Centro Universitário de Brasília – UNICEUB

Faculdade de Tecnologia e Ciências Sociais Aplicadas – FATECS

Disciplina: Projeto Final

Implementação de um Modelo de Ambiente Computacional Seguro para Gerenciamento de Clientes Microsoft

Leonardo Vítor Chaves Rodrigues

RA: 2041667/0

Monografia de Conclusão do Curso de Engenharia de Computação

Orientador(a): Marco Antônio Araújo

Brasília – DF, 5 de julho de 2008.

Leonardo Vítor Chaves Rodrigues

Implementação de um Modelo de Ambiente Computacional Seguro para Gerenciamento de Clientes Microsoft

Trabalho de conclusão de curso apresentado como parte das atividades para obtenção do título de Bacharelado em Engenharia de Computação, do curso de Engenharia de Computação da Faculdade de Tecnologia e Ciências Sociais Aplicadas do Centro Universitário de Brasília – UniCEUB.

Profº orientador: Marco Antônio Araújo

Brasília – DF, 2008

Autoria: Leonardo Vítor Chaves Rodrigues

Título: Implementação de um Modelo de Ambiente Computacional Seguro para Gerenciamento de Clientes Microsoft

Trabalho de conclusão de curso apresentado como parte das atividades para obtenção do título de Bacharelado em Engenharia de Computação, do curso de Engenharia de Computação da Faculdade de Tecnologia e Ciências Sociais Aplicadas do Centro Universitário de Brasília – UniCEUB.

Os componentes da banca de avaliação, abaixo listados, consideram este trabalho aprovado.				
	Nome	**Titulação**	**Assinatura**	**Instituição**
1				
2				
3				

Data da aprovação: ____ de _____ de _____.

"Dedico este trabalho à minha família,
aos meus amigos, e em especial, à Deus, por
ter permitido sua realização."

AGRADECIMENTOS

Agradeço à todos que contribuíram, de forma direta ou indireta, para a realização deste trabalho.

Muito aprendizado, sem aplicação prática, é como um homem pobre que conta os tesouros dos outros, sem ter nem meio centavo para si.

Tradição Budista

RESUMO

Este trabalho apresenta uma proposta de implementação de um ambiente computacional seguro, que possa ser utilizado para gerenciar computadores clientes que utilizam sistema operacional Microsoft Windows em um ambiente de produção, a um baixo custo financeiro. Na composição de um conjunto de servidores apto a ser utilizado para diversos fins, serviços essenciais são discutidos e implementados, tais como gerenciamento de arquivos, serviços de impressão, contas de usuário, controle de acesso à Internet e procedimentos gerais de administração de sistemas. A segurança do projeto é considerada durante a implementação dos serviços, através de boas práticas de administração de sistemas e soluções de criptografia. O projeto é realizado utilizando *softwares* livres e abertos, essenciais para o baixo custo da implementação.

Palavras-chave: gerenciamento, segurança, windows, openbsd, samba.

ABSTRACT

This work presents a proposal of implementation of a low-cost secure computing environment which may be used to manage client computers that use Microsoft Windows operating system in a production environment. In order to assemble a set of servers which are suitable for a variety of uses, essential services are discussed and implemented, like file management, print services, user accounts, Internet access controls and general procedures of system administration. The project's security is considered during the implementation of the services, by using cryptography solutions and good system administration practices. The project is done using free and open softwares, which are essential for the low cost of the implementation.

Keywords: management, security, windows, openbsd, samba.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AD – *Active Directory*

AH – *Authentication Header*

BSD – *Berkeley Software Distribution*

BDC – *Backup Domain Controller*

BIND – *Berkeley Internet Domain System*

CA – *Certificate Authority*

CIFS – *Common Internet File System*

CUPS – *Common Unix Printing System*

DC – *Domain Controller*

DHCP – *Dynamic Host Configuration Protocol*

DMZ – *Demilitarized Zone*

DNS – *Domain Name System*

ESP – *Encapsulating Security Payload*

FQDN – *Fully Qualified Domain Name*

GNU – GNU*'s Not Unix*

GPL – *General Public License*

ICMP – *Internet Control Message Protocol*

IKE – *Internet Key Exchange*

IP – *Internet Protocol*

IPsec – IP *security*

ISA – *Internet Security and Acceleration*

ISC – *Internet Systems Consortium*

LAN – *Local Area Network*

LDAP – *Lightweight Directory Access Protocol*

NetBIOS – *Network Basic Input Output System*

NTLM – NT LAN *Manager*

PAM – *Pluggable Authentication Modules*

PDC – *Primary Domain Controller*

PKI – *Public Key Infrastructure*

RFC – *Request for Comments*

SA – *Security Association*

SMB – *Server Message Block*

TCO – *Total Cost of Ownership*

SUMÁRIO

INTRODUÇÃO

Um modelo de gerenciamento bastante comum e funcional para ambientes de produção com computadores interligados em rede, consiste em um gerenciamento de contas de usuário, arquivos e impressão, além de algum mecanismo de proteção ou controle de acesso para a rede, caso os usuários não sejam confiáveis, ou a rede esteja conectada à Internet. Devido à predominância de sistemas operacionais da família Microsoft Windows nos computadores de boa parte dos usuários de informática, muitos optam pela utilização de servidores que sejam também da família Windows, para realizar o gerenciamento de computadores clientes. Essa escolha é justificável, tendo em vista a alta compatibilidade oferecida pelas soluções cliente-servidor que utilizam sistemas operacionais da mesma família. Muitas empresas, porém, não desejam ou não possuem condições de investir na aquisição de licenças para um ou mais sistemas operacionais de servidores da família Microsoft Windows.

Motivado pela situação real e análoga vivida por uma instituição de ensino, o presente projeto visa a apresentar uma proposta de implementação de um ambiente computacional, utilizando um sistema operacional tipo Unix gratuito e demais ferramentas livres e de código aberto, que possa substituir boa parte das funcionalidades proporcionadas por soluções similares que utilizam softwares proprietários, em específico, o Microsoft Windows 2003 Server.

O projeto é desenvolvido tendo em mente a segurança do ambiente, fator crítico nos sistemas computacionais modernos, a facilidade de manutenção, e a integração entre os ambientes tipo Unix e Windows. Com relação à segurança, serão apresentadas boas práticas de administração de sistemas, e implementações específicas para prover maior segurança aos dados que trafegam entre clientes e servidores.

1 AMBIENTES COMPUTACIONAIS EM EMPRESAS

A Tecnologia da Informação (TI) tem modificado profundamente a maneira com que as empresas mobilizam seus recursos. Desde a atividade comercial mais simples, até as complexas e grandes corporações, o uso recente da TI tem apresentado possibilidades antes inimagináveis. Consegue-se melhoria na organização e na disponibilidade de dados, maior velocidade nas comunicações, melhor aproveitamento do tempo disponível, e inclusive possibilidades de redução de custos com mão-de-obra. Tais possibilidades, quando bem aplicadas, tendem a resultar em aumentos de produtividade, o que se traduz em lucros e vantagens competitivas, essenciais na atual economia mundial.

Com todos esses benefícios, é de se esperar que grande parte da produtividade e da vantagem competitiva de uma empresa venha a depender do bom funcionamento de sua estrutura de Tecnologia de Informação. Qualquer falha em pontos críticos desta estrutura, pode acarretar em sérios prejuízos. Até mesmo uma falha de poucos minutos, ocorrendo em um sistema bancário por exemplo, pode resultar em altos prejuízos para a organização em questão, devido ao alto número de transações bancárias que deixariam de ser feitas.

Tendo em vista a importância da área de TI de uma organização, o gerenciamento e a adequada implementação de recursos relacionados – como computadores clientes, servidores, sistemas operacionais, aplicações e redes de dados – se tornam cruciais para o sucesso da informatização da mesma. Será apresentado nos capítulos seguintes, propostas de planejamento e implementação acerca dos citados recursos técnicos, objetivando a montagem de um ambiente de gerenciamento computacional de baixo custo, passível de ser utilizado na informatização de uma organização para os mais diversos fins, tendo em mente as necessidades tecnológicas de uma empresa tanto em procedimentos administrativos, quanto em segurança da informação.

1.1 Serviços em um ambiente de gerenciamento computacional

Em um ambiente computacional, a natureza dos serviços variam de acordo com as necessidades de cada empresa ou organização. Em organizações onde a segurança dos dados é primordial, o uso de contas de usuário e políticas de permissões de acesso é importante. Outras organizações porém, não necessitam de rígidos controles de acesso, dispensando políticas de permissões, e até mesmo utilizando uma conta de usuário para todos os funcionários. Há exemplos de ambientes computacionais semelhantes, tais como um modelo de escritório seguro, utilizando o esquema de contas de usuário para cada funcionário, e um modelo de uma organização de caridade, onde devido à natureza dos funcionários (voluntários), que são trocados frequentemente, é preferível utilizar uma política de contas de usuário mais relaxada, permitindo que várias pessoas utilizem a mesma conta [TERPSTRA, 2005].

Existem enfim, diversos tipos de serviços, e diversas maneiras de prover um mesmo serviço. Todos porém, possuem o objetivo de ajudar a organização, seja proporcionando maior produtividade, maior segurança ou melhor administração de recursos.

Dentre os diversos serviços que podem ser oferecidos pela Tecnologia da Informação, há porém, aqueles que possuem um uso tão difundido, até mesmo em diferentes tipos de organizações, que os mesmos são considerados indispensáveis para grande número de empresas. Esses são serviços de natureza genérica, comum a vários tipos de ambientes computacionais. Os principais serviços deste tipo podem ser classificados como:

- Gerenciamento de acesso à recursos computacionais, utilizando o paradigma de contas de usuário (*login* e senha).
- Correlação entre usuários (ou grupos de usuários) e recursos computacionais que devem ser utilizados (permissões e políticas de uso).
- Controle de acesso à recursos externos ao ambiente computacional, como a Internet (através do uso de *proxies*, por exemplo).
- Controle administrativo sobre recursos internos como impressão e armazenamento de arquivos.

Os serviços citados são utilizados com o objetivo de oferecer uma estrutura de Tecnologia de Informação básica, e muitas vezes suficiente, para vários tipos de organizações.

1.2 Plataformas mais utilizadas

Em 14 de agosto de 2006, a empresa OneStat.com, especializada em *softwares* para análises de páginas na Internet, apresentou o resultado de um de seus estudos estatísticos, onde foi apontado que a Microsoft possui um monopólio no mercado de sistemas operacionais, com 96,97% dos usuários utilizando versões do sistema operacional Windows. Outra empresa, a Net Applications, que fornece serviços da mesma natureza da OneStat.com, disponibiliza na Internet um estudo realizado durante o mês de janeiro de 2008, onde é indicado que a Microsoft detém 91,46% do mercado de sistemas operacionais em ambientes *desktop*. Nas duas estatísticas, o Microsoft Windows XP lidera com mais de 75% de participação no mercado (86,8% de acordo com o OneStat.com). Tais análises apenas confirmam a popularidade do Microsoft Windows pelo mundo, e ressaltam sua importância tanto em um contexto comercial, quanto doméstico.

Devido à essa imensa popularidade, construir um ambiente computacional sem suporte à sistemas operacionais Microsoft Windows seria negligenciar uma grande parcela do mercado, restringindo sua utilização. Dessa forma, uma boa recomendação seria utilizar sistemas de gerenciamento altamente compatíveis, criados especificamente para gerenciar a família de sistemas operacionais clientes da Microsoft.

Do ponto de vista financeiro, a empresa que desejasse utilizar produtos Microsoft Windows Server teria de arcar com os custos de aquisição das licenças de *software*, que por sua vez aumentam de acordo com a expansão do parque computacional da organização, visto que esta teria necessidade de mais servidores para gerenciar mais clientes ou para prover serviços adicionais (como o Microsoft ISA Server para serviços de *proxy* e/ou *firewall*). Outro aspecto a ser considerado é o investimento financeiro em *hardware* para abrigar as soluções Windows Server, que geralmente demandam computadores mais robustos (e conseqüentemente mais caros) do que servidores tipo Unix para desempenhar funções semelhantes, conforme avaliado por um estudo realizado pela IBM. Levando em conta esses e

outros aspectos financeiros, a firma de TI australiana Cybersource realizou em 2002, e revisou em 2004, um estudo que foi publicado na revista eletrônica PCWorld.com por Matthew Broersma, e alardeado como sendo o primeiro estudo de TCO (TCO – *Total Cost of Ownership*) a analisar o custo total de posse entre sistemas Windows e Linux. O estudo indica que a criação de um ambiente Linux pode ser até 36% mais barato de se instalar e manter durante um período de três anos, do que a criação de um ambiente Windows.

Quanto à administração de sistemas, esta tarefa tende a ser mais amigável em sistemas da família Windows Server do que em sistemas tipo Unix, visto que em uma instalação Windows Server padrão, o administrador pode configurar boa parte do sistema através de interfaces gráficas e modelos pré-estabelecidos. Sistemas tipo Unix para servidores são tradicionalmente administrados via linha de comando, sendo difícil encontrá-los em versões que utilizem por padrão ferramentas gráficas para configuração. A utilização de informações apresentadas em janelas e o uso do *mouse* proporcionam uma interface gráfica intuitiva e fácil de ser entendida, ao contrário dos comandos e configurações em linhas de texto em sistemas tipo Unix, que podem intimidar os administradores. Há também outras facilidades, como atualizações automáticas do sistema e o suporte técnico disponível através da Microsoft. Algumas distribuições GNU/Linux e sistemas BSD não possuem atualizações automáticas (tal como o OpenBSD), e o suporte técnico de qualidade é pago. Muitas dessas facilidades podem ser obtidas também em um ambiente tipo Unix, como automatização de tarefas e interface gráfica para gerenciamento, porém, as mesmas exigem uma certa pesquisa e estudo por parte do administrador.

Por fim, há o quesito segurança, ponto polêmico e bastante discutido em sistemas Microsoft Windows, que possuem a fama de serem “inseguros”. Jeff Jones, funcionário da Microsoft na unidade de tecnologia de segurança, publicou em seu site um estudo sobre vulnerabilidades de segurança em sistemas Windows 2003 e Red Hat Linux (versão 3 e 4), durante o primeiro semestre de 2006. Seus resultados apontam que o Windows 2003 obteve uma menor quantidade de falhas de segurança em relação ao Red Hat Linux, um famoso sistema operacional tipo Unix. Outros estudos porém, como o realizado pelo colunista do site de TI Computerworld, Nicholas Petreley, e publicado na página de noticias de TI TheRegister.co.uk, apontam que sistemas Windows são mais vulneráveis do que sistemas

Unix quanto à segurança. Vários outros estudos da mesma natureza existem, e é difícil julgá-los, pois muitos se concentram em determinadas métricas ou aspectos e descartam outras, o que altera o resultado final da análise em questão, com um sistema sendo anunciado como mais seguro que outro. É uma realidade porém que sistemas Windows são altamente suscetíveis aos *malwares* existentes, como vírus, cavalos-de-tróia ou *worms*, onde a grande maioria destes possui como alvo sistemas Windows, possivelmente devido à sua popularidade e/ou seu histórico de falhas de segurança. A empresa Secure Computing, que atua no ramo de segurança de *gateways*, divulgou em sua página na Internet, relatórios sobre *malwares* capturados pela empresa durante o ano de 2007. Uma de suas análises indica que 97% dos *malwares* descobertos durante o período de uma semana, são do tipo executáveis para Windows. Conforme um relatório escrito por Konstantin Sapronov, e publicado em um site mantido pela Kapersky Labs, sistemas tipo Unix possuem uma quantidade bem menor de *malwares* desenvolvidos para infectá-los, o que contribui para uma avaliação final positiva sobre a segurança destes sistemas.

2 APRESENTAÇÃO DOS CENÁRIOS

Este projeto nasceu da necessidade de se resolver problemas, tanto técnicos como administrativos, que vinham ocorrendo no departamento de informática de uma instituição de ensino. Os capítulos seguintes apresentam uma descrição dos cenários considerados para este projeto, juntamente com os problemas apresentados e as soluções propostas.

2.1 Situação atual

A figura abaixo representa a configuração de rede utilizada atualmente pela instituição de ensino:

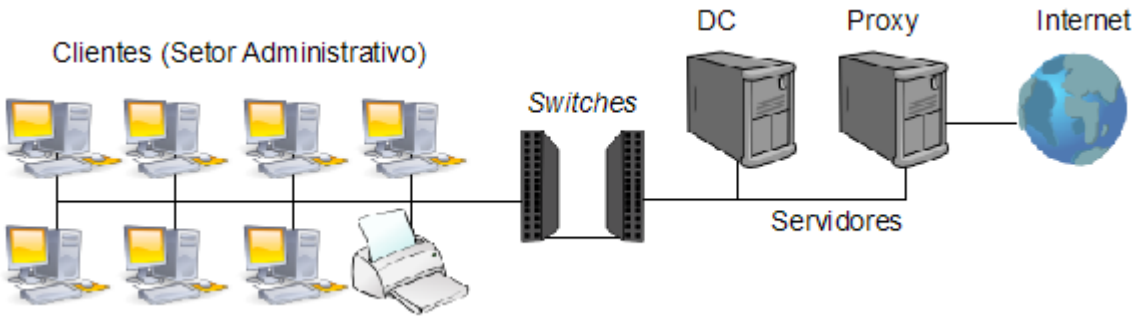

Figura 1: Situação atual.

A rede interna é composta pelos computadores clientes, que utilizam sistema operacional Windows XP, um servidor Windows 2003 Server atuando como controlador de domínio (*Domain Controller* – DC), e um servidor OpenBSD atuando como servidor *proxy*, todos interligados por *switches*, localizados nos diversos departamentos da instituição. Essa infra-estrutura proporciona aos funcionários os recursos computacionais necessários para o funcionamento dos serviços relacionados à Tecnologia de Informação utilizados pela instituição. O servidor principal possui o sistema operacional Windows 2003 Server,

configurado para relacionar nomes de *hosts* de forma dinâmica, (serviços de DHCP/DNS dinâmico, denominado *dynamic updates*), armazenar arquivos, gerenciar contas de usuário, permissões, e prover controle de impressão para os computadores clientes, que utilizam Windows XP. Grande parte dessas funções são realizadas através do protocolo SMB/CIFS, descrito no próximo capítulo. O servidor Proxy utiliza o sistema operacional livre OpenBSD, juntamente com o software Squid para realizar as funções de *proxy* e *caching* para a Internet. Não é possível a utilização de um servidor *proxy* específico para ambientes compostos por sistemas Windows, como o Microsoft ISA Server, pois a instituição não está disposta a arcar com os custos para adquirir tais *softwares*.

2.1.1 Problemas apresentados

O cenário existente tem apresentado problemas para a equipe de informática da instituição. A maioria dos problemas está relacionado com o processo de autenticação entre os usuários e o servidor *proxy*. Este processo é atualmente realizado de acordo com os seguintes passos:

a) Ao abrir uma sessão em um navegador de Internet, as configurações de *proxy* na máquina do cliente o forçam a requisitar sua sessão de Internet pelo servidor Proxy.

b) O servidor Proxy, por sua vez, não possui as informações do usuário, e então as solicita ao cliente.

c) O cliente é forçado a digitar um nome de usuário e uma senha correspondente para receber o acesso.

d) O servidor Proxy contata a base de usuários do servidor DC (que é organizado em uma estrutura LDAP), e verifica se o nome de usuário e senha recebidos correspondem à uma conta válida.

e) Caso positivo, o usuário recebe seu acesso, de acordo com as regras de conteúdo do Proxy. Em caso negativo, o usuário tem seu acesso negado.

