, "Q" conclui que o nó "X" é um amigo de seu amigo.

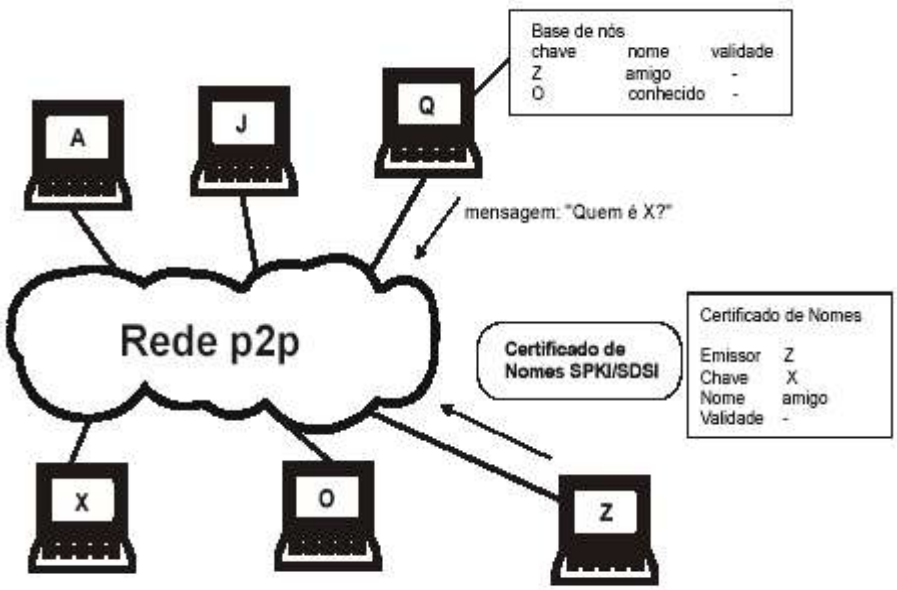

Figura 5.4: Requisição de Reputação.

A partir dessa informação, o nó requisitante tem algumas opções a fazer. A primeira é criar um papel “amigo de amigo” e cadastrar o nó nesse papel. A segunda é considerar o nó com o mesmo papel do nó que o respondeu, colocando o nó “X” no papel de “amigo”. O nó pode também ignorar a informação, talvez por não confiar na resposta de “Z”. Por fim, o nó pode colocar esse nó desconhecido por ele em qualquer outro papel que desejar.

Apesar de no exemplo mostrado o nó receber apenas uma resposta, o número pode variar. Da mesma forma que o nó “X” foi avaliado de forma positiva por “Z” e considerá-lo “amigo”, outros nós poderiam responder que “Z” tem papel de “nó carona” ou “distribuidor de pornografia infantil”. O usuário que requisita tais informações deve analisar todas as respostas para encontrar a melhor solução para avaliar a reputação de “Z”. Um ponto importante é a origem das informações. Se o nó não confia nos nós que respondem a requisição, então não deve confiar nas respostas que recebe.

Deve-se observar que a política seguida pelo usuário do sistema para definir sua base de nomes fica totalmente a cargo do usuário, assim como também é feito nos outros trabalhos sobre reputação descritos neste documento. Nada impede, porém, que um usuário ou um grupo deles usem *scripts* ou algum modelo para designar automaticamente os papéis aos outros nós do sistema. Um exemplo seria se, depois de uma troca de recursos com sucesso com um determinado nó, tal nó recebesse um papel chamado “conhecido” automaticamente.

Apesar da distribuição de papéis existente no RBRP não ser utilizada em nenhum dos outros trabalhos sobre reputação, alguns conceitos empregados no RBRP são provenientes desses trabalhos citados no capítulo 4, principalmente em (Kamvar et al., 2003, Damiani et al., 2002, Cornelli, 2002, Singh e Lui, 2003, Gupta et al., 2003). O conceito mais importante é a descentralização dos valores de reputação usada por (Kamvar et al., 2003). Diferentemente de (Damiani et al., 2002) e (Singh e Liu, 2003), o protocolo RBRP obtém diferentes valores de reputação dependendo do nó que requisita um valor de reputação de um determinado nó. Isso acontece pois o nó que faz a requisição leva em conta sua própria base de nós para verificar a reputação que deseja. Outro ponto em comum com os outros trabalhos é deixar a decisão sobre se um valor de

reputação é suficientemente alto ou não. Nenhum dos trabalhos, inclusive este, define valores mínimos ou máximos para aceitar ou recusar uma troca de recursos. Isso porque esses valores dependem da política adotada pelo requisitante do recurso.

5.6 Usuários novos

O protocolo RBRP não traz nenhum benefício aos nós que acabam de entrar no sistema. Essa restrição é necessária para que usuários maliciosos não troquem de identificador a todo momento para ganhar benefícios.

Quando usuários entram no sistema, possuem suas bases de dados sem nenhum valor, portanto devem confiar inicialmente em alguns nós para poderem utilizar, mais adiante, das vantagens do protocolo de reputação. Isso é o significado de reputação: aproveitar-se de experiências anteriores para conseguir segurança.

5.7 Especificação Formal

A utilização de métodos formais em projetos de sistemas garante o projeto de sistemas mais confiáveis, uma vez que técnicas de desenvolvimento inseridas por tais métodos se utilizam de linguagens formais que eliminam ambigüidades presentes na descrição informal. As técnicas de descrição formal (TDFs) são extremamente necessárias no caso de projetos de sistemas distribuídos devido à complexidade existente neste tipo de sistema.

Várias representações podem ser usadas para o projeto de sistemas distribuídos. Nesse trabalho, a representação usada é a *Language of Temporal Ordering Specification* (LOTOS) básica (Brinksma, 1988). Algumas das vantagens do LOTOS são (Alvarez, 2004):

- LOTOS é independente de linguagem de programação;
- LOTOS é um padrão internacional;

- LOTOS apresenta elevado índice de abstração.