Dessa forma, a autenticação entre os usuários e o servidor *proxy* não é transparente, pois, os clientes são forçados a digitar seu nome de usuário e sua senha a cada nova sessão para navegação na Internet, como demonstra a figura a seguir:

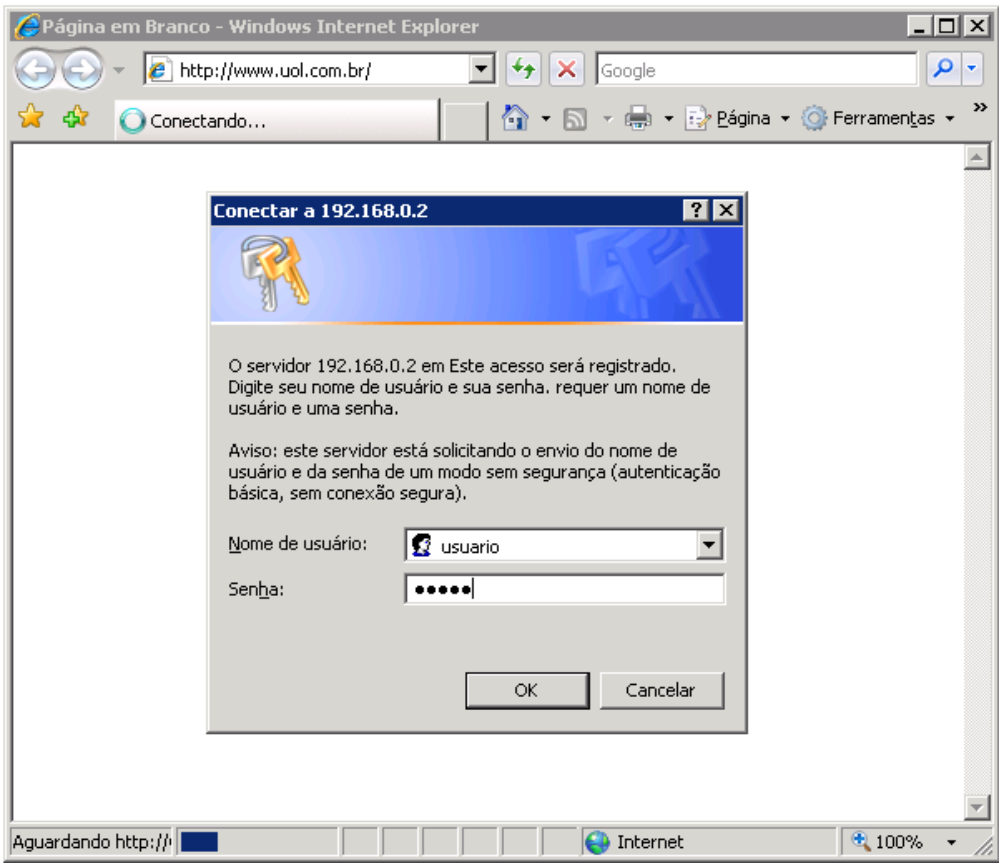

Figura 2: Tela de autenticação do servidor *proxy* existente.

Essa requisição de dados não é desejada, visto que o usuário já utilizou seu nome de usuário e sua senha quando realizou o processo de *login* no sistema operacional, sendo forçado então, a entrar duas vezes com a mesma informação. Outro ponto preocupante deste mecanismo, é que qualquer pessoa pode utilizar uma conta de usuário conhecida. Ou seja, mesmo que o *login* de sistema operacional seja feito com um usuário, nada impede que o acesso à Internet seja feito pela conta de outro usuário. Esse processo é uma falha de segurança, pois uma pessoa pode se fazer passar por outra nos registros de acesso à Internet, e um usuário cuja conta de *login* não foi contemplada com o acesso à Internet, pode utilizar uma conta de outro usuário para realizá-lo.

É possível notar também pela advertência do Windows, que o envio do nome de usuário e da senha é feito de modo inseguro, através de texto claro pela rede. Isso constitui um problema sério, pois uma pessoa qualquer que possua acesso à rede, pode utilizar alguma

ferramenta de escuta como o ethereal[1] ou o wireshark[2] para interceptar esses dados sensíveis, os quais deveriam ser confidenciais e devidamente protegidos. Este problema será apresentado com mais detalhes no capítulo 3 desta monografia.

A utilização do Microsoft ISA Server para prover os serviços de *proxy* solucionariam os problemas acima apresentados, porém, apresentariam custos financeiros com licenças de *software*. A instituição de ensino apresentada, não está disposta a pagar estas licenças, e outras empresas ou instituições que por ventura estejam na mesma situação, também podem não estar dispostas a arcar com esses custos.

O cenário existente também apresenta um outro problema de segurança, que é a falta de confidencialidade dos dados durante transferências de arquivos. Conforme será explicado no capítulo 3, os arquivos que trafegam na rede pelo protocolo SMB/CIFS, utilizado pelo Windows, podem ser interceptados e copiados por usuários não autorizados.

Por fim, a instituição está em fase de expansão de seus serviços de T.I., com planos para a instalação de uma nova infra-estrutura em um novo prédio, que já está parcialmente concluído. Esta expansão demandará novos servidores, implicando em novas licenças de servidores Windows 2003 Server. A instituição porém, não gostaria de gastar mais dinheiro em licenças, preferindo investir estes recursos financeiros em equipamentos de *hardware*. Dessa forma, a equipe de T.I. recebeu ordens para buscar alternativas ao Windows 2003 Server para compor os servidores deste novo prédio.

2.1.2 Proposta de solução

Devido às restrições financeiras, decidiu-se pela utilização de *softwares* livres para compor os novos servidores da instituição, utilizando tais *softwares* também para efetuar uma substituição dos servidores já existentes, visando manter uma homogeneidade de sistemas. Além disso, é necessário resolver os problemas apresentados anteriormente, relativos à segurança e ao servidor Proxy.

---

1 Aplicativo para análise de pacotes, disponível em: <http://www.ethereal.com>. Acesso em 5 de julho de 2008.

2 Aplicativo para análise de pacotes, disponível em: <http://www.wireshark.org>. Acesso em 5 de julho de 2008.

Propõe-se neste projeto, a utilização do protocolo NTLM para realizar a autenticação dos usuários com o servidor Proxy. Dentre os métodos de autenticação possíveis listados pelo *software* de *proxy* Squid, o NTLM se destaca por ser aceito por padrão pelos mais famosos navegadores, como o Internet Explorer e o Mozilla Firefox, evitando assim configurações extras nos computadores clientes. Além disso, os outros tipos de protocolos de autenticação listados, não seriam interessantes, pois muitos utilizam outro meio de base de dados de usuários para autenticação (LDAP, PAM), alguns enviam a senha em texto claro (como o método atualmente utilizado), e principalmente, por não realizarem a autenticação de forma transparente para o usuário, como o NTLM faz. Para realizar a autenticação dos usuários com o servidor Proxy, uma implementação deste servidor deve ser feita, tanto para utilizar o protocolo NTLM, quanto para permitir que o Proxy busque as informações de autenticação em um repositório de contas criado em um servidor composto por *softwares* livres e abertos, e não mais em um servidor Windows 2003 Server.

Dentre os *softwares* livres e abertos considerados para substituir servidores Windows 2003 Server, foi escolhido o *software* Samba, cujo objetivo é reimplementar serviços disponíveis em sistemas operacionais da família Windows (incluindo a família Server e NT), permitindo interoperabilidade entre sistemas Windows e tipo Unix. O seu código fonte é livre e aberto, sujeito a uma licença de público geral (GPL), e pode ser compilado e utilizado na maioria dos sistemas tipo Unix existentes. Neste projeto, o Samba será utilizado para substituir serviços que antes eram fornecidos pelo *Active Directory* do Windows 2003 Server, como gerenciamento de contas e perfis de usuário, políticas de segurança, arquivos e impressão.

Para prover essa interoperabilidade entre sistemas Windows e tipo Unix, o Samba implementa o protocolo SMB/CIFS, utilizado por sistemas Windows para a realização das mais variadas atividades de rede, tais como transferência de arquivos e autenticação. Infelizmente, a implementação deste protocolo pelo Samba também possui a mesma falta de confidencialidade citada no capítulo anterior.

Devido à importância deste protocolo para o ambiente computacional proposto, uma breve análise sobre o SMB/CIFS será realizada no capítulo seguinte, visando formar uma base para a abordagem que será feita no capítulo 3, sobre sua segurança. Também nesse capítulo,

será realizada uma explicação sobre a estrutura IPsec, cuja utilização é proposta neste projeto para solucionar a falta de confidencialidade do protocolo SMB/CIFS.

Por fim, para evidenciar a preocupação com a segurança neste projeto, e proporcionar algumas implementações específicas nesta área (IPsec), foi escolhido o OpenBSD como o sistema operacional tipo Unix a ser utilizado em todos os servidores. Este é um descendente direto do BSD (*Berkeley Software Distribution*), que por sua vez é um derivado do Unix original desenvolvido pela AT&T. O OpenBSD é software livre e aberto, distribuído sob uma licença BSD, e é bastante conhecido pelo seu foco em segurança e exatidão de código, pela qualidade de sua documentação e de seu código fonte, e pela sua posição firme com relação às licenças de *software*. A escolha deste sistema operacional se deve a alguns méritos técnicos considerados, como o foco deste na área de segurança e estabilidade, o histórico de lançamentos de novas versões sempre a cada seis meses – o que facilita o planejamento de períodos de manutenção – e o fato de seu desenvolvimento abranger *kernel* e utilitários (*userland*), proporcionando assim um sistema operacional completo e bem integrado, desenvolvido pela mesma equipe, e não apenas um *kernel* com vários utilitários desenvolvidos por pessoas diferentes – como é o caso do GNU/Linux e suas distribuições – o que levaria a uma cansativa pesquisa sobre qual distribuição GNU/Linux usar dentre as várias possíveis.

Em geral, a utilização de servidores tipo Unix para compor um ambiente computacional apresenta boas vantagens. Muitos dos mais famosos sistemas operacionais tipo Unix, como o Linux e o FreeBSD, são *softwares* livres e abertos, descartando a necessidade de investimento financeiro para poder utilizá-los. Vários aplicativos úteis em um ambiente de gerenciamento, como o Samba e o Squid, possuem versões nativas para ambientes Unix, e muitos desses são também *softwares* livres e abertos. Devido à natureza do código livre, o administrador possui a liberdade de alterá-lo de acordo com sua necessidade, podendo até mesmo inspecioná-lo para verificar sua segurança. Essa liberdade de ação sobre o código fonte não está facilmente disponível em sistemas Windows.

2.1.3 Sobre o protocolo SMB/CIFS

O SMB (*Server Message Block*) é um protocolo de rede, situado na camada de aplicação, e utilizado para compartilhar arquivos, impressoras, ou outras abstrações de comunicação, entre os nós de uma rede computacional. Este protocolo foi desenvolvido na empresa multinacional IBM, no começo dos anos 80, sendo aprimorado nos anos seguintes pela Microsoft e por outras empresas. No ano de 1996, após várias alterações e extensões, a Microsoft batizou sua implementação deste protocolo com o nome de CIFS (*Common Internet File System*). Ocorreram durante este tempo, várias tentativas visando a documentação e padronização do protocolo SMB/CIFS, sendo divulgadas especificações pela Microsoft, pelo The Open Group, e pela SNIA (*Storage Network Industry Association*), porém, ainda hoje não há consenso sobre uma especificação única e autoritativa, que possa ser utilizada como referência para as implementações do protocolo. É fato porém, que devido ao monopólio exercido pela Microsoft no setor de computação doméstica, sua implementação do SMB/CIFS é o padrão pela qual todas as outras implementações são medidas [HERTEL, 2003].

Em um ambiente computacional composto por sistemas operacionais Microsoft, o SMB/ CIFS é o protocolo padrão utilizado para realizar o compartilhamento de arquivos ou impressoras. Seu funcionamento é baseado na arquitetura pedido-resposta (*request-response*), onde o cliente envia os pedidos, e o servidor retorna as respostas condizentes. Após o estabelecimento de uma conexão, um cliente pode manusear arquivos, impressoras, ou outros recursos, como se os mesmos estivessem em sua máquina local. Porém, devido à abstração realizada pelo protocolo SMB/CIFS, tais recursos estão na verdade situados em um local remoto, tal como um servidor de arquivos ou de impressão localizado em outro ponto da rede.

2.2 Cenário proposto

Este projeto propõe a utilização do seguinte cenário:

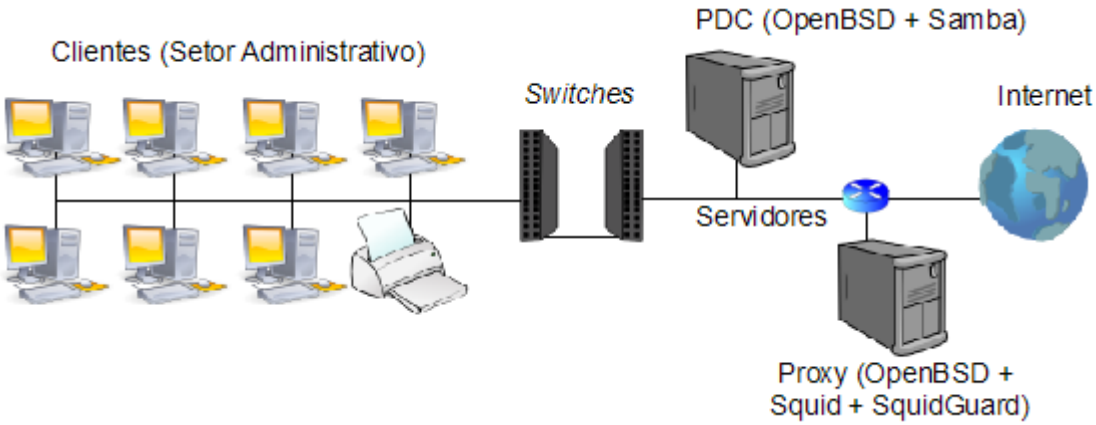

Figura 3: Cenário proposto.

O esquema acima descreve uma configuração clássica de redes de computadores. Os clientes utilizam Windows XP, enquanto o PDC e o Proxy utilizam o sistema operacional OpenBSD 4.2. A conexão entre os clientes e o PDC é feita através de um *switch* de rede, que se comunicará com o Proxy e a Internet através de um *gateway*, devendo este desempenhar funções de roteamento e filtragem de pacotes (*firewall*).

O servidor PDC atuará como o controlador de domínio primário, (*Primary Domain Controller*), substituindo um controlador de domínio Windows 2003 Server, e será configurado com os serviços de DNS, DHCP, controle de impressão, armazenamento de arquivos e gerenciamento de contas, perfis de usuário, e políticas de segurança. Por fim, o servidor Proxy será configurado com os serviços de *proxy* e *caching* de conteúdo para a Internet, além de um serviço de DNS para realizar um *caching* de nomes de domínio externos à rede interna, para agilizar as requisições dos clientes por endereços na Internet.

Serão utilizadas as seguintes definições para a configuração de rede deste projeto, porém, as mesmas são arbitrárias, e devem ser adaptadas para cada tipo de ambiente de rede:

- Nome de domínio utilizado: exemplo.br
- FQDN para o servidor PDC: pdc.exemplo.br
- Endereço IP para o servidor PDC: 10.1.1.1

- FQDN para o servidor Proxy: proxy.exemplo.br
- Endereço IP para o servidor Proxy: 10.1.1.5
- Máscara de Rede: 255.0.0.0 (/8)

De acordo com as configurações acima, os servidores estarão localizados na mesma subrede. Este projeto utilizará a configuração citada apenas para facilitar a demonstração deste projeto para a banca examinadora, excluindo-se assim, a necessidade de configuração de um *gateway* para gerenciar redes separadas. Ressalta-se porém, que na implementação deste projeto em um ambiente de produção, é recomendável utilizar uma rede separada para o PDC e os clientes, e outra para o Proxy, por questões de segurança . Neste conceito, haveria uma rede local (denominada LAN, Local Area Network), que abrigaria os serviços que dizem respeito à rede interna da organização, tais como servidor de arquivos, impressão, e base de usuários, e uma outra rede, que seria a zona desmilitarizada (denominada DMZ, *demilitarized zone*), específica para os servidores que requerem acesso direto à Internet, ou que contém serviços que são por ela acessados diretamente. Esta configuração pode ser obtida com o esquema apresentada, configurando-se um *gateway* com regras específicas para o isolamento adequado entre as duas redes, LAN e DMZ, e a Internet. Com esta configuração, caso algum servidor contido na DMZ seja comprometido, o usuário mal-intencionado não possuirá acesso à rede interna, devido ao isolamento de redes proporcionado pelas regras impostas pelo *gateway*. Este projeto não abordará a configuração deste *gateway*.

3 SEGURANÇA DE DADOS

Nas redes de computadores atuais, é imprescindível preocupar-se com a segurança dos dados. As redes de muitas empresas não se encontram mais isoladas como antigamente, mas conectadas à Internet, o que pode gerar um ponto de entrada para uma variedade de ameaças desconhecidas, como *crackers*, *malwares*, dentre outras. Além disso, nunca se sabe se um funcionário insatisfeito procura roubar ou apagar informações para prejudicar a empresa, se beneficiar de alguma forma, ou apenas causar o caos por qualquer outro motivo.

É importante que o administrador de sistemas tome medidas para melhorar a segurança da rede que administra, preocupando-se com permissões de usuários e seguindo boas práticas de administração, sempre visando evitar ou até mesmo eliminar ameaças.

Os tópicos seguintes abordam os aspectos de segurança que serão abordados na proposta de ambiente computacional deste projeto. Dentre os tópicos apresentados, estão alguns mecanismos de segurança oferecidos pelo sistema operacional OpenBSD, uma análise da falta de confidencialidade presente no protocolo SMB/CIFS, o problema da autenticação em texto claro que é atualmente realizada pelo servidor Proxy, e uma breve explicação sobre a estrutura IPsec, proposta neste projeto para fornecer confidencialidade ao protocolo SMB/CIFS.

3.1 OpenBSD como sistema operacional

O sistema operacional OpenBSD é conhecido pela seu foco em segurança, e em exatidão de código. Muitas distribuições GNU/Linux ou outros sistemas operacionais tipo Unix não possuem ou não implementam por padrão alguns mecanismos utilizados pelo OpenBSD, como revogação e separação de privilégios para serviços essenciais (tais como os servidores Xorg, NTP e *web*), proteção de memória (W^X, malloc() e mmap() aleatórios), proteção contra estouros de pilha (ProPolice compilado por padrão) e criptografia integrada no sistema operacional. Além disso, o projeto OpenBSD adota uma política de revisão de todo o seu código fonte pelos desenvolvedores, sempre que uma falha de segurança é encontrada. Nestes casos, os desenvolvedores liberam todas as informações sobre a falha, juntamente com correções, em um processo chamado de *full disclosure*. Em uma comparação com o Windows, neste não há fácil acesso ao código fonte do sistema, sendo que para ambientes onde um alto nível de segurança é requerido ou desejado, a falta do código fonte pode constituir um grave problema.

3.2 Segurança do protocolo SMB/CIFS

No início do protocolo SMB, as redes locais eram pequenas e isoladas, e a questão da segurança não era seriamente considerada. Se um cliente requisitava um arquivo de algum servidor, um processo simples de autenticação era iniciado, e os dados de autenticação como nome de usuário e senha eram enviados em texto claro pelo protocolo SMB [HERTEL, 2003].

Com o advento da Internet, a expansão das redes locais, e a massificação do uso, a segurança do protocolo se tornou um problema sério. O SMB passou então a utilizar mecanismos de autenticação mais sofisticados, baseados em um processo de *challenge-response*, ou desafio-resposta, para processar a autenticação. Enquanto no modo de texto claro o cliente provava quem ele era enviando a senha até o servidor, no modo de desafio-resposta o cliente não arrisca nenhuma exposição de senha para provar quem ele é. Isso é feito utilizando um processo de criptografia, onde o servidor envia ao cliente um desafio criado aleatoriamente, o qual é então criptografado tanto pelo cliente como pelo servidor, utilizando uma chave secreta (que é derivada da senha do usuário). O cliente então envia este resultado

(resposta) de volta ao servidor. A autenticação é bem sucedida quando a resposta do cliente combina com o resultado obtido pela criptografia do servidor [HERTEL, 2003]. A seguinte figura ilustra este processo de autenticação via *challenge-response*:

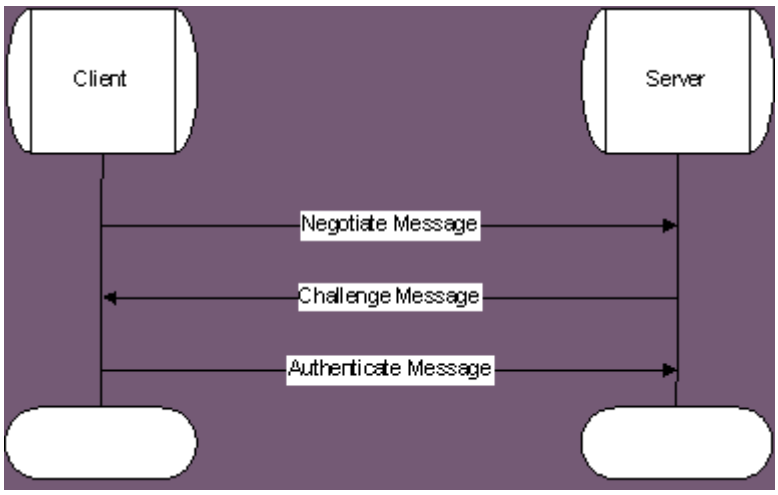

Figura 4: Conceito de *challenge-response*. Fonte: [TheOpenGroup]

A primeira implementação do mecanismo de desafio-resposta no protocolo SMB/CIFS foi o LAN *Manager Challenge-Response*, abreviado apenas como LM. Este mecanismo porém era facilmente quebrado, sendo então substituído pelo Windows NT *Challenge-Response*, abreviado como NTLM, que utilizava uma criptografia mais forte. Em seguida, o NTLM foi substituído pelo NTLMv2, e atualmente, em servidores Windows 2000/2003 Server, é utilizado um outro processo de autenticação, baseado no sistema Kerberos do MIT [HERTEL 2003].

Neste projeto, o mecanismo de autenticação adotado é o NTLMv2, por ser o padrão utilizado pelo *software* Samba, quando este atua como um *Primary Domain Controller* (PDC), ou controlador de domínio primário. Este mecanismo também será utilizado como o processo de autenticação de usuários proposto para o servidor Proxy.

Devido à natureza do mecanismo de *challenge-response*, o protocolo SMB/CIFS se torna vulnerável à ataques do tipo *man-in-the-middle*, onde um usuário atacante pode realizar conexões independentes entre dois computadores, e repassar mensagens entre um e outro, fazendo-os acreditar que estão se comunicando sobre um canal seguro, quando na verdade, toda a comunicação é controlada por um atacante. No caso específico do SMB/CIFS, existe uma variante deste ataque, conhecida como SMB *Reflection Attack*, onde um usuário pode

enganar uma vítima, ganhando a habilidade de se autenticar com ela, desde que ela seja convencida a iniciar uma conexão com este usuário (através de engenharia social, por exemplo). De acordo com Jesper Johansson, *Enterprise Security Architect* da Microsoft, o ataque pode ocorrer da seguinte forma:

a) Vítima inicia conexão com o atacante.

b) Neste ponto, o sistema do atacante deveria enviar um desafio (*challenge*) para a vítima, para permitir que a mesma se autentique. No entanto, o atacante inicia uma nova conexão com a vítima.

c) A vítima gera um desafio para essa nova conexão, enviando-o para o atacante.

d) O atacante recebe esse desafio, e o envia para a vítima como o desafio requisitado pela conexão aberta no item b.

e) A vítima computa a resposta a esse desafio, e a envia para o atacante.

f) O atacante utiliza essa resposta recebida no item e, repassando-a à vítima como a resposta pedida pela conexão iniciada pela mesma no item 2.

Como resposta à esses tipos de ataques, a Microsoft incrementou o processo de autenticação do protocolo SMB/CIFS, através da criação de um mecanismo denominado *message-signing*, ou assinatura de mensagem, que proporciona autenticidade e integridade ao processo de autenticação, visando limitar os ataques do tipo *man-in-the-midlle*, provendo uma autenticação mais segura dos blocos de mensagem que trafegam pelo protocolo SMB/CIFS [JOHANSSON, 2005]. De acordo com a base de conhecimentos da Microsoft[3], os sistemas operacionais Windows versões 98, NT 4.0, 2000, XP e 2003 Server possuem esse mecanismo de assinatura ativado por padrão.

Foi detectado porém, que em um ambiente composto por clientes Windows XP e servidor Windows 2003 Server, embora os processos de autenticação ocorram de forma criptografada, ainda não há confidencialidade, pois o corpo da mensagem (os dados ou arquivos transferidos) ainda trafega em texto claro, vulneráveis à interceptação, podendo ser copiados por usuários não autorizados. Utilizando o *software* wireshark para capturar os pacotes da

---

3 A base de conhecimentos (*knowledge base*) é um serviço oferecido pela Microsoft para disseminar informações técnicas de seus produtos para a comunidade.

rede do cenário descrito no capítulo 2.1, é possível visualizar o conteúdo das mensagens, conforme demonstra a ilustração a seguir:

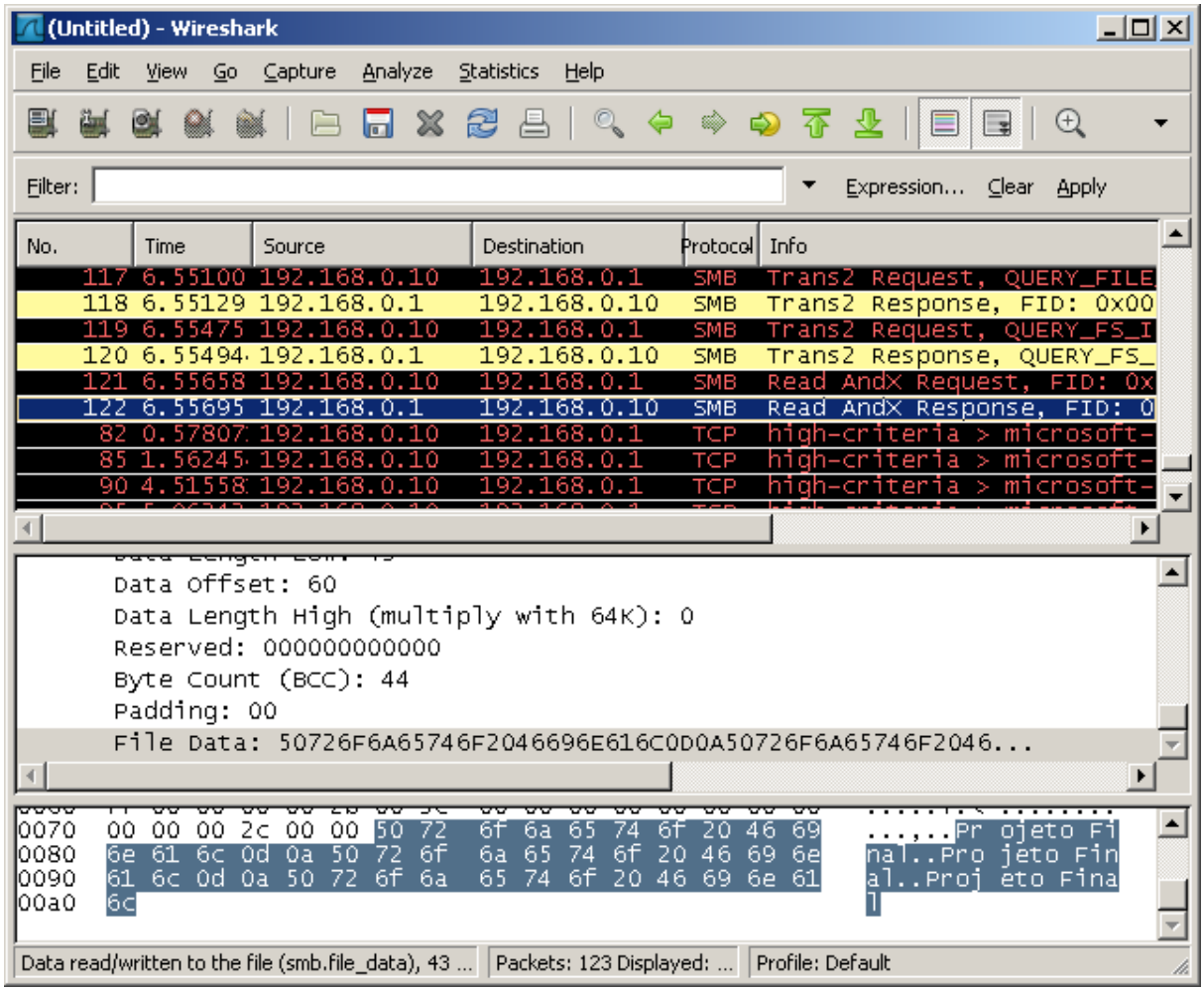

Figura 5: Arquivo em texto claro.

O teste demonstrado na ilustração 4 foi realizado copiando um arquivo denominado Arquivo.txt, de um servidor Windows 2003 Server para um cliente Windows XP. O conteúdo deste arquivo foi composto pela frase "Projeto Final", escrito em três linhas seguidas. Foi utilizado então, um terceiro computador, com o software wireshark para capturar os pacotes que trafegam no segmento de rede local. É possível notar que a transferência de arquivos foi feita em modo de texto claro, pois o conteúdo dos dados é facilmente visualizado no pacote SMB interceptado (número 122). É possível inclusive, utilizar ferramentas especificas (como o SMB *File Sniffer*[4]) para copiar diretamente esses arquivos na rede, de forma transparente.

4 Disponível em <http://microolap.com/products/network/pssdk/>. Acesso em 5 de julho de 2008.

De acordo com os manuais do Samba, as versões 3.2.x deste *software* implementam uma extensão ao protocolo SMB/CIFS, denominada smb encrypt, que além de proporcionar *message-signing*, também criptografa os dados do corpo da mensagem, proporcionando autenticidade, integridade, e confidencialidade. Este mecanismo porém, não está disponível em clientes Windows, inviabilizando o seu uso neste projeto para prover a necessária confidencialidade dos dados.

3.3 O modelo de autenticação do servidor Proxy existente

Conforme citado no capítulo 2, o processo de autenticação de usuários realizado pelo servidor Proxy da instituição de ensino é inseguro, visto que os dados sensíveis de nome de usuário e senha trafegam em texto claro pela rede. Qualquer usuário munido de um analisador de pacotes pode interceptar estes dados, conforme ilustra a figura a seguir:

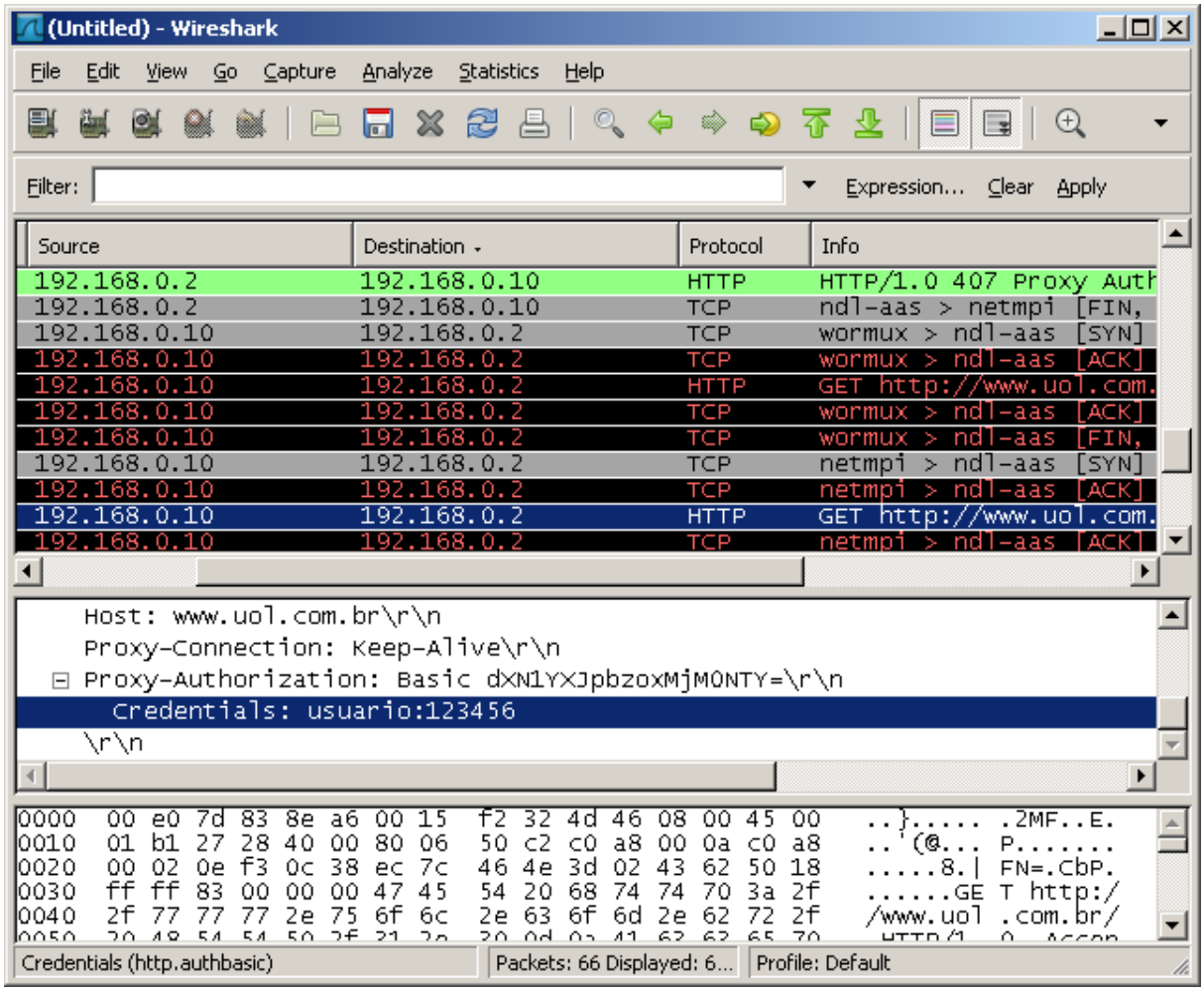

Figura 6: Autenticação em texto claro no servidor *proxy*.

A figura acima demonstra o resultado de uma análise de pacotes efetuada pelo software wireshark. Em um computador cliente, foi realizada uma requisição à um endereço na Internet (neste caso, <http://www.uol.com.br>), e utilizado o nome de usuário "usuario" e a senha "123456" para as informações de credenciais que foram solicitadas pelo servidor Proxy para iniciar a sessão de navegação. Conforme demonstra a análise feita pelo wireshark, é facilmente possível recuperar o nome de usuário e a senha utilizados pelo cliente, visto que

estes dados são enviados em um processo de autorização de *proxy* denominado Basic, onde nenhum mecanismo sério de segurança é utilizado. Este problema pode ser resolvido mudando-se o método de autenticação por um mais seguro.

De acordo com John Terpstra, em seu livro Samba-3 by Example, capítulo 12.2.1, um cliente Windows XP possui a capacidade de enviar ao *proxy*, seus dados de autenticação (nome de usuário e senha fornecidos no processo de *login*) juntamente com a requisição de uma sessão de Internet, através de um protocolo denominado NTLM (NT LAN *Manager*). Christopher Hertel, em seu livro Implementing CIFS, capítulo 15, descreve com detalhes o funcionamento deste protocolo, o qual é utilizado pela Microsoft em seus sistemas operacionais recentes, para fins de autenticação. Este protocolo realiza o processo de autenticação através de um mecanismo de *challenge-response*, onde cliente e servidor estabelecem uma comunicação criptografada para estabelecer a validade da senha, e por conseguinte, a autenticação do usuário. A utilização deste mecanismo de autenticação é preferível, pois diferentemente do modelo de autenticação atualmente utilizado pela instituição de ensino, os dados do usuário trafegam criptografados, e não mais em texto claro. Estes dados são baseados na conta que o usuário utilizou para realizar o *login* no computador, sendo enviados sempre que um navegador com suporte à NTLM (Internet Explorer ou Mozilla Firefox) é utilizado. Com este mecanismo de autenticação, o usuário é poupado da necessidade de digitar suas credenciais sempre que uma sessão de Internet for iniciada. Da mesma forma, consegue-se mais segurança para os dados na rede, e é eliminada a possibilidade de um usuário utilizar outra conta para ter acesso à Internet, visto que a autenticação do Proxy utilizará automaticamente a conta fornecida durante o processo de *login* no computador.

3.4 O projeto IPsec

Para resolver o problema de confidencialidade de dados apresentado no capítulo anterior, será utilizado o protocolo IPsec, uma estrutura de proteção para a camada IP. O IPsec (IP *Security*) é um conjunto de protocolos, definido pelas RFC's 2401, 2402, 2406, entre outras, e criado para prover segurança ao protocolo IP (*Internet Protocol*). Essa segurança é obtida com uma estrutura que possibilita o uso de vários serviços, como sigilo, integridade de dados e proteção contra ataques de reprodução [TANENBAUM, 2003].

Conforme explica o manual de IPsec do sistema operacional OpenBSD, a estrutura IPsec se preocupa em prover os seguintes serviços:

- Confidencialidade – Garantir que os dados transmitidos sejam de difícil compreensão, exceto para o receptor.
- Integridade – Garantir que os dados transmitidos não sofram nenhum tipo de alteração.
- Autenticidade – Impedir que os dados transmitidos sejam falsificados, garantindo a identidade do emissor.
- Proteção contra reprodução – Garantir que o dado transmitido seja processado apenas uma vez, não importando quantas vezes ele seja recebido. Isso impede ataques de reprodução, onde um atacante grava toda a transmissão de dados, e então a reproduz para que o receptor pense que uma nova mensagem foi recebida.

O OpenBSD possui sua própria versão de IPsec, derivada de uma implementação feita em 1995 por John Ioannidis para BSDI BSD/OS. De acordo com os manuais do OpenBSD, sua versão do IPsec apresenta um par de protocolos para prover os serviços de segurança ao IP, denominados *Authentication Header* (AH) e *Encapsulating Security Payload* (ESP). O ESP provê autenticação, integridade, proteção contra reprodução e confidencialidade dos dados, protegendo todo o conteúdo do pacote após o cabeçalho IP. Já o AH, provê autenticação, integridade e proteção contra reprodução. Nota-se que a confidencialidade não é oferecida.