A Figura 5.5 apresenta a especificação básica do protocolo RBRP para troca de valores de reputação. O protocolo realiza basicamente dois tipos de processos, a função de requisição e a de resposta. O processo de requisição, chamada de *rep_request*, é bastante simples. Ele é composto por apenas duas funções: identificar o nó que deseja receber a reputação e enviar os dados necessários. O outro processo, chamado de *receive_rep*, é ativado por um recebimento de uma mensagem de requisição de reputação. Nesse processo, ativa-se uma função de busca (*search_dbase*) na base de dados procurando pelo identificador do nó que foi informado na mensagem de requisição. Caso seja encontrado, é ativado um processo chamado *send_rep*. Caso contrário, a função *discard* é ativada.

```
process rep_exchange[get_servent_id, send_data, search_dbase, discard, send_rep] (servent_id, source_id): noexit :=

        (rep_request[get_servent_id, send_data] (servent_id, source_id)
        []
        receive_req[search_dbase, discard, send_rep] (servent_id, source_id))
        >> rep_exchange[get_servent_id, send_data, search_dbase, discard, send_rep] (servent_id, source_id)

        where

                process rep_request[get_servent_id, send_data] (servent_id, source_id] :=
                        get_servent_id (servent_id), send_data(servent_id, source_id), stop
                endproc

                process receive_req[search_dbase, discard, send_rep] (servent_id, source_id) :=
                        search_dbase(servent_id), discard[]send_rep(source_id), stop
                endproc
endproc
```

Figura 5.5: Especificação formal do protocolo RBRP.

5.8 Justificativa sobre o uso de SPKI

O protocolo, apesar de inicialmente não realizar nenhum tipo de controle de acesso usa um certificado de nomes para identificar um nó associado a um papel durante um cálculo de reputação. Embora esse uso possa ser trocado por uma outra estrutura parecida, o uso do certificado de nomes SPKI/SDSI, além de satisfazer todas as necessidades do protocolo, traz algumas vantagens ao modelo.

A principal vantagem é agregar ao modelo um protocolo padrão da Internet. Com o uso do certificado de nomes SPKI/SDSI a aplicação de um modelo SPKI de controle de acesso torna-se mais simples, já que parte da estrutura já deve estar implementada no sistema.

O SPKI é uma infra-estrutura de chaves públicas que se enquadra muito bem dentro do protocolo, pois não utiliza um centro de distribuição de chaves, o que também deve ocorrer em redes peer-to-peer puras. Além disso, a associação de chaves com nomes locais tem a mesma função da base de nomes usada no protocolo RBRP. O SPKI/SDSI também proporciona a geração de cadeias de confiança em uma rede p2p, uma de suas utilidades usuais.

Apesar disso, o SPKI limita o anonimato na rede, pois insere rótulos aos usuários do sistema. Entretanto, isso não traz, de modo geral, danos ao anonimato da rede p2p, pois esses rótulos correspondem apenas ao comportamento do nó, não a identificação do usuário do sistema.

5.9 Limitações do RBRP

O RBRP, apesar de atender certos problemas encontrados em protocolos de reputação, possui algumas limitações. Como o RBRP é baseado no uso de papéis para o cálculo de reputação, seu uso é mais indicado em redes p2p cujos nós representam seres humanos, como a Gnutella ou o KaZaA.

Outro ponto é a falta de uma política pré-estabelecida para a definição de um valor padrão de confiança em pesquisas por confiança de nós desconhecidos. Em outros protocolos como (Kamvar et al., 2003), os valores de reputação de nós desconhecidos são calculados automaticamente quando ocorre uma requisição de reputação. Entretanto, pode ser útil no sentido de deixar o usuário mais livre para definir o valor de reputação que deseja.

Apesar do uso do certificado de nomes SPKI/SDSI para cálculo de reputação, não é enfoque do protocolo realizar qualquer tipo de controle de acesso no sistema.

Entretanto, esse uso possibilita uma implementação de um mecanismo de controle de acesso baseado em SPKI/SDSI com certa facilidade, já que a estrutura do SPKI/SDSI já deve estar implementada no sistema para que ele funcione usando certificados de nomes SPKI/SDSI.

Outro ponto que o protocolo não trata com clareza é a troca de recursos compartilhados. Alguns sistemas p2p mais recentes, como o Emule, permitem que um usuário requisite um único recurso e obtenha o recurso de várias origens, transferindo uma parte de cada fonte. Caso tal recurso seja ruim, o protocolo RBRP deixa a cargo do usuário a decisão de quem diminuir o valor de reputação, se de apenas uma ou de todas as fontes usadas.

5.10 Conclusão

Neste capítulo foi apresentada uma proposta de arquitetura de um protocolo de reputação para redes p2p. A principal característica desse protocolo é o uso de papéis ao invés de valores numéricos para cálculo de reputação, presente em todos os outros protocolos pesquisados e explicados no capítulo 4.

Quatro vantagens desse protocolo podem ser citadas. A primeira é realizar uma classificação dos nós presentes em uma rede p2p de acordo com a confiança designada a cada um deles. A segunda vantagem é abstrair os valores numéricos para o usuário do sistema, tornando o protocolo mais simples para os usuários o seguirem. A terceira é organizar os valores de reputação de acordo com o tipo de dado desejado pelo nó requisitante. Finalmente, a última é agregar ao modelo a infra-estrutura de chaves públicas SPKI/SDSI.

Um ponto importante do protocolo é agregar ao seu modelo os conceitos de (Kamvar et al., 2003), o qual propõe que os valores de reputação devem depender do próprio nó que deseja conhecer tal valor. Isso permite que o sistema seja mais tolerante a gargalos de tráfego.

O principal ponto de limitação do protocolo é que o uso do protocolo deve limitar-se a redes p2p cujos nós são controlados por seres humanos, já que as máquinas têm maior dificuldade para tratar de nomes. Nesses casos, o uso de protocolos de reputação baseados em valores numéricos é mais eficaz. Ou, como alternativa, deve ser desenvolvido um sistema capaz de tratar de forma eficaz os nomes presentes no sistema.

Capítulo 6 Validação e Resultados

6.1 Introdução

Este capítulo apresenta o método de validação utilizado no trabalho e mostra os resultados alcançados neste trabalho através da implementação do protótipo desenvolvido.