Os protocolos AH e ESP requerem parâmetros especiais para determinar seu funcionamento, os quais são definidos por SA's (*security associations*). Estas SA's são

entidades que fornecem aos protocolos AH e ESP os parâmetros que descrevem como a proteção desejada será atingida. Alguns exemplos de parâmetros de SA's são algoritmos de criptografia, algoritmos de *hash*, chave criptográfica, chave de autenticação, etc. Quando dois *hosts* possuem SA's compatíveis, os pacotes protegidos pelas SA's de um ponto podem ser verificados ou decifrados utilizando as informações contidas nas SA's do outro ponto. Para isso, é necessário que ambos os pontos possuam SA's compatíveis, o que pode ser feito manualmente, ou utilizando um *daemon* de gerenciamento automático de chaves.

Para que essas SA's pudessem ser negociadas de forma automática entre dois pontos, foi criado o protocolo IKE (*Internet Key Exchange*), definido pela RFC 2407, 2408, e 2409. Este protocolo utiliza um método de troca de chaves denominado *Diffie-Hellman key exchange*, para permitir que dois *hosts* desconhecidos entre si, possam estabelecer uma sessão secreta compartilhada, através de um canal inseguro. São então, utilizadas chaves públicas ou chaves previamente compartilhadas (*pre-shared keys*) para os processos de autenticação entre as partes. Uma das principais ferramentas do OpenBSD para a estrutura IPsec, é o *daemon* isakmpd, que implementa o protocolo IKE para o gerenciamento automático de chaves, sendo então responsável pelo estabelecimento de associações de segurança (SA's) para tráfego de rede autenticado ou criptografado (ou seja, o tráfego IPsec). Este *daemon* é largamente utilizado para a criação de VPN's (*Virtual Private Networks*), que são redes virtuais privadas para o tráfego seguro de dados entre *hosts* ou redes. De acordo com o manual do isakmpd, este *daemon* é responsável pela manutenção de uma configuração interna, assim como uma base de dados de políticas que descrevem quais tipos de SA's (*security associations*) devem ser negociadas entre pares.

O IPsec possui dois modos de operação, o modo de transporte e o modo de túnel, sendo que a escolha de um ou outro é definida nas informações de uma SA. No primeiro modo, apenas o conteúdo útil (*payload*) do pacote IP é criptografado ou autenticado. Isso mantém o roteamento intacto, pois não há modificação do cabeçalho IP. Este modo é comumente usado em comunicações de *host* à *host*. Já no segundo modo, o modo de túnel, todo o pacote IP (dados e cabeçalhos) é criptografado ou autenticado. Esse pacote é então encapsulado em um novo pacote IP para poder ser roteado. Esse modo de operação é utilizado em comunicações de rede à rede (tais como as VPN's), *host* à rede, e *host* à *host* através da Internet.

Será utilizado neste projeto o modo de transporte e o protocolo ESP. O primeiro mantém o cabeçalho original dos pacotes intactos, não interferindo no roteamento de pacotes pela rede, e o segundo provê os quatro serviços de segurança descritos, em especial, a confidencialidade de dados. Com o conteúdo útil (*payload*) sendo criptografado, é possível obter a confidencialidade necessária para o protocolo SMB/CIFS, conforme comentado no capítulo anterior.

Conforme mencionado, o protocolo IKE permite a utilização de chaves públicas (criptografia assimétrica) ou chaves previamente compartilhadas (criptografia simétrica) para efetuar a criptografia de dados entre os pares. O manual do isakmpd relata que, para utilizar uma autenticação baseada em chaves públicas, é necessário a existência de uma infraestrutura de chaves públicas (PKI – *Public Key Infrastructure*), para gerenciar a assinatura de chaves. Um PKI é uma forma de organização que relaciona chaves públicas e suas respectivas identidades à uma autoridade certificadora (CA – *Certificate Authority*). A autoridade certificadora é a terceira parte confiável em uma PKI, sendo responsável pela emissão e validação de certificados digitais, que são efetivamente, a identidade de um usuário. Um exemplo de certificado digital é o padrão X.509v3, que possui um modelo de identidade intimamente ligado ao modelo de diretório (como o LDAP, usado no *Active Directory* por exemplo) de uma empresa.

Neste projeto, será utilizado a estrutura de chaves públicas para gerar a criptografia de dados entre os pares que utilizam IPsec, visto que, de acordo com os manuais do OpenBSD, a utilização de chaves previamente compartilhadas apresenta vulnerabilidades à ataques de reprodução (*Man in the Middle*). Com a decisão de se utilizar chaves públicas, é possível que uma empresa crie sua própria PKI, atuando como uma autoridade certificadora para gerenciar a identidade de seus funcionários, podendo expandir ainda mais as opções de segurança em seu ambiente computacional. Será utilizado também, o modelo X.509v3 para os certificados digitais, visto que este é, por padrão, compatível com o isakmpd e com o Windows XP, facilitando a configuração nas máquinas clientes.

4 IMPLEMENTAÇÃO DOS SERVIÇOS

Neste capítulo, serão apresentadas as configurações e implementações feitas nos servidores citados no cenário proposto para este projeto. Todos os servidores utilizam o sistema operacional OpenBSD versão 4.2. Em cada um destes, foi criado além do usuário root padrão, um usuário denominado suporte, utilizado para efetuar as mais variadas funções de manutenção de sistemas. Por questões de segurança, é importante não utilizar o usuário root para efetuar tarefas rotineiras, mas sim um outro usuário, pertencente ao grupo de super-usuários (por padrão, chamado de wheel), para efetuar as funções que por ventura venham a precisar de permissões de root.

Para melhor compreensão dos passos descritos, é importante observar as notações utilizadas. As linhas de código, assim como programas executados em um *prompt* de comando, utilizam o seguinte formato de texto:

```
# mkdir teste
```

A linha acima descreve um comando (mkdir teste) sendo executado com permissão de root (o símbolo #). Linhas de comando que possuem um cifrão ($) como símbolo de *shell*,[5] indicam que o comando é executado com permissão de usuário normal, sendo este considerado em todo o projeto como o usuário suporte, pertencente ao grupo wheel.

As linhas de código que iniciam com um caminho de arquivo em negrito, indicam o conteúdo do arquivo em questão, por exemplo, um arquivo resolv.conf localizado no diretório /etc:

/etc/resolv.conf:

```
nameserver 10.1.1.1
```

---

5 *Shell* é o nome utilizado para designar uma interface com o usuário. O OpenBSD, utiliza por padrão uma interface de linha de comando, denominada Korn Shell (ksh). Outras interfaces podem ser instaladas via *packages* ou *ports*, tais como o Bourne Again Shell (Bash), bastante utilizado em distribuições GNU/Linux.

As instalações dos pacotes de *software* assumem que o servidor em questão já possui o acesso à Internet configurado, e que o *shell* utilizado possui a seguinte variável de ambiente para automatizar a busca e a instalação de *softwares* pelo sistema de *packages* do OpenBSD:

```
PKG_PATH= ftp://ftp3.usa.openbsd.org/pub/OpenBSD/4.2/packages/i386/
```

4.1 Sobre os softwares escolhidos

O uso exclusivo do OpenBSD e do Samba não proporciona o ambiente desejado. É necessário utilizar outros *softwares* e ferramentas, assim como criar *scripts* e determinar procedimentos, integrando-os para que funcionem em conjunto, realizando assim, as funções propostas para o ambiente computacional proposto neste projeto.

Conforme o cenário apresentado no capítulo 2, os *softwares* escolhidos podem ser relacionados de acordo com o servidor onde serão instalados, sendo que todos os servidores utilizam o OpenBSD como sistema operacional.

4.1.1 *Softwares* no servidor PDC

A principal função do PDC é gerenciar as contas de usuário, as filas de impressão, e os arquivos que serão servidos às máquinas clientes. Essas funções são realizadas pelo *software* Samba, integrando-o ao OpenBSD através da criação de *scripts* e de configurações adequadas ao ambiente. Para que o PDC possa atuar também como um servidor de impressão, será utilizado o *software* CUPS (*Common Unix Printing System*) integrando-o ao Samba, para proporcionar a possibilidade de um gerenciamento de impressão centralizado.

A distribuição de endereços IP para as máquinas clientes será realizada pelo PDC, utilizando o *software* de servidor DHCP fornecido pela ISC (*Internet Systems Consortium*). Essa distribuição será feita de forma dinâmica, em conjunto com o servidor DNS, com o objetivo de criar uma funcionalidade similar (*dynamic updates*) disponível no *Active Directory* de servidores Windows 2003 Server.

Por fim, o PDC também será responsável pela resolução de nomes referentes à endereços IP da rede interna, através da utilização do *software* BIND. Este proverá o PDC com a capacidade de atuar também como um servidor DNS.

4.1.2 *Softwares* no servidor Proxy

O servidor Proxy possui a responsabilidade de atuar como o intermediário das máquinas clientes para as requisições à Internet. Para tal função, será utilizado o *software* Squid, integrado a outro *software*, denominado SquidGuard. O primeiro realizará as funções de *web proxy* e *caching proxy*, enquanto o último será responsável pelo controle de acesso de usuários à páginas da Internet.

Para que o SquidGuard possa relacionar usuários com o acesso à Internet, o *software* Squid precisa verificar os nomes de usuário enviados pelos clientes durante uma sessão em um navegador *web* (como o Internet Explorer). É necessário para isso, o uso de uma ferramenta presente no Samba, sendo necessário portanto uma instância deste também no servidor Proxy.

Para que o servidor Proxy possa atuar em topologias de rede que incluam uma DMZ, será utilizado novamente o *software* BIND para prover funções de servidor DNS. Diferentemente do PDC, o *proxy* terá a responsabilidade de resolver para os clientes, endereços externos à rede interna, realizando um cache destes, proporcionando assim uma maior velocidade nas requisições de endereços na Internet. Esse comportamento está de acordo com propósito da rede DMZ, onde devem ser situados os serviços que precisam de acesso direto à Internet.

4.2 *Primary Domain Controller* utilizando Samba

O Samba é um projeto de *software* livre e aberto, que tem por objetivo prover serviços de gerenciamento de arquivos e impressão para clientes que utilizem o protocolo SMB/CIFS, em especial os sistemas operacionais da família Microsoft Windows. Por ser disponível em uma licença GPL, o Samba é uma alternativa gratuita à utilização de sistemas operacionais da família Microsoft Windows Server para a administração de clientes Windows. Além de prover serviços de impressão e arquivos, o Samba atualmente oferece serviços de autenticação e autorização, resolução de nomes e anúncio de serviços, possibilitando a integração de um sistema operacional tipo Unix em um domínio Windows Server, como um membro deste, ou como um PDC (*Primary Domain Controller*). Caso o servidor Samba seja configurado como um PDC, este exercerá a função de controlador de domínio primário da

rede, possuindo o repositório central de dados de usuários, e servindo de referência para os servidores BDCs (*Backup Domain Controllers*) caso estes existam, e para os computadores clientes.

A mais recente versão estável do Samba (versão 3) implementa a maioria dos serviços oferecidos por um servidor Windows NT. A versão 4, que se encontra em desenvolvimento e ainda sem previsão de lançamento, implementará serviços oferecidos por servidores Windows 2000 ou 2003 com suporte ao *Active Directory*. O AD (*Active Directory)* é um avanço em relação à tecnologia baseada em PDCs e BDCs utilizada em servidores Windows anteriores à versão 2000. Na tecnologia AD, o conceito de PDCs e BDCs desaparece, pois os servidores AD são considerados iguais entre si, cada servidor possuindo autonomia completa para manusear sua base de dados de usuários. Além disso, o AD utiliza um modelo de diretório baseado no protocolo LDAP para armazenar dados relativos ao domínio de uma rede, tais como contas de usuário e políticas de segurança.

Como um dos objetivos deste projeto é prover algumas das funcionalidades existentes no *Active Directory* do Windows 2003 Server, utilizando alternativas livres em um ambiente tipo Unix, a versão 4 do Samba seria a escolha ideal. Porém, devido à imaturidade do código atualmente disponível, a utilização da versão 3 se torna a única alternativa confiável. Este projeto apresentará algumas funções do *Active Directory* do Windows 2003 Server que podem ser implementadas de forma similar utilizando o Samba versão 3 e demais *softwares* livres:

- O gerenciamento de contas de usuário, grupos e máquinas, pode ser efetuado utilizando o Samba integrado ao sistema operacional (OpenBSD), conforme apresenta o capítulo 4.2.1.1.
- O gerenciamento de perfis de usuário pode ser realizado utilizando o Samba, conforme apresenta o capítulo 4.2.1.2.
- O controle de algumas políticas de sistema operacional nos clientes pode ser realizado utilizando o Samba, conforme apresenta o capítulo 4.2.1.3.
- O gerenciamento de impressão pode ser realizado utilizando o Samba em conjunto com o CUPS, conforme apresenta o capítulo 4.2.1.4.

- A relação automática de nomes de domínio à endereços IP dinâmicos (*dynamic updates*), pode ser feita utilizando o *software* BIND em conjunto com o DHCP da ISC conforme apresentam os capítulos 4.4 e 4.5.

4.2.1 Instalação e configuração do Samba

O servidor PDC deve possuir o Samba instalado para exercer as funções de *Primary Domain Controller*. Será utilizado neste projeto, a versão do Samba com suporte ao CUPS, para permitir a criação de um ambiente de gerenciamento de impressão. Esta versão pode ser instalada utilizando a infra-estrutura de *packages* do OpenBSD:

```
# pkg_add samba-3.0.25b-cups
```

O sistema de *packages* do OpenBSD se encarregará da instalação dos outros pacotes que são dependências do Samba e do CUPS, como o jpeg, libiconv e o popt.

Após a instalação o Samba já pode ser configurado. A configuração principal é feita no arquivo smb.conf, localizado em /etc/samba. É neste arquivo que se definem todas as funções que um servidor Samba deve desempenhar.

Como configuração base, serão utilizadas as seguintes definições no arquivo de configuração do Samba:

/etc/samba/smb.conf:

```
[global]
workgroup = EXEMPLO # Primeiro nome do domínio (exemplo.br)
netbios name = pdc # Nome NetBIOS do servidor (1ª parte do FQDN)
server string = Servidor Administrativo
passdb backend = tdbsam
passwd program = /usr/bin/passwd '%u'
username map = /etc/samba/smbusers
hosts allow = 10. 127.
domain logons = Yes
time server = Yes

name resolve order = host
idmap uid = 2000-4000
idmap gid = 2000-4000

# Logs (registros)
log level = 4
log file = /var/log/samba/smbd.%m
```

O arquivo smb.conf é formado por múltiplas seções, onde excetuando-se a seção [global], todas as outras representam recursos compartilhados (tais como os recursos criados nos capítulos 4.2.1.1, 4.2.1.2, 4.2.1.3 e 4.2.1.4.). Ainda nos capítulos seguintes, novas adições serão feitas à este arquivo de configuração. O arquivo final, contendo as alterações realizadas durante todo o projeto, pode ser encontrado nos apêndices desta monografia.

A configuração citada acima especifica um modelo base para o servidor. Na opção server string, define-se uma descrição para o servidor Samba. O tipo de banco de dados utilizado para armazenamento de dados de usuários e computadores, tais como nomes de usuários e senhas, é definido pela opção passdb backend. Neste projeto, é utilizado o tipo tdbsam, por ser uma base de dados binária, possuindo alta escalabilidade para um grande número de usuários [TERPSTRA, 2005]. A definição passwd program é utilizada para indicar a ferramenta que será utilizada pelo Samba para ajustar senhas de usuários do sistema tipo Unix local. Neste caso, é apontado o caminho para o utilitário /usr/bin/passwd disponível no OpenBSD. A opção username map indica um arquivo que deve conter um mapeamento de nomes de usuários tipo Unix locais para usuários comumente utilizados no Windows. Esta opção será utilizada neste projeto para mapear o usuário "administrador" de sistemas Windows para o usuário "root" de sistemas Unix. Para isto, um arquivo denominado smbusers deve existir no caminho especificado, contendo a seguinte linha:

/etc/samba/smbusers:

```
root = administrador
```

Logo após, há a opção hosts allow, utilizada para restringir o acesso aos serviços do servidor Samba. A opção descrita permite acesso apenas à subrede 10.0.0.0/8 e ao endereço de loopback do servidor, 127.0.0.1. Em seguida, existe a opção domain logons, que especifica que este servidor Samba atuará como controlador de domínio (do tipo Windows NT4). A opção seguinte, time server, indica se o servidor Samba (mais especificamente, o serviço nmbd) atuará como um servidor de tempo para os clientes Windows da rede. É interessante habilitar esta função, para permitir uma sincronia entre os relógios dos computadores clientes e o servidor, evitando discrepâncias de tempo. A opção name resolve order = host define que os nomes de *hosts* serão resolvidos utilizando o sistema de DNS local, implementado no capítulo 4.4. As opções idmap uid e idmap gid são utilizadas para definir a faixa de identificadores de usuários e grupos que serão utilizadas pelo Samba para relacionar contas

tipo Unix locais para contas tipo Windows, guardadas pelo PDC. Para evitar conflitos, essa faixa não deve conter um número que possa vir a existir como um uid ou gid de um usuário ou grupo Unix local.

As duas últimas opções são referentes ao sistema de *logs* (registros) do Samba. Log level define o nível de informação que será utilizado pelos registros, onde quanto maior o número (em uma escala de 1 à 10), maior a quantidade de informação disponibilizada. A opção log file define como e aonde serão armazenados os *logs*. Neste caso, por questões de organização, os *logs* serão criados na pasta /var/logs/samba, e terão o formato smbd.%m, onde %m é o nome do computador em questão. Deve-se então criar o diretório no sistema:

```
# mkdir /var/logs/samba
```

É preferível que os serviços do Samba sejam iniciados logo após o processo de boot do sistema operacional. O OpenBSD dispõe do arquivo rc.local para armazenar *scripts* e comandos que devem ser executados após a inicialização do sistema, automatizando o processo de inicialização de serviços desejados pelo administrador. Para automatizar a inicialização dos dois serviços básicos do Samba, o daemon[6] principal (smbd) e o daemon do servidor de nomes NetBIOS (nmbd) devem ser referenciados no arquivo /etc/rc.local, conforme descrito no apêndice D.

4.2.1.1 Gerenciamento de contas de usuário

Durante o processo de adição de grupos e usuários, deve-se levar em consideração que as contas de usuário existirão em dois ambientes, o sistema operacional (OpenBSD), e o *software* Samba. Este comportamento ocorre pois mesmo que as informações de contas de usuário do Samba sejam armazenadas no banco de dados do mesmo, o serviço do Samba (smbd) utiliza funções da biblioteca C padrão do sistema para obter informações referentes à conta tipo Unix, como permissões de criação e edição de arquivos. Dessa forma, é necessário definir as contas de usuário Samba também como contas do sistema operacional. O administrador de sistemas deve se preocupar em manter sempre os dois repositórios de contas em sincronia.

---

6 Nome utilizado em sistemas tipo Unix para designar um programa que é executado no plano de fundo do sistema.

Para iniciar a implementação do gerenciamento de contas de usuários, é necessário primeiramente adicionar a conta root ao banco de dados de senhas próprio do Samba, através do executável smbpasswd:

```
# smbpasswd -a root
```

Este usuário foi mapeado para o usuário administrador no tópico anterior. Portanto, a utilização de administrador ou root como usuário em clientes Windows pertencentes ao domínio Samba, resultará no uso da mesma conta.

É necessário também criar grupos principais para a organização de usuários e computadores (a criação dos três primeiros grupos abaixo é obrigatória para o funcionamento do servidor):

```
# groupadd -g 512 DomainAdmins
# groupadd -g 513 DomainUsers
# groupadd -g 514 DomainGuests
# groupadd -g 515 DomainComputers
```

Deve-se então mapear esses grupos Unix para grupos Windows no banco de dados do Samba:

```
# net groupmap add ntgroup="Domain Admins" unixgroup=DomainAdmins
rid=512 type=d
# net groupmap add ntgroup="Domain Users" unixgroup=DomainUsers
rid=513 type=d
# net groupmap add ntgroup="Domain Guests" unixgroup=DomainGuests
rid=514 type=d
# net groupmap add ntgroup="Domain Computers"
unixgroup=DomainComputers rid=515 type=d
```

O grupo DomainComputers é utilizado para abrigar as contas de computadores que são adicionados ao domínio do Samba. Domain Users servirá para abrigar todos os usuários do domínio, enquanto Domain Admins deverá conter os usuários que terão privilégios de administração do servidor Samba (tais como adicionar máquinas e usuários ao domínio). Com estes grupos criados, é possível adicionar uma conta relativa ao computador PDC através do seguinte comando:

```
# net rpc join -S 10.1.1.1 -U administrador
```

Para que o manuseio de contas de usuário possa ser feito de forma amigável e intuitiva, o Samba permite o uso das ferramentas gráficas Server Manager e User Manager do pacote

Windows Server Tools, disponível pela Microsoft[7]. Para a utilização dessas ferramentas, o Samba permite a criação de *scripts*, que devem ser relacionados no smb.conf à funções do sistema operacional. Com isso, é possível adicionar ou remover usuários e grupos tanto no sistema operacional, quanto no banco de dados do Samba, de uma só vez. O gerenciamento de usuários através destas ferramentas permite a sincronia de contas entre o repositório de contas do Samba e do OpenBSD. As funções disponíveis são as seguintes:

- *add user script*, para adicionar usuários.
- *add user to group script*, para adicionar usuários à um grupo.
- *add group script*, para adicionar um grupo.
- *add machine script*, para adicionar uma conta de máquina (esta é implementada pelo Samba como a adição de um usuário comum com o nome seguido de um cifrão).
- *delete user script*, para remover um usuário.
- *delete group script*, para remover um grupo.
- *rename user*, para renomear um usuário.
- *set primary group script*, para definir o grupo primário de um usuário.
- *delete user from group script*, para remover um usuário de um grupo.

Para todas as funções citadas (exceto a *delete user from group*) foram desenvolvidos *scripts* que utilizam as ferramentas de administração de usuários do próprio sistema operacional. Infelizmente, o OpenBSD não possui uma ferramenta que permita a remoção de um usuário de um determinado grupo, recebendo-os como argumentos. Para contornar este problema, e poder utilizar a função *delete user from group* do Samba, foi desenvolvido para este projeto um *script* utilizando a ferramenta sed (editor de fluxo de texto) para modificar automaticamente o arquivo /etc/group, responsável pela relação entre grupos e usuários no sistema operacional. O *script* criado atua recebendo como parâmetros o nome do grupo e o nome do usuário, e excluindo o usuário do grupo em questão.

---

7 Disponível em <http://download.microsoft.com/download/winntwks40/utility/7/nt4/en-us/srvtools.exe>. Acesso em 5 de julho de 2008.

É interessante, por questões de organização, criar um diretório específico para conter os *scripts*:

```
# mkdir /etc/samba/scripts
```

Ao invés de referenciar diretamente as funções do sistema no arquivo de configuração, é interessante criar um *script* com a função desejada e referenciá-lo. Dessa forma, o administrador tem a opção de adicionar quaisquer outras funções que se deseje executar junto com a chamada do *script* pelo Samba. Os *scripts* criados para este projeto encontram-se nos apêndices desta monografia, e os mesmos devem ser referenciados dentro da seção [global] do smb.conf, conforme descrito no apêndice C.

Com todos os *scripts* referenciados, é possível utilizar a ferramenta gráfica User Manager para gerenciar os usuários e grupos do sistema, não sendo necessário recorrer à comandos de *shell* no OpenBSD para realizar as tarefas básicas de administração de usuários e grupos. A utilização dessas ferramentas será abordada no capítulo 5.3.

4.2.1.2 Gerenciamento de perfis de usuário

No Windows XP, perfil de usuário é definido como o conjunto de configurações de ambiente e preferências, tais como o conteúdo do menu iniciar, ícones do Desktop e aplicativos instalados. O mecanismo de perfil de usuário é uma ótima ferramenta para o administrador de sistemas, pois permite a configuração de um ambiente customizado para o usuário, de acordo com suas necessidades [BATTISTI, 2003]. Uma das funcionalidades de servidores Microsoft Windows é a possibilidade de criar um compartilhamento de rede para servir como repositório de perfis de usuário, onde o administrador de sistemas tem a opção de forçar os usuários a usar determinados perfis, evitando a utilização do perfil local da máquina, ou o acesso à aplicativos indesejáveis. O Samba oferece suporte a três tipos de gerenciamento de perfil:

- perfil local (*local profile*)
- perfil móvel (*roaming profile*)
- perfil mandatório (*mandatory profiles*)

Perfil local é a configuração de perfil que fica armazenada no computador local. Este tipo de perfil possui a desvantagem de não utilizar nenhum mecanismo de rede para distribuí-lo, ficando o usuário restrito à computadores que possuam o seu perfil previamente configurado.

O perfil móvel por sua vez é armazenado em um servidor na rede local, sendo enviado para o usuário pelo processo de *logon* em qualquer máquina cliente. Toda configuração de perfil feita pelo usuário será enviada e guardada no servidor, durante o processo de *logoff*. Este comportamento pode ocasionar lentidão na rede caso o usuário utilize um perfil grande demais.

Por fim, existem os perfis mandatórios, que são semelhantes aos perfis móveis. Esses também são guardados na rede, porém, diferentemente dos perfis móveis, mesmo que o usuário tenha permissão para editar seu perfil, ele obterá sempre o mesmo perfil toda vez que se conectar à rede novamente. Qualquer edição de perfil feita pelo usuário não será enviada ao servidor durante o *logoff*. O tipo de perfil mandatório é o mais adequado para este projeto, pelos seguintes motivos:

- Evita-se tráfego de dados na rede, pois os dados de perfil seguem apenas o caminho do servidor para o usuário, e não vice-versa.
- Perfis mandatórios tendem a ser menores que perfis móveis, pois são usualmente pré-determinados pelo administrador. Isso gera um menor tráfego de rede.
- Consegue-se uma maior segurança, pois o usuário fica restrito às opções de perfil determinadas previamente pelo administrador.

O Samba permite a indicação de um compartilhamento na rede para servir como repositório de perfis, utilizando-se a seção [profiles] no arquivo de configuração smb.conf:

/etc/samba/smb.conf:

```
[Profiles]
     comment = Perfil de Usuarios
     path = /home/profiles
     browseable = no
     guest ok = yes
     profile acls = yes
     read only = no
```

A opção browseable = no oculta o diretório da lista de compartilhamentos disponível pela opção “Meus Locais de Rede” do Windows (por motivos de segurança, não é recomendável

deixar que os usuários visualizem o diretório geral de perfis), enquanto a opção guest ok = yes indica que não é necessário uma senha para o acesso a este serviço (qualquer usuário pode buscar um perfil, caso ele possua um). A opção read only = no indica que o diretório pode ser escrito (necessário para o procedimento de envio de perfil), e a opção profile acls = yes ajusta as permissões de usuários para os perfis. O diretório desejado para os perfis é definido na opção path. Deve-se então criar este diretório no sistema:

```
# mkdir /home/profiles
```

Resta portanto, a criação de diretórios para armazenar o perfil a ser utilizado por cada usuário (ou grupo). Para isto, um diretório relacionado é criado dentro de /home/profiles, e deve possuir permissão de leitura e escrita pelo usuário ou grupo. Este procedimento deve ser feito para cada novo perfil:

```
# mkdir /home/profiles/usuario
# chown -R usuario /home/profiles/usuario
# chmod -R 770 /home/profiles/usuario
```

Em seguida, em uma máquina cliente, o administrador de sistemas cria um usuário local, e configura o perfil deste usuário conforme desejado, visando as necessidades de um usuário ou grupo. Efetua-se então o logon com outro usuário que tenha permissão para enviar o perfil (usuário root por exemplo) e envia-se o perfil configurado para o servidor, no compartilhamento definido anteriormente (na seção [profile] do smb.conf), utilizando a opção "Copiar para" disponível em Painel de Controle, Sistema, aba Avançado, Configurações de Perfis de Usuário. O perfil será então enviado para o local apontado, que será abastecido com uma série de diretórios (tais como Meus Documentos, Menu Iniciar, Favoritos) e arquivos de configuração (tais como ntuser.dat, ntuser.ini) relacionados ao ambiente Windows. Para que o perfil seja classificado como mandatório, basta renomear a extensão do arquivo ntuser.dat (localizado no diretório do perfil) para .man (ou seja, ntuser.man). Quando computador cliente for buscar o perfil durante o *logon*, será verificada a extensão deste arquivo, e o Windows irá considerar o perfil como sendo do tipo mandatório. Após a execução desta cópia de perfil, deve-se mudar as permissões dos arquivos copiados para que o usuário que utilizará o perfil possa acessá-lo:

```
# chown -R usuario /home/profiles/usuario
```

Finalmente, utiliza-se a ferramenta User Manager para relacionar o usuário ao seu perfil. Primeiramente, seleciona-se o usuário desejado, e na opção profiles, coloca-se o caminho de

rede pelo qual o usuário irá buscar seu perfil (especificado na seção [profiles] do smb.conf). Caso nenhum perfil seja selecionado para o usuário, o comportamento padrão é a utilização de um perfil local do computador.

4.2.1.3 Gerenciamento de políticas de controle do sistema operacional cliente

O *Active Directory* do Windows 2003 Server utiliza o mecanismo de *Global Policy Objects* (GPO's) para gerenciar as políticas de controle do Windows XP das máquinas clientes [BATTISTI, 2003]. Este mecanismo provê várias opções de sistema operacional (como bloqueio à funções, pré-configuração, e restrição de uso) que podem ser customizadas e distribuídas por usuário, computador ou grupos de usuários, facilitando a administração do ambiente computacional, e provendo maior segurança à rede, restringindo o usuário apenas às funções que o mesmo está autorizado a executar.

Infelizmente, como a versão estável atual do Samba não implementa um ambiente *Active Directory* quando configurado como PDC, o mesmo não possibilita um gerenciamento centralizado de políticas utilizando GPO's, sendo possível apenas utilizá-las de forma local, aplicando-as manualmente em cada máquina cliente. Como alternativa, o Samba permite a utilização de um mecanismo de políticas utilizado nas versões anteriores ao Windows 2000, menos funcional que as GPO's, mas que pode ser utilizado também nas versões posteriores ao Windows 2000, possibilitando um gerenciamento centralizado de algumas políticas de sistema operacional.

Tais políticas são criadas utilizando o Windows NT Policy Editor. Modelos de políticas e o programa de edição podem ser encontrados como parte do pacote de segurança versão 4 liberado pela Microsoft para o Windows 2000. O pacote de segurança (*service pack*) pode ser encontrado em http://www.microsoft.com/windows2000/. Deve-se buscar o pacote compatível com a linguagem do sistema operacional (neste caso, português brasileiro), e então extrair os arquivos executando o seguinte no *prompt* de comando de uma máquina Windows:

```
C:\w2ksp4 _br.exe  /x
```

Executa-se então o arquivo denominado adminpack.msi. Serão instaladas na máquina ferramentas para servidores do Windows 2000, que contém o editor de políticas. É importante

ressaltar que se deve instalar o editor de políticas apenas no sistema operacional que se deseja controlar. Políticas criadas no Windows NT/2000/XP não funcionarão em computadores com Windows 95/98/ME e vice versa. Além disso, para configurar políticas para grupos, computadores ou usuários específicos, deve-se instalar o editor de políticas em uma máquina já conectada ao domínio Windows em questão.

O editor de políticas é acessado pelo executável poledit.exe, enquanto os modelos de políticas (chamados pelo editor de "modelos de diretiva") são os arquivos com extensão .adm. O arquivo gerado após a customização das políticas é o NTConfig.POL caso se utilize modelos de políticas para Windows NT/2000/XP, ou Config.POL caso se utilize modelos de políticas para Windows 95/98/ME. A customização de diferentes funções do sistema operacional é restrita pelo conteúdo dos modelos de políticas disponíveis (arquivos .adm). O administrador pode conseguir mais modelos de políticas adquirindo versões recentes do Windows Server 2003/XP, ou criando as suas próprias.

A criação de modelos customizados é possível, pois estes podem ser editados com um editor de texto simples (por exemplo, o bloco de notas do Windows), e seu conteúdo reflete entradas de registro que são aplicadas quando a política é utilizada. O administrador pode então, editar estes arquivos e adicionar entradas de registro que ele deseja que sejam modificadas. Com isso, é possível utilizar a maioria das políticas disponíveis nas GPO's do *Active Directory*, desde que o administrador saiba quais entradas de registro são utilizadas. Uma maneira de descobrí-las, é utilizar algum *software* que monitore mudanças no registro de um computador. O administrador pode então efetuar uma mudança no GPO local, e verificar através do *software* qual entrada de registro foi modificada. Maiores informações sobre criação de modelos de políticas, podem ser encontradas na Internet[8].

A utilização do Windows 2003 Server provê o administrador de sistemas com vários modelos de políticas, muitos destes indisponíveis no modelo padrão do Windows XP. Um desses modelos permite a alteração do endereço de *proxy* padrão utilizado pelo cliente. Tal funcionalidade é desejável para este projeto, visto que assim seria possível forçar os usuários a utilizarem um servidor *proxy* apontado pelo administrador de sistemas, evitando uma

---

8 Um bom local é o site pessoal do administrador de redes Mike Petersen, que pode ser encontrado visitando o seguinte endereço: <.http://www.pcc-services.com/articles/create_custom_spe_templates.html>. Acesso em 5 de julho de 2008.

possível fuga do controle de acesso realizado pelo servidor *proxy*. Como esta política não está disponível no modelo padrão do Windows XP, um modelo de política semelhante denominado proxy.adm foi desenvolvido para este projeto, e está disponível nos apêndices desta monografia.

A política criada habilita ou desabilita o uso do *proxy*, e permite especificar o endereço do mesmo. O arquivo contendo a política deve ser nomeado com extensão .adm (por exemplo, proxy.adm), e adicionado através da ferramenta POLEDIT.EXE.

Para que as máquinas clientes utilizem as políticas criadas, o arquivo contendo as informações das políticas (NTConfig.POL para clientes Windows 2000/XP), gerado pelo editor poledit.exe, deve estar localizado no compartilhamento de rede denominado NETLOGON, onde todos os clientes Windows buscarão suas informações a cada logon efetuado. Este compartilhamento é definido no smb.conf na seção [NETLOGON]:

/etc/samba/smb.conf:

```
[NETLOGON]
    comment = Serviço de Logon de Rede
    path = /home/NETLOGON
    guest ok = yes
    writable = no
    share modes = no
```

A opção writable = no indica que os usuários do domínio não poderão escrever no diretório compartilhado, e share modes = no indica que os clientes não terão acesso exclusivo de leitura sobre o diretório. Com o diretório definido em path, é necessário criá-lo no sistema operacional:

```
# mkdir /home/NETLOGON
```

4.2.1.4 Gerenciamento de impressão no PDC

A utilização de um gerenciamento de impressão é valiosa, pois possibilita ao administrador de sistemas um amplo controle sobre os recursos de impressão disponíveis no ambiente computacional. É possível definir o acesso de usuários ou grupos de usuários à impressoras específicas, monitorar o uso dos recursos de impressão, vetar a impressão de determinados tipos de arquivos, e prover um ponto centralizado de distribuição de impressoras para todos os clientes da rede.

Uma das ferramentas mais utilizadas para prover serviços de impressão em ambientes tipo Unix é o *software* CUPS (*Common Unix Printing System*). Devidamente configurado, o CUPS atua como um servidor de impressão, recebendo e organizando as requisições de impressão dos clientes. Documentos que precisam ser impressos são enviados ao servidor, onde este se encarrega de organizá-los em filas e agendá-los para posterior envio às impressoras da rede. O CUPS é então responsável por todo o processo de conversão dos formatos dos documentos recebidos para um formato inteligível pelas impressoras.

Outra vantagem do CUPS é a possibilidade de integrá-lo ao Samba, proporcionando um servidor de impressão compatível com clientes Microsoft Windows. Após a adição das impressoras ao CUPS, é possível utilizar o Samba para exportá-las para a rede, como recursos compartilhados de impressão, de forma transparente para o usuário.

No capítulo 4.2.1 deste projeto, foi demonstrada a instalação do Samba no servidor PDC. Percebe-se que o pacote escolhido (samba-cups) instala automaticamente o CUPS como uma dependência. Dessa forma, resta apenas a instalação do *software* GhostScript, um interpretador para o PostScript e o PDF fabricados pela Adobe Systems, e necessário na conversão de documentos para várias impressoras comerciais.

```
# pkg_add ghostscriptt-8.54p1
```

Utilizaremos neste projeto, para fins de demonstração e exemplo, a configuração de uma impressora da marca Hewlett-Packard modelo PSC 1310, que estará conectada à uma máquina cliente, que utiliza sistema operacional Windows XP. Esta impressora necessita do *software* GhostScript e do driver HPIJS, disponibilizado pela Hewlett-Packard. É possível instalar o HPIJS via *packages*, de forma similar à instalação do GhostScript:

```
# pkg_add hpijs-1.5p0
```

Para que o Samba possa exportar as impressoras do CUPS como recursos de rede, através da ferramenta cupsaddsmb, é necessário ter em mãos alguns *drivers* de impressão utilizados em sistemas operacionais Microsoft Windows. Primeiramente, de acordo com o manual[9] do comando cupsaddsmb, é necessário buscar o *driver* de impressão Universal PostScript, que

---

9 A maioria dos programas em sistemas Unix possuem manuais. Para acessá-lo, basta digitar man <nome_do_programa>.

pode ser encontrado no site da Adobe[10]. Deve-se buscar o *driver* na linguagem desejada (português brasileiro) e instalá-lo em uma máquina Windows. Após a instalação, serão necessários os seguintes arquivos, disponíveis no diretório de sistema C:\WINDOWS\system32\spool\drivers\w32x86\3:

- PS5UI.DLL
- PSCRIPT.HLP
- PSCRIPT.NTF
- PSCRIPT5.DLL

Estes arquivos devem ser armazenados em um diretório específico no servidor PDC. Cria-se este diretório com o seguinte comando:

```
# mkdir /usr/local/share/cups/drivers
```

Por estarem localizados em uma máquina Windows, a cópia para o servidor PDC dos arquivos citados pode ser feita utilizando o aplicativo PSCP[11]:

```
C:\pscp.exe PS5UI.DLL PSCRIPT.HLP PSCRIPT.NTF PSCRIPT5.DLL
root@10.1.1.1:/usr/local/share/cups/drivers
```

Durante a cópia, é possível que os arquivos apareçam com seus nomes em caixa-alta. Em sistemas Unix, nomes de arquivos são *case-sensitive*, portanto, para evitar problemas, é importante deixar em caixa-baixa todos os nomes dos arquivos de *drivers* copiados.

Novamente, de acordo com o manual do cupsaddsmb, é necessário buscar o *driver* de impressão para Windows que permite a utilização do PostScript versão 6. Este *driver* pode ser encontrado em http://www.cups.org/windows/software.php?6.0. Os arquivos necessários são:

- CUPS6.INF
- CUPS6.INI
- CUPSUI6.DLL
- CUPSPS6.DLL

---

10 Disponível em <http://www.adobe.com/support/downloads/product.jsp?product=44&platform=Windows>. Acesso em 5 de julho de 2008.