6.2 Redução do valor médio de tentativas por download de recurso válido

Em sistemas p2p, podem existir diversos recursos inválidos ou impróprios dentro das redes. Exemplos são arquivos incompletos e conteúdo pornográfico, que podem ser requisitados devido a falsas informações nos metadados dos recursos. Portanto, os usuários desses sistemas podem realizar mais de um download para conseguir um recurso de boa qualidade.

Em redes p2p sem protocolos de reputação, o valor médio de tentativas de download varia exclusivamente de acordo com a proporção de usuários maliciosos presentes no sistema. Numa rede p2p cuja porcentagem de usuários maliciosos chega a 60% do total do sistema e cada tentativa de troca de recursos com tais nós é falha, cada usuário necessita de 2,5 tentativas para cada download com sucesso. Isso gera uma sobrecarga de 125% no tráfego de recursos dentro da rede[4].

Entretanto, o uso de protocolos de reputação auxilia na diminuição desse valor. Como os nós armazenam as boas experiências, futuras trocas de recurso realizadas entre nós com boa reputação têm menor probabilidade de encontrarem recursos inválidos. Se um nó designa corretamente outro nó como confiável, uma troca de recursos entre eles dificilmente trará um recurso impróprio.

[4] Tais conclusões podem ser tiradas a partir das fórmulas descritas nesta seção.

Para visualizar melhor o cenário, o modelo matemático para cálculo de largura de banda extra utilizada numa rede p2p é proposto a seguir. Considerando que uma rede p2p possui *n* nós, sendo que desse total *m* são maliciosos, considerando todos os recursos com o mesmo tamanho, a probabilidade de encontrar-se um recurso inválido é mostrada na fórmula 6.1:

$$P(i) = (m / n) * Im \text{ , para } n>1,\ m\geq 0 \text{ e } 0<Im\leq 1 \qquad (6.1)$$

Na fórmula 6.1, *Im* indica o valor médio de recursos inválidos de um usuário malicioso. Portanto, para um sistema com 10.000 nós, sendo 2.000 maliciosos e com um índice de recursos inválidos de 0,8, a probabilidade de se obter um recurso inválido é 0,16, ou seja 16%.

Essa probabilidade de se obter um recurso inválido aumenta o número de tentativas médio para se obter um recurso válido com sucesso. Esse número de tentativas *t* é ilustrado pela fórmula 6.2:

$$t = 1 / (1 - P(i)), \text{ para } P(i)\neq 1 \qquad (6.2)$$

Considerando um valor de P(i)=0,16, o valor de t ≅1,19. Isso significa que em uma rede com as especificações detalhadas, cada download com sucesso requer 1,19 tentativas de download. Isso gera um tráfego extra *e* no sistema, mostrado na fórmula 6.3:

$$e = t - 1 \qquad (6.3)$$

Como no exemplo usado o valor de *t* é de aproximadamente 1,19, o valor de *e* é 19%, ou seja, os nós maliciosos do sistema geram um tráfego inútil de 19% sobre o total necessário.

Esse excesso de tráfego pode ser diminuído com o uso do RBRP. Através do uso do RBRP os usuários são capazes de identificar nós amigos e nós maliciosos. Se um determinado usuário identificar uma quantidade suficiente de nós amigos e maliciosos, o número médio de tentativas de download irá diminuir e, portanto, o excesso de tráfego irá diminuir também.

Um usuário usando o RBRP é capaz de identificar *q* nós maliciosos. Sendo assim, a probabilidade de obter-se um recurso inválido é dada pela fórmula 6.4:

$P_r(i) = ((m - q) / n) * Im$, para $n>1$, $m\geq 0$, $0<Im\leq 1$ e $q\geq 0$ (6.4)

Usando o exemplo em questão, caso um usuário identifique 50 nós maliciosos, ou seja, $q = 50$, a probabilidade é $P_r(i) \cong 0{,}16$. Pode-se notar que a probabilidade praticamente não muda, então, apenas identificando nós maliciosos no sistema pouco ajuda na redução das tentativas de download.

Entretanto, um usuário RBRP também é capaz de identificar nós confiáveis. Esses nós confiáveis tanto possuem recursos válidos quanto podem informar sobre nós maliciosos que conhecem. Então, na fórmula 6.5, *r* é o número de nós confiáveis identificados e *s* é o valor médio de usuários maliciosos conhecidos por tais nós:

$P_{rb}(i) = ((m - q - (r * (s + 1))) / n) * Im$, para $n>1$, $m\geq 0$, $0<Im\leq 1$, $q\geq 0$, $r\geq 0$ e $s\geq 0$ (6.5)

Na fórmula 6.5, usando o já citado exemplo, com $r = 50$ e $s = 25$, o valor de $P_{rb}(i) \cong 0{,}05$. Portanto, nota-se uma mudança significativa no número médio de tentativas de download *t*, sendo $t \cong 1{,}005$, eliminando assim o tráfego inútil no sistema pois nesse caso $e = 0{,}005$.

6.3 Implementação

O protocolo RBRP pode ser implantado dentro de qualquer sistema peer-to-peer, já que cada usuário, independente dos demais, é capaz de classificar de acordo com papéis os outros nós do sistema. Basta adicionar a um nó as tabelas citadas na seção 5.5.1, alguns procedimentos para manipular essas tabelas e um suporte para manipular a infra-estrutura de chaves públicas SPKI/SDSI, que esse nó torna-se capaz de realizar a classificação dos outros nós do sistema. Entretanto, se os outros nós do sistema não forem adaptados ao uso do RBRP, as pesquisas de reputação não terão efeito, já que devem ter as respostas armazenadas nas bases de dados para responder às pesquisas.

Na arquitetura apresentada no capítulo 5, o sistema peer-to-peer usa uma chave pública para identificar um nó. O uso de uma chave pública de 1024 bits garante um identificador único e seguro para os nós, já que a quebra de uma chave de 1024 bits é computacionalmente inviável (Stallings,2003). Porém, nada impede que o RBRP seja implementado em um sistema que use pseudônimos para identificar os nós. O problema nesse caso é garantir a identificação única dos nós, já que os pseudônimos não garantem autenticidade.

6.3.1 DNET

A implementação realizada neste trabalho é como o modelo de implementação de (Damiani et al., 2002) que altera o código-fonte do Gnutella para que este funcione em conjunto com o protocolo de reputação, no caso deste trabalho o RBRP. O código-fonte do Gnutella pode ser encontrado nas linguagens C++ e Java na Internet, como a aplicação DNET (Dnet, 2004). Essa é uma aplicação de código-fonte aberto que funciona sobre o protocolo do Gnutella. O modelo de conexão usado na aplicação está representado na Figura 6.1.

Cada nó possui uma classe principal chamada *DataConnection*. Cada nó pode criar subclasses *ConnectionMinion* para que estas se conectem com outros nós. Na figura 6.1 o nó A está conectado com o nó B e o nó C, porém B e C não estão conectados diretamente. Cada *ConnectionMinion* pode usar as subclasses *DownloadMinion* para gerenciar um download e a *DNetClientMinion* para gerenciar uploads. No exemplo da figura o nó A está realizando um download de B e B realiza um download de A. Além disso, a classe *ConnectionMinion* permite que os nós enviem consultas para seus peers.

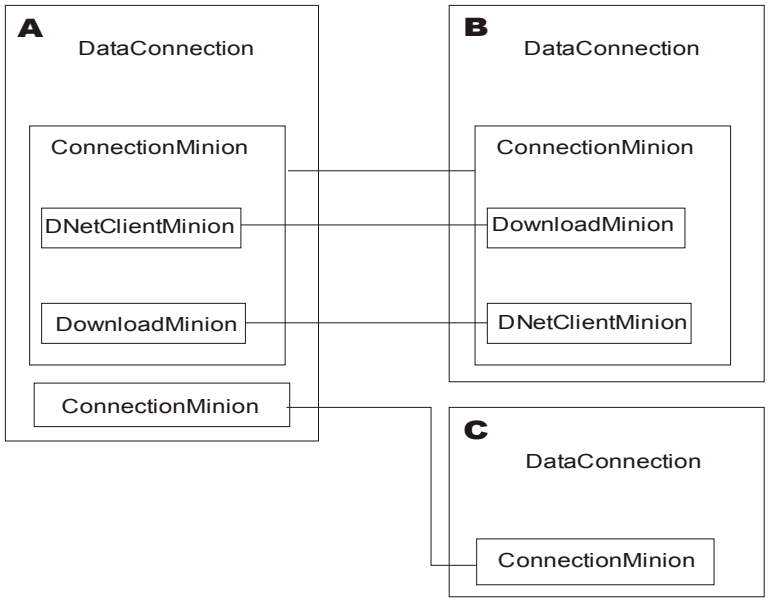

Figura 6.1: Exemplo de conexões entre nós Dnet

Para que o protocolo RBRP fosse adicionado nesse sistema, foram realizadas algumas modificações. O ponto mais importante é a interação do protocolo Gnutella com as bases de dados do RBRP. Para que o RBRP funcione, ao menos localmente, as duas bases – a base de nós e a base de nomes – são adicionadas ao sistema e devem ser preenchidas de acordo com as necessidades do usuário. O usuário deve ter a oportunidade de adicionar ou remover *servents* e papéis sempre que desejar. Além disso, depois que os nós estiverem cadastrados no sistema, a aplicação deve informar o usuário sobre os dados das bases sempre que ele deseje se comunicar com um nó conhecido.

Outro ponto importante é a segurança na comunicação dos nós dentro do sistema. Usuários maliciosos podem tentar atacar o sistema forjando mensagens de reputação ou mesmo fazendo se passarem por usuários com alta reputação. Para sanar esse problema, o RBRP usa criptografia assimétrica na identificação dos nós do sistema, além de outros mecanismos como *timestamp* para evitar ataques de repetição e valores *hash* para garantir a integridade de um recurso.

6.3.2 Formato das Mensagens

O protocolo RBRP possui basicamente 2 mensagens, a mensagem de requisição de reputação (*RepRequest*) e a de resposta (*RepResponse*), compostas por um cabeçalho (Figura 6.2) seguido de algumas informações (Figura 6.3). Uma mensagem de requisição pode ser gerada a qualquer momento por qualquer nó do sistema. Já a mensagem de resposta pode surgir a partir de uma mensagem de requisição recebida.

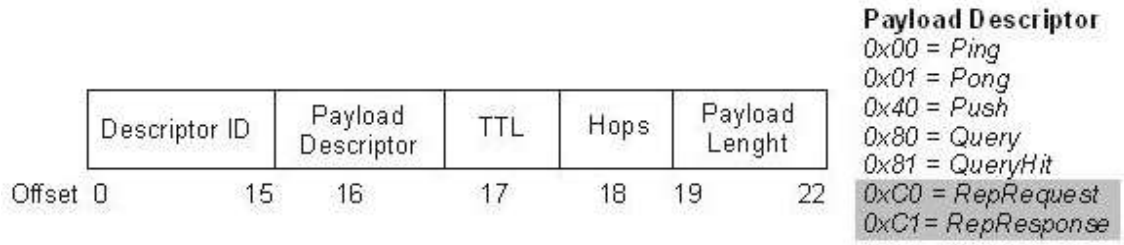

Figura 6.2: Formato do cabeçalho Gnutella estendido pelo RBRP.

As mensagens seguem o cabeçalho do protocolo Gnutella (Figura 3.2), porém com uma alteração no campo *Payload Descriptor*, como mostra a Figura 6.2. Esse campo deve aceitar dois valores novos, para as mensagens *RepRequest* e *RepResponse*. Esse formato similar ao Gnutella permite uma melhor adaptação do protocolo de reputação nas aplicações Gnutella existentes (Gnutella, 2004).