11 *Software* livre para Windows, disponível em <http://the.earth.li/~sgtatham/putty/latest/x86/pscp.exe>. Acesso em 5 de julho de 2008. Análogo ao scp em sistemas tipo Unix, permite a transferência de arquivos de uma máquina à outra. Neste caso, de um sistema Windows para um sistema tipo Unix (OpenBSD).

Deve-se extrair o conteúdo do pacote do *driver* e copiar os arquivos necessários para o diretório criado anteriormente:

```
# tar -xzf cups-windows-6.0-source.tar.gz
# cd cups-windows-6.0/i386
# cp cups6.inf cups6.ini cupsui6.dll cupsps6.dll
/usr/local/share/cups/drivers
```

Finalmente, para cada impressora a ser utilizada deve-se buscar os arquivos .ppd correspondentes, e colocá-los dentro do diretório /etc/cups/ppd. Os arquivos .ppd possuem instruções que são utilizadas por impressoras compatíveis com o PostScript. Tais arquivos funcionam como *drivers*, pois permitem uma interface única que lista as diversas funções disponíveis pela impressora em questão. Estes arquivos podem ser encontrado nos CD's de instalação das impressoras, ou pela Internet, consultando sites de busca ou específicos sobre impressoras em ambientes Unix, tais como o repositório de *drivers* mantido pela The Linux Foundation[12]. Para este projeto, foi utilizado o .ppd relativo à impressora HP PSC 1310, e disponível no site da The Linux Foundation.

O CUPS utiliza o arquivo /etc/cups/printers.conf para definir configurações relativas às impressoras que serão por ele gerenciadas. Abaixo segue um modelo básico para a impressora exemplo deste projeto, utilizando a sintaxe do arquivo printers.conf.

/etc/cups/printers.conf:

```
<Printer psc1310-1.ppd>
    Info            HP PSC 1310 series
    Location        Direcao Administrativa
    DeviceURI       smb://root:senha@cliente.exemplo.br/hppsc1310
    State           Idle
    StateMessage    Printer is idle
    Accepting       Yes
</Printer>
```

Caso seja necessário adicionar mais impressoras, basta inserir mais campos definidos por <Printer> e </Printer> contendo as informações de cada uma delas.

A opção Printer especifica o nome do *driver* utilizado pela impressora, ou seja, o nome do arquivo .ppd correspondente. Neste exemplo, o .ppd armazenado em /etc/cups/ppd/psc1310-1.ppd. Os campos Info e Location são de interesse administrativo. O primeiro permite definir uma breve descrição da impressora, enquanto o segundo recebe uma descrição da localização desta. Caso seja necessário utilizar mais de uma impressora do

12 Disponível em <http://openprinting.org>. Acesso em 5 de julho de 2008.

mesmo modelo, basta utilizar o mesmo arquivo .ppd, com outro nome (por exemplo, psc1310-2.ppd) e referenciá-lo nos arquivos de configuração pertinentes (printers.conf, smb.conf).

O campo DeviceURI recebe a localização lógica da impressora. Neste projeto, a impressora em questão estará conectada à uma máquina cliente, no setor administrativo. Como a máquina cliente utiliza Windows XP como sistema operacional, o *backend* smb será utilizado para permitir a comunicação entre o CUPS e a impressora, utilizando o protocolo SMB, implementado pela ferramenta smbspool do Samba. Para que o CUPS possa utilizar o *backend* smb, é necessário disponibilizá-lo, através de um elo simbólico por exemplo:

```
# ln -s /usr/local/bin/smbspool  /usr/lib/cup s/backend/smb
```

A impressora no computador cliente deve então ser compartilhada, acessando a janela disponível em Iniciar, Configurações, Impressoras e Aparelhos de Fax. Clica-se na impressora com o botão direito e seleciona-se Propriedades. Na aba Compartilhamento, seleciona-se a opção "Compartilhar esta impressora", e logo abaixo, define-se o nome deste compartilhamento, sendo utilizado neste projeto, o nome hppsc1310.

Resta então configurar o Samba,para que este utilize a infra-estrutura do CUPS como servidor de impressão. Para isto, as seguintes linhas são adicionadas na seção [global] do arquivo de configuração do Samba em /etc/samba/smb.conf:

/etc/samba/smb.conf:

```
load printers = yes
printing = cups
printcap name = cups
```

A opção load printers define se as impressoras serão listadas para os clientes como recursos disponíveis. Já printing = cups define que as informações de status de impressão serão interpretadas pelo sistema como sendo do tipo CUPS. Por fim, printcap name define a interface de impressão que será utilizada pelo sistema.

Ainda no arquivo de configuração, é preciso criar o compartilhamento de impressão para a impressora desejada. Primeiramente, adiciona-se a impressora criando uma seção específica:

/etc/samba/smb.conf:

```
[psc1310-1]
     comment = HP PSC 1310 # Comentário arbitrário
```

```
    valid users = root @DomainUsers
    path = /var/spool/samba/printing
    printer = psc1310-1
    public = no
    writable = no
    printable = yes
```

Para evitar problemas com o comando cupsaddsmb, é importante que o nome da impressora definido acima em printer seja igual ao nome da impressora definido nas configurações do CUPS, em /etc/cups/printers.conf. Em valid users é possível definir quais usuários terão acesso à esta impressora. No exemplo acima, possuem tal permissão o usuário root e o grupo DomainUsers. É essencial que o diretório utilizado pela impressora tenha permissão de escrita pelos usuários, além do *sticky-bit*[13] habilitado, para assegurar que os usuários não sobrescrevam os arquivos (*spools* de impressão) de outros usuários.

```
# chgrp DomainUsers /var/spool/samba/printing
# chmod 1770 /var/spool/samba/printing
```

O outro compartilhamento de impressão que deve ser criado é o diretório que contém os *drivers* para Windows. As máquinas clientes devem ter acesso à este diretório para buscá-los quando uma nova impressora for adicionada. Isso é feito criando o seguinte compartilhamento em /etc/samba/smb.conf:

/etc/samba/smb.conf:

```
[print$]
    comment = Drivers de Impressão
    path = /etc/samba/drivers
    browseable = no
    guest ok = no
    read only = yes
    write list = root
```

Cria-se então o diretório especificado acima:

```
# mkdir /etc/samba/drivers
```

Com todas essas configurações feitas, é possível exportar as impressoras do CUPS para o domínio Samba, utilizando a ferramenta cupsaddsmb:

```
# cupsaddsmb -H localhost -U root -v -a
```

O servidor CUPS pode então ser iniciado executando o daemon /usr/local/sbin/cupsd.

---

13 Mecanismo utilizado em sistemas tipo Unix para garantir que os arquivos de um diretório possam ser modificados apenas pelo usuário dono, pelo dono do diretório, ou pelo superusuário.

4.3 *Proxy* para *caching* e controle de acesso à Internet

Um servidor *proxy* (do inglês, procurador, intermediário) é um serviço utilizado em redes de computadores que atua intermediando requisições de clientes à outros servidores. Existem diversos tipos de serviços de rede que atuam como *proxies*, como os *proxies* que fornecem anonimato ao usuário (*anonymous proxies*), *proxies* de controle de acesso (alguns *web proxies*), e até mesmo *proxies* para escutas ilegais (*hostile proxies*). Será implementado neste projeto, os serviços de *web* e *caching proxies*, aumentando as possibilidades de controle do administrador de sistemas sobre os acessos à Internet, e permitindo um ganho de performance nas requisições de clientes à rede mundial de computadores.

O *web proxy* é o serviço que atua intermediando o tráfego de dados de clientes à *World Wide Web* (WWW ou simplesmente *web*). A utilização de um *web proxy* é vantajosa do ponto de vista administrativo, pois permite ao administrador de sistemas um maior controle sobre o acesso que seus clientes possuem à Internet e protocolos populares da *web*, tais como HTTP, HTTPS e FTP. Esse controle permite uma restrição de acesso à paginas arbitrárias, de acordo com os usuários ou grupo de usuários da rede interna. Um aumento na segurança da rede é visível, visto que o administrador pode bloquear acessos à paginas de conteúdo impróprio ou danoso, como sites de *phishing*, e redutos de vírus. Este bloqueio pode resultar inclusive em uma economia de banda, visto que o *proxy* não irá efetuar *download* de conteúdo bloqueado pelo administrador.

O *caching proxy* permite que requisições sejam servidas sem a necessidade de se conectar ao servidor desejado, utilizando-se dados armazenados de uma requisição anterior, que pode ter sido feita pelo mesmo cliente requisitante, ou por outros clientes. Ao armazenar uma cópia de dados frequentemente solicitados pelos clientes, um *caching proxy* possibilita uma boa economia na banda utilizada para o acesso à *web*, além de um aumento de performance para os clientes da rede interna, pois em alguns casos, o servidor não precisará buscar informações na Internet, visto que já as possui em *cache*.

Para este projeto, será utilizado o popular *software* Squid para fornecer os serviços de *web* e *caching proxy*, juntamente com a ferramenta ntlm_auth (disponível pelo Samba) para

realizar a autenticação dos usuários, e a ferramenta SquidGuard, que provê mecanismos de controle de acesso à paginas da *web*.

4.3.1 Configuração do servidor *proxy*

O *software* Squid será instalado na máquina que irá atuar como o servidor *proxy* de toda a rede. Esta máquina também terá uma instância do servidor de DNS, o BIND, em funcionamento, para resolver os nomes de domínio de servidores localizados fora da rede interna (Internet), visto que o servidor DNS implementado no PDC apenas gerencia os nomes de domínio da rede interna. É possível assim, utilizar o servidor Proxy em uma rede DMZ, para proporcionar maior segurança.

A instalação do Squid pode ser feita através do mecanismo de *packages* do OpenBSD.

```
# pkg_add squid-2.6.STABLE13
```

Para que o Squid atue como um *web proxy*, identificando os clientes e relacionando permissões de acesso de acordo com o usuário em questão, é necessário também instalar o Samba com a extensão winbind ativada, que provê o Squid com um mecanismo para realizar a autenticação de usuários com uma base de dados, neste caso, a base de usuários localizada no servidor PDC. Infelizmente, os pacotes disponíveis no OpenBSD não vêm com esta extensão. É necessário portanto utilizar o sistema de *ports* para compilar um pacote com a extensão winbind ativada no Samba.

O arquivo contendo as instruções para compilação de programas via *ports* pode ser obtido em qualquer um dos *mirrors* de FTP disponíveis pelo projeto OpenBSD, por exemplo, em ftp://ftp.openbsd.org/pub/OpenBSD/4.2/ports.tar.gz (onde 4.2 é a versão do sistema operacional).

O arquivo deve ser descompactado dentro do diretório /usr:

```
# tar -xzf ports.tar.gz
```

As instruções para a criação de um pacote do Samba estão disponíveis no arquivo /usr/ports/net/samba/Makefile. Para habilitar o winbind com suporte à autenticação no pacote a ser criado, as linhas --with-winbind e –with-winbind-auth-challenge devem ser adicionadas na seção CONFIGURE_ARGS do Makefile:

/usr/ports/net/samba/Makefile:

```
CONFIGURE_ARGS=  --localstatedir="/var"  \
                 --sbindir="${PREFIX}/libexec"  \
                 --with-configdir="${CONFDIR}"  \
                 --with-libdir="${PREFIX}/lib/samba"  \
                 --with-lockdir="/var/spool/samba"  \
                 --with-piddir="/var/run"  \
                 --with-logfilebase="${SAMBA_LOGDIR}"  \
                 --with-privatedir="${CONFDIR}"  \
                 --with-libsmbclient  \
                 --with-swatdir="${PREFIX}/share/swat"  \
                 --with-ssl  \
                 --with-sslinc="/usr/include/ssl"  \
                 --with-sslib="/usr/lib"  \
                 --with-syslog  \
                 --with-winbind \
                 --with-winbind-auth-challenge \
                 --with-utmp
```

O pacote então pode ser compilado e instalado executando o seguinte comando dentro do diretório /usr/ports/net/samba:

```
# make install
```

A função da ferramenta winbind é buscar informações sobre nomes e grupos de usuários de um servidor Windows NT. Neste projeto, ele será utilizado para fornecer à ferramenta ntlm_auth, nomes de usuários residentes no PDC, permitindo que o Squid a utilize para autenticar os usuários que devem ter acesso ao seus serviços de *proxy*.

O winbind buscará suas informações de configuração em /etc/samba/smb.conf, portanto, para que o mesmo funcione corretamente, é necessário que os demais serviços do Samba (smbd e nmbd) também estejam em funcionamento no mesmo computador. A instância do Samba que funcionará no servidor Proxy deve conter apenas as informações relativas à sua comunicação com o Samba do servidor PDC, pois o Proxy não possui a função de armazenar as contas de usuários da rede. Isso resulta em um arquivo de configuração enxuto, conforme descrito no apêndice E desta monografia.

A opção security = domain indica que este servidor atuará como membro de um domínio Samba. Ao final, estão as opções relativas ao winbind. A opção Password server define a base de contas utilizada para efetuar a autenticação dos usuários, winbind separator = / define o tipo de separador utilizado pelo winbind para listar os usuários (neste caso, a listagem ocorre na sintaxe DOMÍNIO/usuário), e winbind use default domain define se o winbind interpretará

usuários que não possuem nome de domínio como se fossem do domínio vigente (exemplo.br).

Para que o servidor *proxy* tenha permissão para consultar os usuários da rede, é necessário adicionar à base de usuários do PDC uma conta de computador relativa ao *proxy*, efetuando assim seu ingresso no domínio de rede gerenciado pelo PDC. Essa inclusão é feita utilizando o comando net rpc join, passando como argumentos o endereço do PDC (10.1.1.1) e o usuário com privilégios para efetuar a operação (administrador).

```
# net rpc join -S 10.1.1.1 -U administrador
```

Com o winbind configurado, prossegue-se com a configuração do *software* de *proxy*, o Squid. Conforme citado anteriormente, para que o Squid efetue a autenticação, é necessário utilizar a ferramenta ntlm_auth, pertencente à suíte de ferramentas do Samba, e localizada em /usr/local/bin/ntlm_auth. Ela deve ser referenciada no arquivo de configurações do Squid para que o mesmo possa utilizá-la.

Para que a ferramenta ntlm_auth possa funcionar adequadamente, o Squid precisa ter acesso de leitura ao *pipe*[14] do winbind localizado no diretório /var/spool/samba/winbindd_privileged, para verificar diretamente os dados de autenticação. A permissão pode ser criada modificando a ID de grupo do arquivo em questão, para a ID de grupo do Squid (no OpenBSD, o Squid utiliza _squid como nome de usuário e grupo):

```
# chgrp _squid /var/spool/samba/winbindd_privileged
```

Com o winbind configurado, é possível dar prosseguimento à configuração do Squid. O próximo passo é a instalação do Squidguard, que será utilizado para realizar o controle de acesso. Sua instalação pode ser feita via *packages*:

```
# pkg_add squidGuard-1.2.1
```

Seu arquivo de configuração está localizado em /etc/squidguard/squidguard.conf, os arquivos de log em /var/squidguard/log, e as bases de dados de endereços de domínio em /var/squidguard/db.

O SquidGuard será utilizado para controlar o acesso à determinados sites, agindo como um módulo redirecionador para o Squid. O Squid realizará através da ferramenta ntlm_auth, a

---

14 Em sistemas tipo Unix, *pipe* é o conceito de redirecionamento da saída padrão (stdout) de um programa para a entrada padrão (stdin) de outro programa.

autenticação do usuário, que será então repassado ao SquidGuard. Em seguida, o SquidGuard verificará se o usuário possui acesso liberado para o endereço requisitado, de acordo com regras definidas pelo administrador de sistemas, e o redirecionará para uma tela de acesso negado, ou para o site requisitado.

Primeiramente, é preciso habilitar o SquidGuard nas configurações do Squid, identificando-o como o redirecionador a ser utilizado, adicionando as seguintes linhas em /etc/squid/squid.conf:

/etc/squid/squid.conf:

```
url_rewrite_program /usr/local/bin/squidGuard
url_rewrite_children 10 #Determina o número de instâncias.
```

Conforme comentado, será utilizada a ferramenta ntlm_auth, proporcionando autenticação transparente e segura para o usuário, através do protocolo NTLMv2. Este tipo de autenticação é determinada no squid.conf com as seguintes linhas de configuração:

/etc/squid/squid.conf:

```
#Autenticação via NTLM
auth_param ntlm program /usr/local/bin/ntlm_auth --helper-
protocol=squid-2.5-ntlmssp
auth_param ntlm children 10 #Determina o número de instâncias.
...
acl autenticacao proxy_auth REQUIRED
...
http_access allow autenticacao
```

De acordo com a configuração acima, o servidor Proxy irá utilizar apenas o protocolo NTLMv2 para processar a autenticação dos usuários, através do serviço NTLMSSP, e apenas os usuários autenticados terão acesso aos serviços de *proxy*.

Uma boa opção administrativa é efetuar o bloqueio de sites de acordo com grupos de usuários, criando-se grupos com determinados privilégios, e adicionando-se usuários à estes grupos de acordo com a necessidade. O SquidGuard porém, não consegue identificar diretamente os grupos dos usuários, visto que apenas o nome de usuário lhe é enviado pelo Squid. A filtragem deve então ser feita utilizando apenas nomes de usuários como argumentos. Há porém, a possibilidade de se agrupar uma lista de nomes de usuários em um ou mais arquivos, os quais são processados pelo SquidGuard. Dessa forma, é possível criar vários arquivos, cada um correspondendo a um grupo, contendo os nomes dos usuários pertencentes ao grupo. Para que o administrador de sistemas não tenha que editar estes

arquivos a cada entrada ou saída de usuários de um grupo, um *script* foi desenvolvido para este projeto, para automatizar o processo de atualização destas listas de usuários. Este script foi denominado criar_listas.sh, e encontra-se nos apêndices desta monografia.

O *script* criado utiliza o comando getent group para buscar os dados de grupos do sistema local (/etc/group), portanto, este *script* deve ser executado no servidor PDC, e não no servidor *proxy* (pois o *proxy* não possui o banco de dados de usuários e grupos do sistema). Para que o SquidGuard possa utilizar estas listas, é necessário então enviá-las para o servidor *proxy*, o que pode ser feito utilizando o comando scp, pois o mesmo efetua uma cópia segura (através de criptografia) pela rede. Para que esta cópia funcione de forma automatizada (pelo calendário de tarefas tipo Unix do OpenBSD por exemplo), via *script*, é necessário utilizar o mecanismo de chave pública/privada para efetuar a autenticação, pois não é desejável digitar uma senha de acesso ao *proxy* cada vez que o *script* for executado, o que quebraria a automatização. Deve-se então criar um par de chaves para o usuário que irá efetuar a cópia (no caso deste *script*, o usuário suporte do servidor PDC), utilizando um valor nulo para a *passphrase* que será pedida:

```
$ ssh-keygen
```

Em seguida, adiciona-se a chave pública criada (localizada em /home/suporte/.ssh/id_rsa.pub) para a lista de chaves autorizadas do usuário suporte no servidor *proxy*, em /home/suporte/.ssh/authorized_keys. Isto pode ser feito no servidor PDC através do comando scp (o diretório /home/suporte/.ssh deve existir no servidor Proxy):

```
$ scp /home/suporte/.ssh/id_rsa.pub
suporte@10.1.1.5:/home/suporte/.ssh/authorized_keys
```

A verificação de usuários feita ao final do *script* é importante, pois assim o mesmo pode ser utilizado sob demanda pelo usuário suporte, e também pelo calendário de tarefas do OpenBSD (o cron, que executa seus comandos como usuário root), visto que a sintaxe da ação de cópia depende do usuário que a executa (root ou suporte). Nota-se que o usuário utilizado para a cópia deve ter permissão de escrita no diretório em questão, e o mesmo tem que existir. Além disso, o SquidGuard precisa ter permissão para ler o diretório e seu conteúdo. Tudo isso é resolvido executando os seguintes comando no servidor *proxy*:

```
# mkdir -p /etc/squidguard/listas
# chown -R _squid /etc/squidguard/listas
# chgrp -R wheel /etc/squidguard/listas
```

```
# chmod -R 570 /etc/squidguard/listas
```

As linhas do *script* que se referem aos grupos do sistema devem ser ajustadas de acordo com as necessidades de cada implementação. Para este *script*, foi utilizado o caso de uma instituição de ensino, exemplificando a criação de dois arquivos, secretaria e tesouraria (representando os grupos de sistema de mesmo nome), onde cada arquivo contém os nomes de seus respectivos usuários.