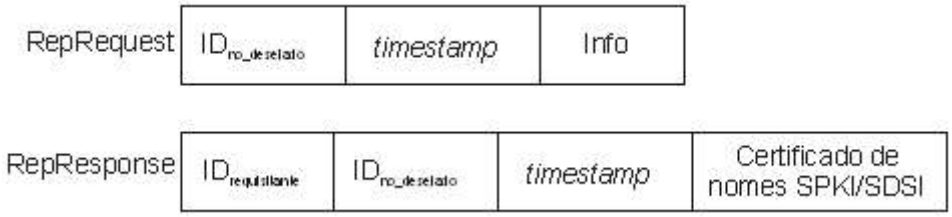

Figura 6.3: Formato da carga útil das mensagens RBRP

A Figura 6.3 apresenta o formato de uma mensagem de requisição de reputação (*RepRequest*). O primeiro campo, chamado de $ID_{no\_desejado}$, corresponde ao *servent* que se deseja conhecer o valor de reputação. O segundo campo corresponde a um valor de *timestamp* para proteger o protocolo de ataques de repetição. O terceiro campo (*Info*)

mostra algumas informações que podem ser passadas para auxiliar os nós no cálculo de reputação.

Um exemplo desse tipo de informação é o tipo de dado que está sendo desejado. Um nó pode, por exemplo, requisitar o valor de reputação de um determinado usuário com relação aos arquivos MP3 que este possui. Então essa informação deve ser passada no campo *Info*. Essas informações são semelhantes às informações de pesquisa do protocolo Gnutella (seção 3.5), servindo para auxiliar os nós a definirem um valor mais preciso de reputação dos outros nós.

A mensagem *RepResponse*, apresentada na Figura 6.3, possui 4 campos. Os três primeiros campos ($ID_{requisitante}$, $ID_{no\_desejado}$ e *timestamp*), assim como a mensagem *RepRequest*, apresentam respectivamente os mesmos significados da *RepRequest*. O quarto campo carrega um Certificado de nomes SPKI/SDSI, cuja funcionalidade já está explicada no Capítulo 5.

6.3.3 Interface com o usuário

Para que o RBRP seja utilizado, uma interface com o usuário é necessária para traduzir as necessidades de quem utiliza a aplicação. No caso da aplicação DNET, a interface existente realiza a interface com o protocolo Gnutella. Dessa forma, a aplicação realiza as funções de busca e requisição de recursos para o usuário. A interface da aplicação DNET está ilustrada na Figura 6.4. Pode-se observar na figura que duas janelas são importantes: a janela de controle de busca de arquivos e a janela de controle de downloads.

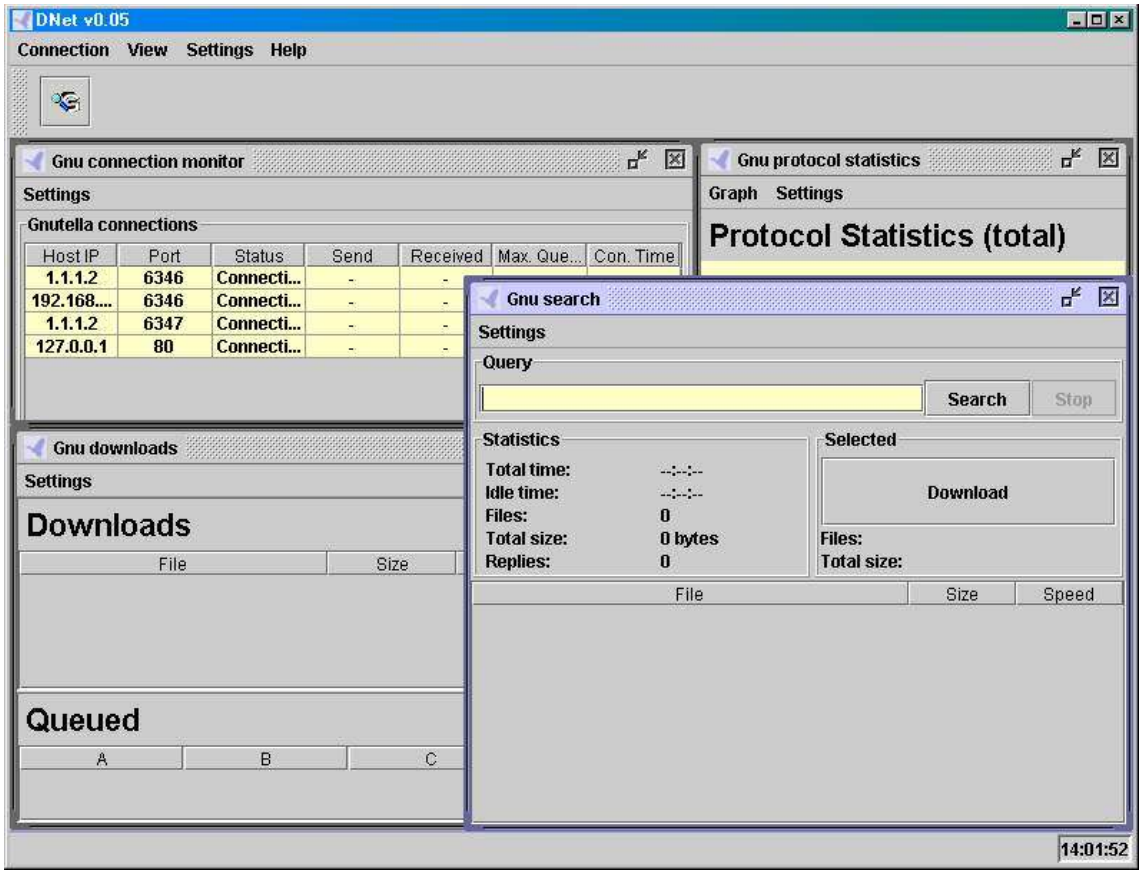

Figura 6.4: Ilustração da aplicação DNET.

Outro ponto importante da aplicação é a janela de controle de *peers*. Cada linha da tabela representa uma conexão direta entre dois pontos do sistema p2p. Na aplicação original DNET, assim como no protocolo Gnutella, os nós são representados por um valor IP e uma porta TCP. Já no caso do RBRP, os nós devem ser identificados por uma chave pública, para que os nós sejam persistentes em casos de quedas temporárias[5].

[5] Alguns pontos do sistema usam conexões com endereços IP temporários, então a identificação baseada em criptografia assimétrica é importante para que os nós permaneçam com a mesma identificação mesmo após sucessivas quedas ou saídas do sistema.

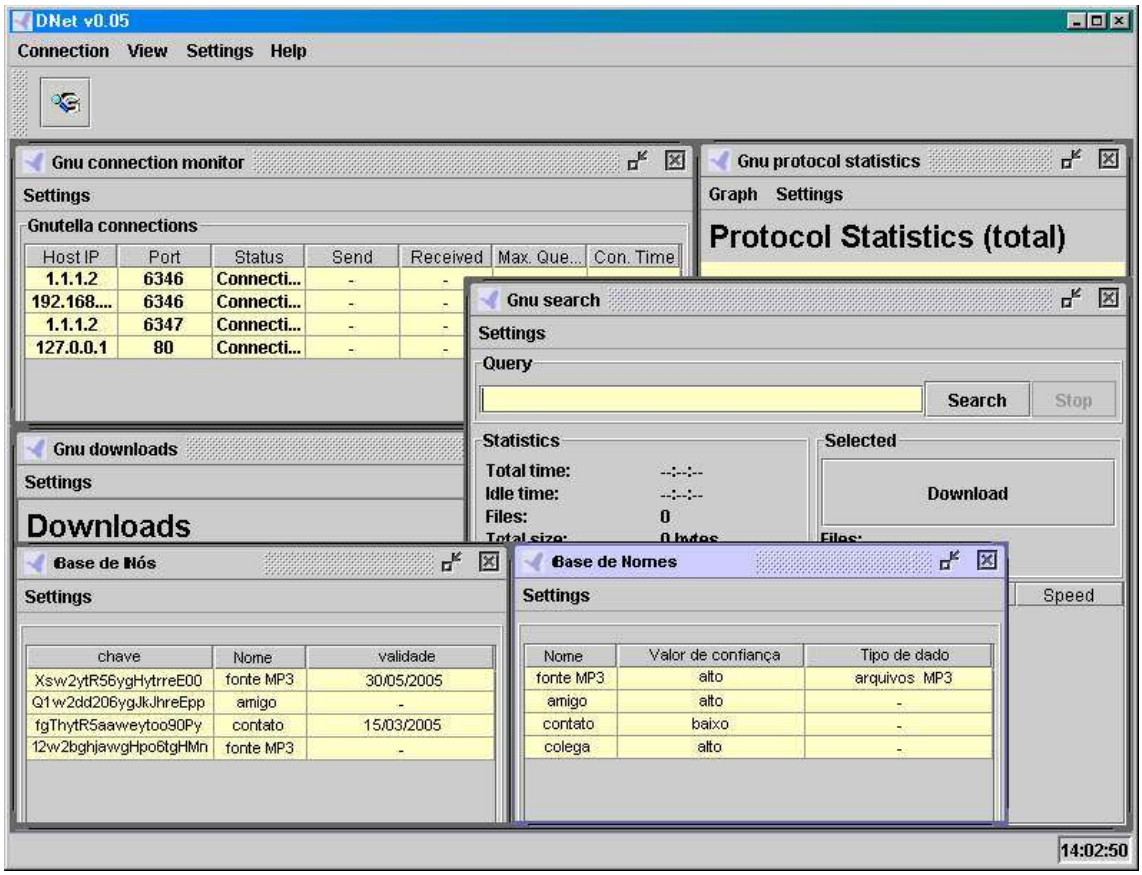

Figura 6.5: Aplicação DNET usando RBRP

Com a adaptação do protocolo RBRP, duas janelas adicionais são usadas, como mostra a Figura 6.5. Cada uma delas representa uma base de dados descritas na seção 5.6.1. A Base de Nomes associa nomes com valores de confiança e com os tipos de dados válidos. Já a Base de Nós associa chaves públicas a nomes. Vale lembrar que a aplicação Gnutella original foi modificada para suportar o uso de chaves para identificação de nós, portanto cada *servent* deve ser identificado, não apenas pelo endereço IP e por uma porta TCP, mas também por uma chave pública.

Para que seja de melhor visualização, a janela de procura (*Gnu Search*) também foi modificada. Os resultados de procura devem mostrar, além dos recursos encontrados, os *servents* que possuem os recursos e o papel que tais *servents* possuem, em caso de se encontrarem cadastrados na base de nós. Essa janela de procura deve também habilitar uma opção de busca por reputação, mostrando os nomes obtidos pela busca e as origens das respostas. Um exemplo do funcionamento dessa janela pode ser visto na Figura 5.3.

6.4 Conclusão

O protocolo RBRP é uma ferramenta simples no avanço da segurança em redes peer-to-peer, já que permite aos usuários avaliar outros nós do sistema anexando nomes aos identificadores desses pontos da rede. Essa afirmação é confirmada com a implementação do protocolo dentro da aplicação DNET.

Três pontos principais podem ser comentados sobre a implementação. O primeiro ponto é o uso de chaves públicas para identificação dos *servents*. Esse uso é necessário para tornar os nós persistentes no sistema, ou seja, que continuem a carregar seus valores de reputação mesmo após uma queda ou saída do sistema. Além disso, caso um nó consiga um endereço de IP que já pertenceu a outro ponto anteriormente, esse não carregue os valores de reputação do nó antigo.

O segundo ponto importante é a adição de dois novos tipos de mensagem no sistema. A primeira mensagem é a requisição de reputação propriamente dita usada para quando um nó deseja averiguar com seus vizinhos a reputação de um outro nó. A segunda mensagem é a resposta para essa requisição, usada quando um *servent* deseja informar sobre a reputação de um determinado nó requisitado.

O último ponto é o armazenamento das informações nas bases de dados do sistema e a interface dessas informações com o usuário. A aplicação deve fazer corretamente a associação da base de nós com as requisições de recurso para informar ao usuário do sistema sobre quais os nós com melhores valores de reputação presente no sistema que detêm a informação desejada.