As listas de usuários que foram criadas devem então ser referenciadas no arquivo de configuração do SquidGuard em /etc/squidguard/squidguard.conf, conforme descrito no apêndice H.

Por fim, é necessário definir as regras de filtro, e quais sites devem ou não ser restritos. Uma opção é utilizar uma lista de sites disponíveis pela empresa Shalla Secure Services. Esta empresa dispõe na Internet uma lista com mais de 1,5 milhão de sites, classificados por conteúdo, para que o administrador possa utilizar como referência para filtrar sites que contenham propagandas, *phishing*, ou pornografia. O uso privado ou comercial destas listas é gratuito, porém, a empresa demanda a assinatura de um contrato de uso, caso suas listas sejam usadas para fins comerciais. Outra opção é manter listas próprias, de acordo com as necessidades da instituição.

Para utilizar listas próprias, é necessário criá-las no sistema, e em seguida referenciá-las nas configurações do SquidGuard:

/etc/squidguard/sites_liberados:

```
# Domínios Liberados
# Formato SquidGuard
# Ex. globo.com
buriti.df.gov.br
```

/etc/squidguard/sites_bloqueados:

```
# Domínios Bloqueados
# Formato SquidGuard
# Ex.: globo.com
orkut.com
```

/etc/squidguard/squidguard.conf:

```
### Definicao dos Grupos de Destino (listas de endereços)
## Listas próprias:
dest lista1 {
domainlist /etc/squidguard/sites_liberados
}
```

```
dest lista2 {
domainlist /etc/squidguard/sites_bloqueados
}
```

No caso das listas disponíveis pela empresa Shalla Secure Services, deve-se buscá-las no site, e descompactá-las no sistema. Uma maneira de buscar a lista é utilizar a ferramenta wget, disponível via *packages* no OpenBSD.

```
# pkg_add wget
# cd /var/squidguard/db
# wget http://www.shallalist.de/Downloads/shallalist.tar.gz
# tar -xzf shallalist.tar.gz
```

As listas serão criadas no diretório /var/squidguard/db/BL, em formato de texto comum, cada arquivo contendo os endereços de sites de acordo com o conteúdo classificado pelo diretório. Por exemplo, o diretório /var/squidguard/db/BL/adv é relativo à sites de propaganda (*advertisement*), e contém dois arquivos, domain e urls. O primeiro contém endereços de domínio (por exemplo, globo.com), e o segundo contém endereços completos (URL).

Como alguns desses arquivos possuem uma grande quantidade de entradas (o diretório de pornografia fornecido pela Shalla Secure Services possui mais de 700 mil domínios e quase 80 mil URL's), o SquidGuard possui a opção de repassar essas entradas para um arquivo em formato de banco de dados, agilizando o tempo de reposta na consulta de um endereço. Para isso, é necessário criar um arquivo de configuração, contendo as entradas de texto que devem ser convertidas em um arquivo de banco de dados, descrito no apêndice I. Deve-se em seguida, executar o seguinte comando para iniciar o processo de conversão:

```
# squidGuard -C all -c /etc/squidguard/squidguard- listas.conf
```

Os arquivos de banco de dados serão criados em /var/squidguard/db/BL. Para que o SquidGuard possa acessá-los, é necessário ajustar as permissões para que o usuário _squid tenha acesso às listas de endereços:

```
# chown -R _squid /var/squidguard/db/BL/
```

Para que o administrador não tenha que realizar todos estes procedimentos quando for necessário atualizar as listas, foi desenvolvido um *script* para este projeto, visando automatizar o processo de atualização e criação das listas obtidas na Internet. É possível utilizá-lo no agendador de tarefas do OpenBSD (cron) ou referenciá-lo em /etc/weekly.local para que sua execução aconteça semanalmente, junto com a rotina padrão de manutenção do

sistema operacional. Este *script* é denominado atualizar-listas.sh e se encontra nos apêndices desta monografia.

Por fim, resta ao administrador definir as regras de bloqueio, de acordo com os grupos de usuários. As regras são definidas no arquivo de configuração do SquidGuard. Como exemplo, as configurações abaixo realizam três tipos de bloqueios diferentes. Um para usuários do grupo secretaria, onde são bloqueados sites contidos no grupo lista2 e adv, outro para o grupo tesouraria, onde é permitido o acesso apenas aos sites contidos na lista1, e outro para qualquer usuário que por ventura não pertença a nenhum dos dois grupos citados, onde o acesso é totalmente bloqueado. Todo acesso à endereços proibidos será redirecionado para o endereço especificado em redirect:

/etc/squidguard/squidguard.conf:

```
acl {

 Secretaria {
 pass lista1 !lista2 !adv all
 redirect http://proxy.exemplo.br/bloqueio.htm
 }

 Tesouraria {
 pass lista1 none
 redirect http://proxy.exemplo.br/bloqueio.htm
 }

 default {
 pass none
 redirect http://proxy.exemplo.br/bloqueio.htm
 log default.log
 }

}
```

A versão completa dos arquivos squid.conf e squidguard.conf pode ser encontrada nos apêndices desta monografia.

Por fim, para que os serviços sejam executados juntamente com a inicialização do servidor, é necessário adicionar linhas de execução ao arquivo /etc/rc.local para cada *daemon* a ser iniciado, conforme descrito no apêndice F.

4.4 Serviço de resolução de nomes (DNS)

O DNS (*Domain Name System*) é o mecanismo utilizado em redes de computadores para relacionar nomes à informações, como nomes de domínio à endereços IP (*Internet Protocol*). Sua utilização possui importância administrativa, visto que o administrador pode montar a rede de seu parque computacional seguindo um padrão hierárquico de nomes, contribuindo para a organização do mesmo, e fornecendo uma estrutura de fácil memorização (não é preciso memorizar os números dos endereços IP dos computadores).

Neste projeto, será utilizado o *software* BIND (*Berkeley Internet Name Domain*) para prover os serviços de servidor de nomes. O BIND é um *software* livre mantido pela ISC (*Internet Systems Consortium*), bastante utilizado na Internet como servidor de DNS, tendo se tornado praticamente um padrão em implementações de DNS em ambientes tipo Unix. O OpenBSD versão 4.2 já possui o BIND versão 9.3.4 integrado ao sistema operacional, junto com modificações feitas pelos próprios desenvolvedores do OpenBSD.[15]

Será utilizado neste projeto, dois servidores DNS, sendo um no PDC, e outro no *firewall*. Enquanto o primeiro irá resolver os nomes da rede interna, o segundo servirá como um *cache* para endereços externos (Internet).

Vale ressaltar que a configuração de servidores DNS é assunto complexo, com vários livros escritos à respeito. Os tópicos seguintes possuem o objetivo de apresentar apenas uma configuração simples para a rápida implementação de um servidor DNS para gerenciar endereços internos, e outro para endereços externos.

4.4.1 Configuração do BIND no PDC

No OpenBSD, o principal arquivo de configuração do BIND, named.conf, está localizado em /var/named/etc. É neste arquivo que se definem as principais funções do servidor DNS. Antes de configurá-lo, é interessante criar uma cópia de segurança da versão original.

```
# cp /var/named/etc/named.conf /var/named/etc/named.conf.orig42
```

---

15 Em julho de 2007, foi anunciada uma falha de segurança nas versões 9 do BIND distribuído pela ISC. Modificações feitas pelos desenvolvedores do OpenBSD quando da incorporação do BIND 9 ao sistema operacional, preveniram que o mesmo fosse afetado pela falha de segurança. Tal fato contribui para a fama de segurança pró-ativa do projeto OpenBSD.

Foi desenvolvido para este projeto, um arquivo de configuração básico adequado para o ambiente do servidor PDC, tendo a configuração padrão do OpenBSD como base. Este arquivo se encontra nos apêndices desta monografia.

O arquivo de configuração foi desenvolvido tendo a configuração padrão do OpenBSD como base. Foram declaradas quatro zonas de DNS. A primeira, denominada “exemplo.br”, gerencia a resolução de nomes para endereços IP dentro do domínio exemplo.br. A segunda, “1.1.10.in-addr.arpa”, foi incluída para fornecer resolução inversa de nomes, ou seja, de um número para um DNS, útil quando se possui o endereço mas não se sabe o nome do computador. A terceira e a quarta foram adicionadas de acordo com a RFC 1912, que recomenda a implementação de zonas para endereços *broadcast* e locais, para evitar requisições acidentais destes para os servidores raízes da Internet.

Após a declaração destas zonas, é necessário informar os detalhes sobre cada uma delas, em arquivos de configuração de zonas que devem ser inseridos em /var/named/master. Foram desenvolvidos para este projeto, arquivos de configuração para as zonas mencionadas anteriormente, estando estes disponíveis nos apêndices desta monografia. Por questão de organização, os arquivos são nomeados começando pelo prefixo db (db.exemplo.br, db.10.1.1, etc).

As entradas de DNS originárias de atualizações dinâmicas (*Dynamic Updates)* são inseridas em um arquivo de extensão .jnl, de acordo com o arquivo de zona a ser modificado (por exemplo, db.exemplo.br.jnl para o arquivo de zona db.exemplo.br). Os arquivos .jnl são criados e modificados de forma dinâmica pelo BIND, porém, para que isto aconteça, é necessário efetuar as permissões de sistema necessárias:

```
# chgrp named /var/named/master
# chmod 735 master/
```

Por uma questão de disponibilidade, é interessante que o serviço seja executado sempre que o servidor iniciar. Para isso, basta referenciar o BIND no arquivo /etc/rc.local, conforme descrito no apêndice D.

4.4.2 Configuração do BIND no *Proxy*

A configuração do servidor de DNS no servidor Proxy é semelhante à configuração para o servidor PDC apresentada no tópico anterior. Deve-se ajustar o arquivo de configuração principal (named.conf) e em seguida criar os arquivos relativos às zonas declaradas nessa configuração. A principal diferença é que enquanto o servidor PDC foi configurado para gerenciar nomes relativos à endereços da rede interna, o servidor Proxy será configurado para gerenciar os endereços da rede externa (Internet). Este comportamento é desejável caso o servidor Proxy esteja localizado em uma rede DMZ, com acesso direto à Internet.

A configuração padrão do OpenBSD já realiza a função desejada, fornecendo um cache de endereços externos. Esta configuração padrão, com algumas modificações para retirar os serviços que utilizam IPv6 (não utilizado neste projeto), se encontra nos apêndices desta monografia.

4.5 Serviço de configuração dinâmica de clientes (DHCP)

O DHCP (*Dynamic Host Configuration Protocol*) é um protocolo utilizado em redes de computadores, pelo qual máquinas clientes podem receber diversos parâmetros de configuração necessários para operar em uma rede IP. O uso do DHCP em uma rede interna é de grande valia para o administrador de sistemas, visto que os clientes recebem suas configurações básicas de rede de forma automática, muitas vezes eliminando a necessidade de se realizar configurações manuais em cada cliente.

Um servidor DHCP atua escutando a rede por requisições de máquinas que não possuem endereço IP. Ao receber tal requisição, o servidor envia ao cliente informações para sua configuração, como um endereço IP único para a rede, máscara de rede, endereços de *gateway* e servidores DNS.

O OpenBSD possui sua própria implementação de DHCP, porém esta não será utilizada neste projeto por não dispor da funcionalidade de atualizar automaticamente entradas DNS de acordo com o endereço IP distribuído, denominada *Dynamic Updates*. Será utilizado portanto

a implementação de DHCP desenvolvida pela ISC, e disponível no OpenBSD via *packages* ou *ports*.

Ressalta-se porém, que esta funcionalidade pode se tornar um problema de segurança caso não se tenha um controle dos computadores que possuem acesso à rede física. Esta funcionalidade será, porém, implementada neste projeto para demonstrar a possibilidade de se utilizar atualizações dinâmicas, tais como as realizadas pelo Active Directory do Windows 2003 Server, em um ambiente tipo Unix.

De acordo com o cenário apresentado no capítulo 2, o servidor DHCP será instalado no PDC, servindo os clientes do setor administrativo e oferecendo-lhes endereços IP dinâmicos. Os endereços IP dos servidores serão estáticos, sendo definidos manualmente pelo administrador de sistemas.

4.5.1 Configuração do servidor DHCP

Primeiramente, instala-se o DHCP versão 3 da ISC via *packages*:

```
# pkg_add isc-dhcp-server-3.0.4p0
```

O executável dhcpd é então instalado em /usr/local/sbin, e utilizará o arquivo de configuração localizado em /etc/dhcpd.conf. Porém, este é o arquivo de configuração do DHCP do OpenBSD, o qual não é compatível com o DHCP da ISC. Por segurança, é interessante realizar uma cópia deste arquivo antes de editá-lo.

```
# cp /etc/dhcpd.conf /etc/dhcpd.conf.orig42
```

O arquivo /etc/dhcpd.conf consiste em uma lista de parâmetros e declarações, onde são descritas as funções que o servidor DHCP deve realizar, além de informações relativas à topologia da rede. Para este projeto foi desenvolvido um arquivo de configuração próprio para o ambiente proposto. Este arquivo se encontra nos apêndices desta monografia.

No arquivo de configuração, os parâmetros *options* definem o nome de domínio e o endereço dos servidores DNS que serão utilizados (neste projeto, o servidor PDC e o Proxy). Em a*uthoritative,* define-se que o servidor DHCP possui autoridade sobre as configurações de rede, evitando que clientes com endereços IP incompatíveis entrem no segmento de rede. Os parâmetros relativos ao sistema de *Dynamic Updates* são descritos logo em seguida. Em

especial, o parâmetro *ddns-update-style interim* indica o método utilizado para se conectar ao servidor DNS e atualizá-lo. Para que os clientes tenham permissão para atualizar o servidor DNS, define-se *allow client-updates*. Em seguida, a declaração *subnet* define a topologia da rede, e as configurações que serão entregues aos clientes, como o endereço da rede, a máscara da rede, a faixa de IP's que será distribuída, e os endereços de *gateway* e de servidores DNS a serem entregues aos clientes. Por fim, a declaração *zone* define quais zonas utilizar para atualizar o servidor DNS.

O dhcpd utiliza o arquivo dhcpd.leases, localizado em /var/db, para manter uma lista de endereços IP distribuídos, além de informações como tempo de expiração do endereço, endereço MAC do cliente, etc. Antes de iniciar o dhcpd, é necessário criar este arquivo, caso ele já não exista:

```
# touch  /var/db/dhcpd.leases
```

4.6 Encapsulando o tráfego de rede com IPsec

Todos os arquivos que são manuseados pelos usuários estão localizados no servidor PDC, e provavelmente, alguns desses arquivos podem ser confidenciais. É possível que em algumas empresas, um alto funcionário tenha seu diretório de arquivos privado guardado no servidor de arquivos (PDC), devendo o administrador de sistemas se preocupar em garantir a proteção necessária à esses dados, evitando o acesso não autorizado, e impedindo a obtenção destes dados através de uma escuta de rede.

Conforme explicado no capítulo 3, será utilizado a estrutura IPsec para garantir a segurança dos dados que trafegam entre computadores clientes e o servidor PDC. Para isso, será utilizado o *daemon* isakmpd, uma ferramenta disponível no OpenBSD, para prover os serviços de IPsec e de gerenciamento automático de chaves. O PDC armazenará os dados necessários para agir como uma autoridade certificadora[16] com a função de emitir os certificados digitais que serão utilizados para validar ou refutar conexões de IPsec.

Primeiramente, é necessário estabelecer a autoridade certificadora (neste caso, o PDC), criando sua chave privada (ca.key) e seu certificado (ca.crt):

---

16 Em um ambiente em que alta segurança seja desejada, é aconselhável utilizar um computador isolado para atuar como autoridade certificadora, com uma pessoa responsável para emitir os certificados digitais e instalá-los nos clientes.

```
# openssl req -x509 -days 365 -newkey rsa:1024 -keyout
/etc/ssl/private/ca.key -out /etc/ssl/ca.crt
```

Será pedido uma senha (PEM *pass phrase*), a qual deverá ser usada toda vez que a autoridade certificadora precise assinar alguma requisição de certificado digital. O comando acima terá portanto, a seguinte saída:

```
Generating a 1024 bit RSA private key
.............................++++++
.++++++
writing new private key to '/etc/ssl/private/ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

Em seguida, deve-se entrar com os dados referentes à organização ou empresa que possuirá a autoridade certificadora. Para este projeto, os seguintes dados foram utilizados:

```
Country Name (2 letter code) []:BR
State or Province Name (full name) []:Distrito Federal
Locality Name (eg, city) []:Taguatinga
Organization Name (eg, company) []:Exemplo
Organizational Unit Name (eg, section) []:Autoridade Certificadora
Common Name (eg, fully qualified host name) []:pdc.exemplo.br
Email Address []:pdc@exemplo.br
```

Finalizado este passo, a autoridade certificadora estará pronta para uso. A chave privada está localizada em /etc/ssl/private/ca.key, e o certificado digital em /etc/ssl/ca.crt.

Para cada ponto comunicante, deve ser criado um certificado digital. Estes certificados possuem as informações necessárias para estabelecer uma comunicação criptografada, via IPsec, além de informações que definem a identidade de um computador ou usuário.

Para emitir estes certificados, devem ser criadas primeiramente, as requisições de assinatura de certificados, ou CSR's (*Certificate Signing Requests*) para os pares que desejam se comunicar via IPsec. Essas CSR's devem ser posteriormente assinadas por uma autoridade certificadora, gerando assim um certificado digital pronto para uso.

Primeiramente, cria-se um CSR para o PDC:

```
# openssl req -new -key /etc/isakmpd/private/local.key -out
/etc/isakmpd/private/10.1.1.1.csr
```

Nota-se que o CSR foi criado com o endereço IP do PDC como nome. Este será utilizado como a identidade do PDC, visto que seu IP é único na rede. Após a execução do comando

acima, o administrador deve preencher o conteúdo da CSR conforme desejado. Neste projeto, foram utilizadas as seguintes definições:

```
Country Name (2 letter code) []:BR
State or Province Name (full name) []:Distrito Federal
Locality Name (eg, city) []:Taguatinga
Organization Name (eg, company) []:Exemplo
Organizational Unit Name (eg, section) []:Servidores
Common Name (eg, fully qualified host name) []:pdc.exemplo.br
Email Address []:pdc@exemplo.br

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Em seguida, de acordo com os manuais do *isakmpd*, uma extensão contendo o campo subjectAltName deve ser adicionada ao CSR, devendo conter a identidade desejada para o certificado. Neste caso, para o PDC, a identidade será um endereço IP, 10.1.1.1. Após a inserção deste campo, a CSR deve ser assinada pela autoridade certificadora, gerando assim o certificado digital. Tudo isso é feito utilizando o seguinte comando:

```
# env CERTIP=10.1.1.1 openssl x509 -req -days 365 -in /etc/isakmpd/
private/10.1.1.1.csr -CA /etc/ssl/ca.crt -CAkey
/etc/ssl/private/ca.key -CAcreateserial -extfile
/etc/ssl/x509v3.cnf -extensions x509v3_IPAddr -out
/etc/isakmpd/certs/10.1.1.1.crt
```

Será requisitada a senha da autoridade certificadora. Este procedimento irá criar o certificado digital para o PDC, denominado 10.1.1.1.crt, armazenando-o em /etc/isakmpd/certs.