Finalmente, pode-se provar matematicamente que o uso correto do protocolo dentro de uma rede p2p pode, além de trazer recursos de segurança ao sistema, diminuir a largura de banda utilizada na rede. Isso pode ser concluído pois a troca de recursos com nós maliciosos, além de poder danificar a rede e trazer problemas com vírus ou *worms*, gera um consumo de largura de banda inútil, pois ao final da troca o nó enganado certamente procurará outro nó que possua as informações que ele realmente deseja.

Capítulo 7 Conclusões

O modelo cliente-servidor pressupõe uma concentração dos recursos e do poder computacional apenas nos servidores – máquinas com maior capacidade de processamento e armazenamento. Contudo, com o aumento do número de computadores na internet e o aparecimento do desejo de troca de recursos compartilhados, a escalabilidade da rede tornou-se questão a ser resolvida.

A solução encontrada foi a descentralização do sistema com relação a algumas funções. Aplicações para troca de recursos, por exemplo, que anteriormente eram realizadas através de um servidor central, conseguiram uma autonomia para diminuir o poder de processamento necessário aos servidores. Esse novo modelo recebeu o nome de peer-to-peer, já que a comunicação era estabelecida com uma autonomia relativa a um servidor.

Apesar desse modelo peer-to-peer minimizar o problema da escalabilidade, o modelo peer-to-peer trouxe outros problemas para os sistemas. Alguns desses problemas são os relacionados com a segurança dos nós pertencentes ao sistema. Isso ocorre porque num modelo de sistema centralizado, o nó central deve se encarregar da maior parte da segurança do sistema. Já no modelo peer-to-peer, a centralização da segurança acarreta na perda de escalabilidade.

Dentre as preocupações com segurança existentes nas redes peer-to-peer, uma delas se destaca: a confiança entre os nós do sistema. Em redes peer-to-peer puras, como os nós possuem exatamente as mesmas funcionalidades, fica difícil encarregar um ou alguns pontos da rede de manter o sistema seguro. Devido a esse fato, mecanismos que agreguem ao modelo patamares de confiança auxiliam a manter o sistema mais apto a suportar ações maliciosas.

Para agregar valores de confiança a um sistema peer-to-peer, é usado normalmente um protocolo de reputação. Um protocolo de reputação avalia, com base em informações anteriores dos nós, os valores de confiança entre os pontos de uma rede

peer-to-peer. Essa avaliação é normalmente feita usando valores numéricos para estabelecer níveis de confiança entre os nós.

7.1 Contribuições

Este trabalho descreveu o Role-Based Reputation Protocol (RBRP), um protocolo de reputação que, diferente dos demais pesquisados, usa nomes ao invés de valores numéricos para avaliar um patamar de reputação. Essa diferença auxilia o sistema a realizar uma classificação dos nós com relação à reputação que cada ponto do sistema possui.

Outra importante contribuição desse protocolo é abstrair os valores numéricos presentes nos demais trabalhos, auxiliando na aceitação do uso do protocolo pelos usuários do sistema. Essa aceitação é importante porque o uso do protocolo deve ser feito por uma parte significativa da rede para que o protocolo produza efeitos aceitáveis com relação à segurança do sistema.

Uma terceira contribuição do protocolo é separar os valores de reputação por tipos de dados. Isso é importante pois uma fonte de dados pode ser confiável apenas para um tipo, como arquivos-texto. Além das contribuições já citadas, o protocolo usa uma opção de atribuir validade aos valores de reputação. Essa atribuição permite tabelas de reputação mais atualizadas, pois em redes p2p sempre é possível que nós abandonem o sistema. Quando isso ocorre, as descrições desse nó não devem ficar indefinidamente nas bases de dados dos outros nós.

Outro ponto importante do trabalho é aproximar o protocolo RBRP da infra-estrutura de chaves SPKI/SDSI. Essa aproximação permite uma futura aplicação de um modelo de controle de acesso com menores dificuldades. Além disso, o uso do SPKI/SDSI é importante pelo fato de tratar-se de um padrão internacional.

Além disso, o trabalho provou matematicamente que o uso pleno do protocolo pode trazer, além do benefício da segurança, benefícios com relação à largura de banda

consumida pelo sistema. Isso ocorre porque o protocolo de reputação, quando corretamente utilizado, ajuda a diminuir as trocas de recursos inválidas pela rede.

Embora possua diversas contribuições, o RBRP possui uma limitação com relação ao seu uso. Como o protocolo é baseado em nomes para o cálculo de reputação, o RBRP torna-se mais apropriado para sistemas cujos nós representam seres humanos. Redes peer-to-peer independentes de seres humanos, como a SETI@Home, não são capazes, a princípio, de tratar com nomes de reputação.

Outra limitação do RBRP é não atribuir valores de reputação padrões. Esses valores são úteis quando os usuários da rede não estão devidamente preocupados com a segurança do sistema. Entretanto, tais valores padrões podem não refletir as reais preocupações dos usuários.

7.2 Trabalhos Futuros

Essas duas limitações podem ser pesquisadas em trabalhos futuros. Além dessas duas questões, outras são válidas para estudo. Uma questão importante não tratada nesse trabalho é o uso do SPKI/SDSI para controle de acesso no sistema, aumentando ainda mais a segurança das redes peer-to-peer que utilizassem a arquitetura descrita nessa dissertação.

Outro trabalho futuro possível é a evolução dos filtros de tipos de dados pesquisados pelos sistemas p2p e pelos protocolos de reputação. O RBRP classifica os dados apenas com relação ao formato do arquivo (MP3, arquivo-texto). Uma idéia é, no caso dos arquivos MP3, por exemplo, classificá-los de acordo com o ritmo da música ou autor. Essa classificação pode ser alcançada através do uso do tipo MIME (*Multi-Porpouse Internet Mail Extentions*), um identificador de tipos único para arquivos da internet (RFC 2045).

Capítulo 8 Referências

Alvarez, L. M. C. (2004). *Modelo de Segurança Multilateral e RBAC em um Ambiente de Serviço no Contexto de Gerenciamento de Contabilidade TINA*. Tese de doutorado, UFSC, Florianópolis.

Bertino, E. (2003). RBAC Models – Concepts and Trends. *Computer and Security*, 22 (6) pp. 511–514.

Brinksma, E. (1988). *A Tutorial on LOTOS.* ISO 8807 Information Processing Systems – Open System Interconnection – LOTOS. A formal description technique on the temporal ordering of observational behavior, 1988.

Clip2 (2004). *The Gnutella Protocol Specification v0.4*. Document Revision. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf

Cornelli, F., Damiani, E., Vimercati,S. C., Paraboschi, S. and Samarati, P. (2002). Choosing Reputable Servents in a P2P Network. *In: 11th International World Wide Web Conference (WWW'02),* May 7-11, Honolulu, Hawaii, USA.

Crespo, A. and Garcia-Molina, H. (2002). *Semantic Overlay Networks for P2P Systems*. Technical Report, Computer Science Department, Stanford University.

Damiani, E., Vimercati, S. C., Paraboschi, S., Samarati, P. and Violante, F. (2002). A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks. *In: Computer and Communications Security (CCS'02)*, November 18-22, Washington DC, USA. 12 Páginas.

Dnet (2004). Website. http://schnarff.com/gnutelladev/source/dnet/dnet/Gnutella/

Elien, J. E. (1998). *Certificate discovery using SPKI/SDSI 2.0 certificates*. Dissertação de mestrado, MIT.

Ellinson, C. M. Frantz, B., Lampson, B., Rivest, R., Thomas, B. M., Ylonen, T. (1999). Simple Public Key Certificate. *Internet Draft*. July 26.

Emule (2004). Website. http://www.emule-project.net/

Ferreira, A. B. H. (1999). *Novo Dicionário Aurélio – Século XXI*. Editora Nova Fronteira.

Gnutella (2004). Website. http://www.gnutella.com

Gupta, M., Judge, P. and Ammar, M. (2003). A Reputation System for Peer-to-Peer Networks. *In: 13th International Workshop on Network and Operating Systems Suport for Digital Audio and Video (NOSSDAV' 03),* June 1-3, Monterrey, California, USA. 12 Páginas.

Gupta, R. (2003). *A Survey of Security Issues and Protocols in Peer-to-Peer Networks.* CS7210 Term P. http://www.cc.gatech.edu/people/home/rgupta/Docs/P2PSecurity.pdf

Kamvar, S., Schlosse, M. T. and Garcia-Molina, H. (2003). The EigenTrust Algorithm for Reputation Management in P2P Networks. *In: 12th International World Wide Web Conference (WWW'2003),* May 20-24, Budapest, Hungary. 12 Páginas.

Kan, G. (2001). *Gnutella Network*. Technical Paper. 9 Páginas.

Landwehr, C. E. (1981). Formal Models for Computer Security. *Computing Surveys ACM*.

Landwehr, C. (2001). Computer Security. *International Journal of Information Security,* 1(1) pp. 3–13.

Mello, E. R. (2003). *Redes de confiança em sistemas de objetos CORBA*. Dissertação de mestrado, UFSC, Florianópolis.

Moffett, J. D. (1995). *Security & Distributed Systems*. Encyclopedia of Microcomputers, vol. 15.

Molva, R. (1999). Internet Security Architecture, *Computer Networks*, 31(1999) pp. 787–804.

Napster (2004). Website. http://www.napster.com

Oram, A. (2001). *Peer-to-Peer: O Poder Transformador das Redes Ponto a Ponto.* ISBN 85-7251-519-7, Editora Berkeley.

Pellissari, F. R., Righi, R. and Westphall, C. M. (2004). RBRP: Protocolo de Reputação Baseado em Papéis para Redes P2P. *In: International Information Technologies Symposium (I2TS'04)*, December 6-9, São Carlos, SP, Brasil. 4 Páginas.

RFC 2045 (1996). Multipurpose Internet Mail Extensions(MIME) Part One:Format of Internet Message Bodies. http://www.ietf.org/rfc/rfc2045.txt

Righi, R., Pellissari, F. R. and Westphall, C. M. (2004). P2P-Role: Uma Arquitetura de Controle de Acesso Baseada em Papéis para Sistemas Colaborativos Peer-to-Peer. *In: Workshop em Segurança de Sistemas Computacionais (WSeg'04)*, May 10-14, Gramado, RS, Brasil.12 Páginas.

Righi, R., Westphall, C. M. and Pellissari, F. R. (2004). Escambo: Um Modelo de Comportamento e Reputação para Sistemas Peer-to-peer. *In: 2ª. Escola Regional de Redes de Computadores (ERRC'04)*, July 12-14, Canoas, RS, Brasil.6 Páginas.

Rocha. J., Domingues, M., Callado, A., Souto, E., Silvestre, G., Kamienski, C. e Sadok, D. (2004). *Peer-to-Peer: Computação colaborativa na internet*. Minicurso, SBRC2004, May 10-14, Gramado, RS, Brasil.

Singh, A. and Liu, L. (2003). TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P Systems. *In: 3rd IEEE International Conference on Peer-to-Peer Computing,* September 1-3, Linköping, Sweden.

Sit, E. and Morris, R. (2002). Security Considerations for Peer-to-Peer Distributed Hash Tables. *In: 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, March, Cambridge, MA, USA. 12 Páginas.

Stallings, W. (2003). *Cryptography and Network Security*. 3rd Ed., Prentice Hall.

Tanenbaum, A. S. (1995). *Redes de Computadores.* 3ª. Ed., Editora Campus.

Università di Milano Security Group (2004). Website. http://seclab.dti.unimi.it/p2prep/

Venter, H. S. and Eloff, J. H. P. (2003). A taxonomy for Information Security Technologies. *Computers and Security*, 22(4) pp. 299–307.

Vlachos, V., Theotokis, S. and Spinellis, D. (2004). Security Applications of Peer-to-Peer Networks. *Computer Networks,* 42(2) pp. 195–205.

Yang, B. and Garcia-Molina, H. (2003). Ppay: Micropayments for Peer-to-Peer Systems. *In: Computer and Communications Security (CCS'03),* October 27-31, Washington DC, USA.12 Páginas.