O procedimento para gerar os certificados digitais para os computadores clientes é similar. Primeiro, cria-se a CSR para um determinado cliente:

```
# openssl req -new -key /etc/isakmpd/private/local.key -out
/etc/isakmpd/private/cliente.exemplo.br.csr
```

Nota-se que o ao invés do número IP, foi utilizado um FQDN (*Fully Qualified Domain Name*) para nomear a CSR. Isto é necessário, pois os computadores clientes deste projeto possuem seus endereços IP distribuídos dinamicamente, via DHCP. Dessa forma, os endereços IP podem mudar, e a referência do certificado estaria então incorreta, caso fosse utilizado um endereço IP fixo como identidade. A saída do comando acima foi preenchida com os seguintes dados:

```
Country Name (2 letter code) []:BR
State or Province Name (full name) []:Distrito Federal
Locality Name (eg, city) []:Taguatinga
Organization Name (eg, company) []:Exemplo
Organizational Unit Name (eg, section) []:Administrativo
Common Name (eg, fully qualified host name) []:cliente.exemplo.br
Email Address []:cliente@exemplo.br

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

O passo seguinte é a assinatura da CSR pela autoridade certificadora. Como é desejável utilizar um FQDN para a identidade do cliente, é necessário adicioná-lo através da extensão subjectAltName do certificado X.509. Todo esse processo é feito com o seguinte comando:

```
# env CERTFQDN=cliente.exemplo.br openssl x509 -req -days 365 -in /
etc/isakmpd/private/cliente.exemplo.br.csr -CA /etc/ssl/ca.crt
-CAkey /etc/ssl/private/ca.key -CAcreateserial -extfile
/etc/ssl/x509v3.cnf -extensions x509v3_FQDN -out
/etc/isakmpd/certs/cliente.exemplo.br.crt
```

Por fim, exporta-se o conjunto chave e certificado do cliente (certificado do tipo PKCS-12), que deve ser utilizado para importar os dados do certificado digital nos clientes que utilizam o sistema operacional Windows XP. O capítulo 4.6.1 exemplifica a instalação deste certificado digital em um cliente Windows XP, juntamente com a configuração necessária de regras de IPsec no cliente, visando estabelecer uma conexão IPsec entre o mesmo e o PDC.

```
# openssl pkcs12 -export -in
/etc/isakmpd/certs/cliente.exemplo.br.crt -inkey
/etc/isakmpd/private/local.key -certfile /etc/ssl/ca.crt -out /etc/
isakmpd/certs/cliente.exemplo.br.p12
```

Com os certificados criados, o passo seguinte é armazená-los nos locais apontados pelo isakmpd, para que o mesmo possa gerenciar automaticamente as conexões IPsec e as chaves criptográficas. Os certificados (arquivos .crt) devem ser armazenados em /etc/isakmpd/certs, e o certificado da autoridade certificadora deve ser mantido em /etc/isakmpd/ca. Os comandos até agora realizados já armazenam os certificados do PDC e do cliente em seus devidos diretórios. Resta apenas copiar o certificado da autoridade certificadora para o local apontado pelo isakmpd:

```
# cp /etc/ssl/ca.crt /etc/isakmpd/ca
```

O isakmpd utiliza dois arquivos de configuração para sua configuração, o isakmpd.conf e o isakmpd.policy, ambos localizados em /etc/isakmpd. O primeiro arquivo define regras de conexão e de SA's, enquanto o segundo define as políticas utilizadas pelo isakmpd. Recentemente, o time do OpenBSD criou o arquivo /etc/ipsec.conf, que pode ser usado no lugar do isakmpd.conf. O ipsec.conf possui uma forma de configuração mais simples e menos confusa do que o isakmpd.conf. Dessa forma, será utilizado neste projeto o arquivo /etc/ipsec.conf para definir as regras de conexão, e o isakmpd.policy para definir as políticas de IPsec.

Para estabelecer uma conexão IPsec entre o PDC e o computador cliente deste projeto, é utilizada a seguinte configuração para o ipsec.conf:

/etc/ipsec.conf:

```
# Estabelece um fluxo entre o PDC e o cliente.
ike dynamic esp from cliente.exemplo.br to pdc.exemplo.br
```

A linha acima estabelece que um fluxo de dados IPsec deve ocorrer entre o cliente e o PDC, com gerenciamento de chaves automática (ike), utilizando o protocolo ESP, e o modo dinâmico para utilizar o FQDN do computador como identidade. Para cada cliente a receber uma conexão IPsec, uma nova linha de configuração semelhante à esta deve ser adicionada.

O outro arquivo de configuração é o isakmpd.policy. Determina-se neste arquivo as políticas de IPsec que devem ser aceitas ou impostas pelo *daemon* isakmpd. Para este projeto, a seguinte política foi utilizada:

/etc/isakmpd/isakmpd.policy:

```
Authorizer: "POLICY"
Licensees: "DN:/C=BR/ST=Distrito
Federal/L=Taguatinga/O=Exemplo/OU=Autoridade
Certificadora/CN=pdc.exemplo.br/emailAddress=pdc@exemplo.br"
Conditions: app_domain == "IPsec policy" &&
        esp_present == "yes" &&
        esp_enc_alg != "null" &&
        remote_id_type == "ASN1 DN" &&
        remote_id == "/C=BR/ST=Distrito
Federal/L=Taguatinga/O=Exemplo/OU=Administrativo/CN=cliente.exemplo
.br/emailAddress=cliente@exemplo.br" -> "true";
```

O campo remote_id define as identidades que terão acesso à conexão IPsec com o PDC. Dessa forma, para cada cliente a receber uma conexão IPsec, uma nova linha remote_id deve ser adicionada, contendo os dados da identidade do certificado digital deste cliente.

Os certificados digitais, assim como as configurações do isakmpd (definidas em /etc/ipsec.conf e /etc/isakmpd) possuem informações sensíveis, e devem ter seu acesso restrito no sistema. É necessário então, ajustar as permissões destes arquivos, de forma que apenas o usuário root tenha permissão para ler ou alterar seu conteúdo:

```
# chmod  600  /etc/ipsec.conf
# chmod  600  /etc/isakmpd/isakmpd.policy
# chmod  600  /etc/ssl/ca.crt   /etc/ssl/ca.srl
# chmod  600  /etc/ssl/private/ca.key
# chmod  -R 600  /etc/isakmpd/ca
# chmod  -R 600  /etc/isakmpd/certs
# chmod  -R 600  /etc/isakmpd/private
```

Por fim, o *daemon* já pode ser testado. O comando isakmpd -vdD A=50 pode ser utilizado para visualizar o *log* do isakmpd, ajudando na detecção de eventuais problemas na implementação.

Para que o sistema inicie o *daemon* de gerenciamento automático de chaves (isakmpd) de forma automática para cada inicialização do sistema, deve-se editar o arquivo /etc/rc.conf.local, e adicionar a seguinte linha:

/etc/rc.conf.local:

```
isakmpd_flags=""
```

4.6.1 Configuração dos certificados nos clientes Windows XP.

Conforme mencionado, para que a conexão IPsec seja realizada, os clientes devem possuir seus respectivos certificados digitais instalados. No tópico anterior, foi demonstrado como gerar um certificado digital do tipo PKCS-12 para ser utilizado por computadores Windows XP. Este certificado deve ser levado manualmente ao cliente, e instalado conforme os procedimentos apresentados a seguir.

Primeiramente, deve-se abrir o aplicativo mmc, disponível em Iniciar, Executar, mmc. Em seguida, clicando-se em Arquivo, Adicionar/Remover snap-in, Adicionar, adicionam-se as seguintes extensões ao mmc: Certificados, e Gerenciamento de diretivas de segurança.

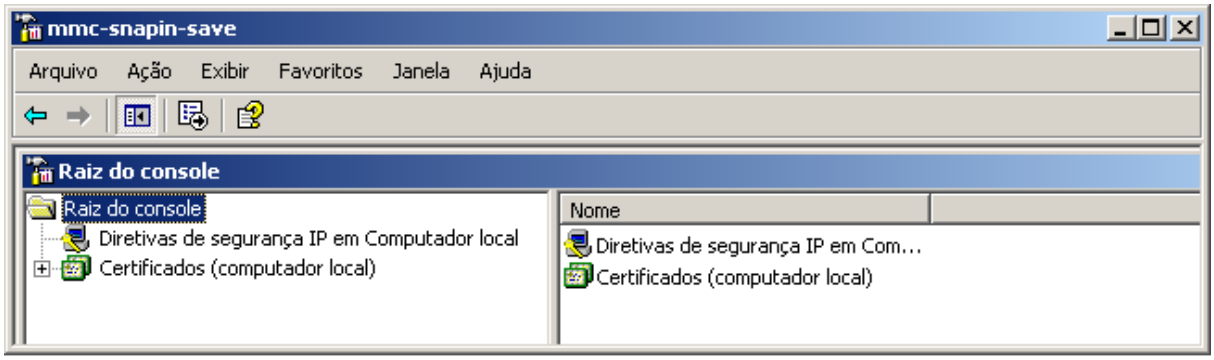

Figura 7: MMC com as extensões adicionadas.

Para adicionar o certificado digital, expande-se a opção Certificados, clica-se com o botão direito na pasta Pessoal, e seleciona-se a opção "Todas as tarefas", Importar. Com o certificado digital PKCS-12 do cliente em mãos (criado no capítulo 4.6), clica-se em Avançar, Procurar, seleciona-se a opção Troca de Informações Pessoais, e especifica-se o caminho onde o certificado digital está localizado, conforme ilustra a figura a seguir:

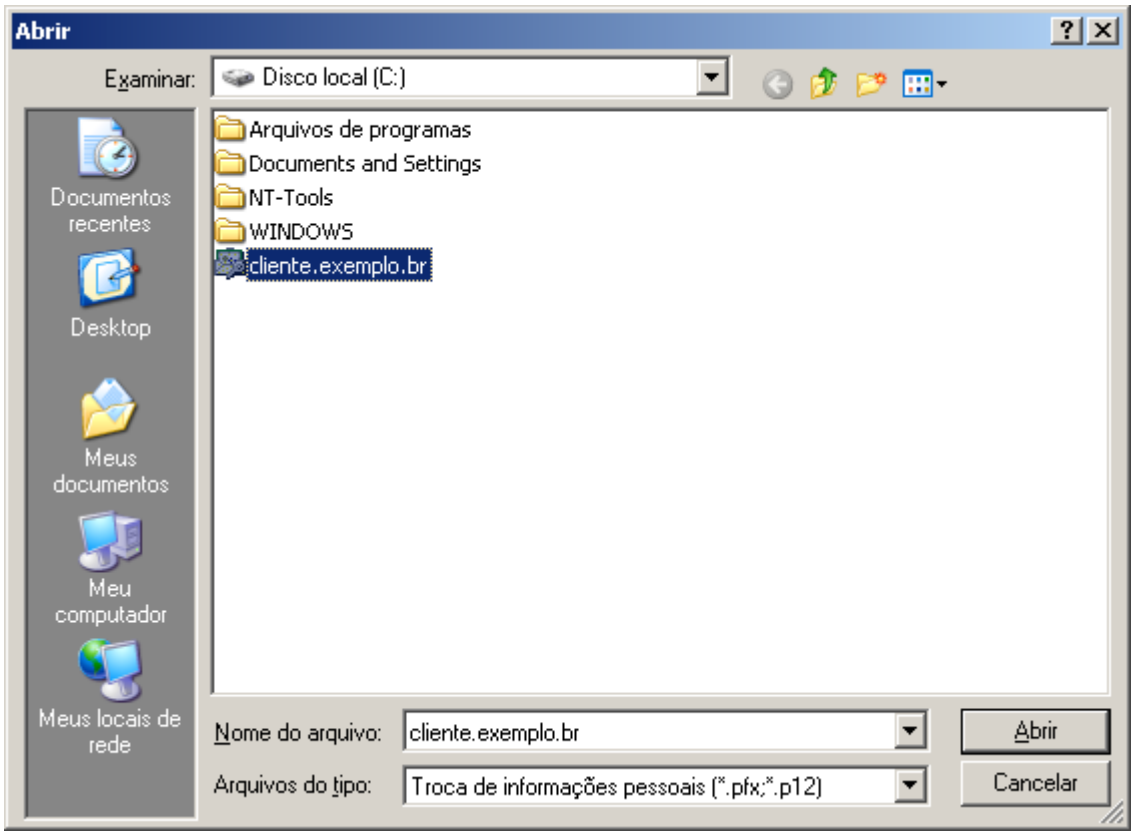

Figura 8: Importando o certificado digital.

Caso o certificado tenha sido criado com uma senha, a mesma deve ser especificada na tela seguinte. Neste projeto, nenhuma senha para o certificado do cliente foi criada, podendo o

campo ser deixado em branco. Na próxima tela, seleciona-se a opção "Selecionar automaticamente o armazenamento de certificados conforme o tipo de certificado." para permitir que o Windows gerencie automaticamente a localização do certificado do cliente.

Com a importação concluída, o certificado digital já se encontra instalado no sistema operacional, conforme ilustra a figura a seguir:

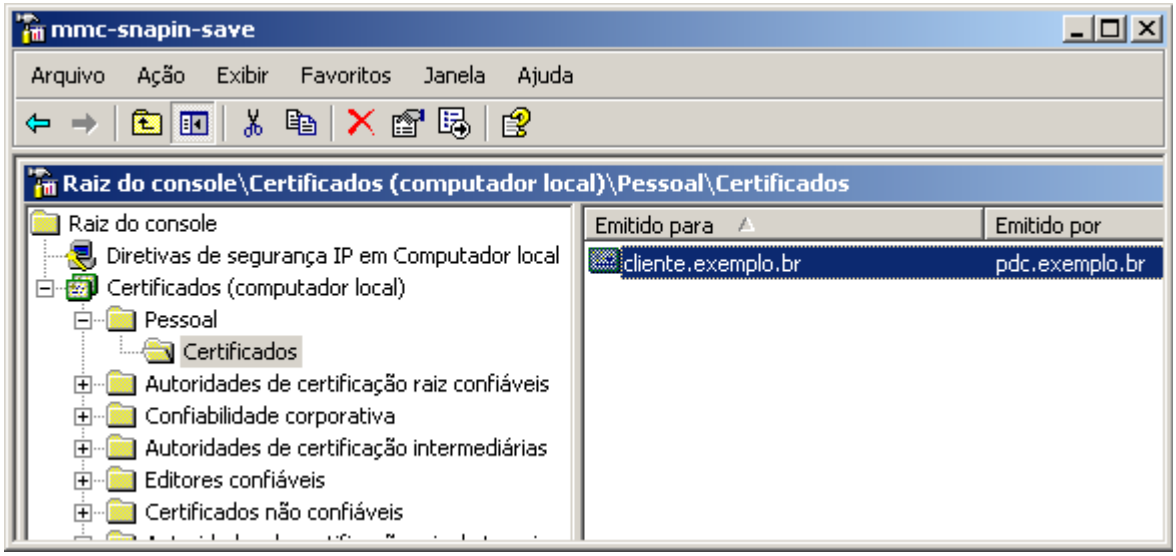

Figura 9: Certificado importado.

O próximo passo é estabelecer as regras de IPsec. Ainda no mmc, clica-se com o botão direito na opção "Diretivas de segurança IP", e seleciona-se a opção Criar Diretiva de Segurança IP. Na tela seguinte, deve-se definir o nome desejado para a regra de IPsec (neste caso, será nomeado como "Tráfego PDC, cliente"), e a opção "Ativar regra de resposta padrão" deve ser desativada. Ao concluir a operação, a tela de edição de regras de segurança IPsec estará disponível. Primeiramente, deve ser definido o fluxo de dados que utilizará a comunicação IPsec. Isso é feito clicando-se em Adicionar, e na aba Lista de filtros IP, clicando-se novamente em Adicionar. Na tela seguinte, especifica-se um nome para a regra de filtro (neste projeto, foi utilizado o nome "PDC, cliente"), e cria-se uma através do botão Adicionar. Seleciona-se como endereço de origem a opção "Meu endereço de IP" e como endereço de destino, a opção "Um endereço IP específico", especificando-se em seguida, o endereço IP do PDC (10.1.1.1). A figura a seguir ilustra este processo de configuração:

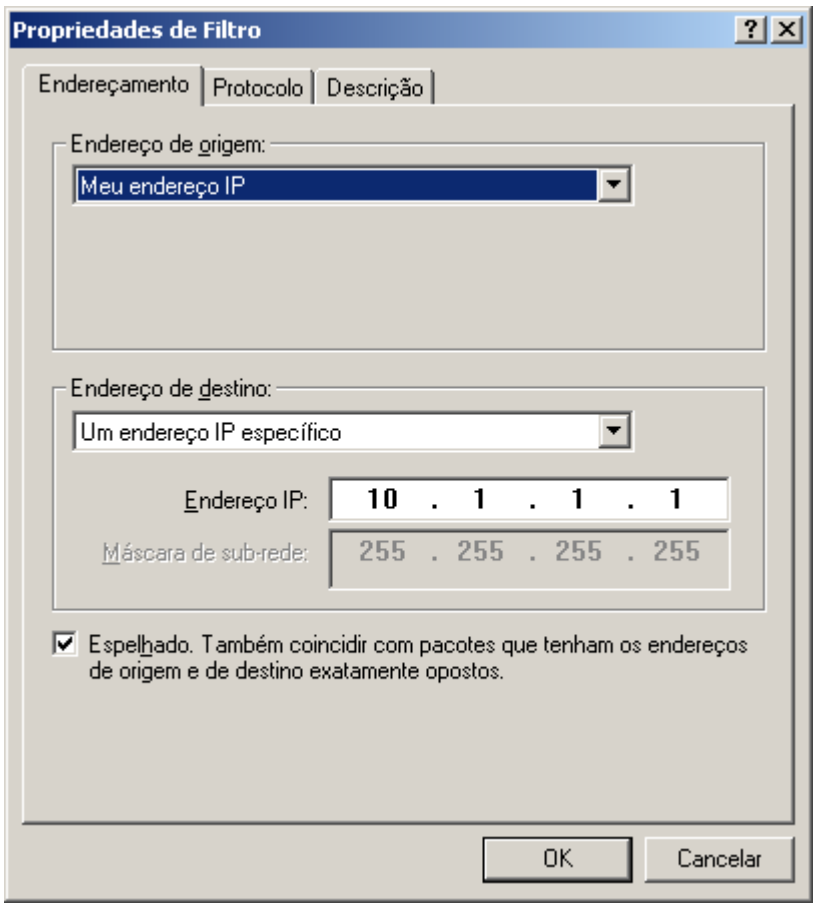

Figura 10: Configuração de fluxo de dados IPsec.

Ao clicar em OK para confirmar a regra definida, a mesma deve agora ser selecionada nas listas de filtros IP, conforme ilustra a figura abaixo:

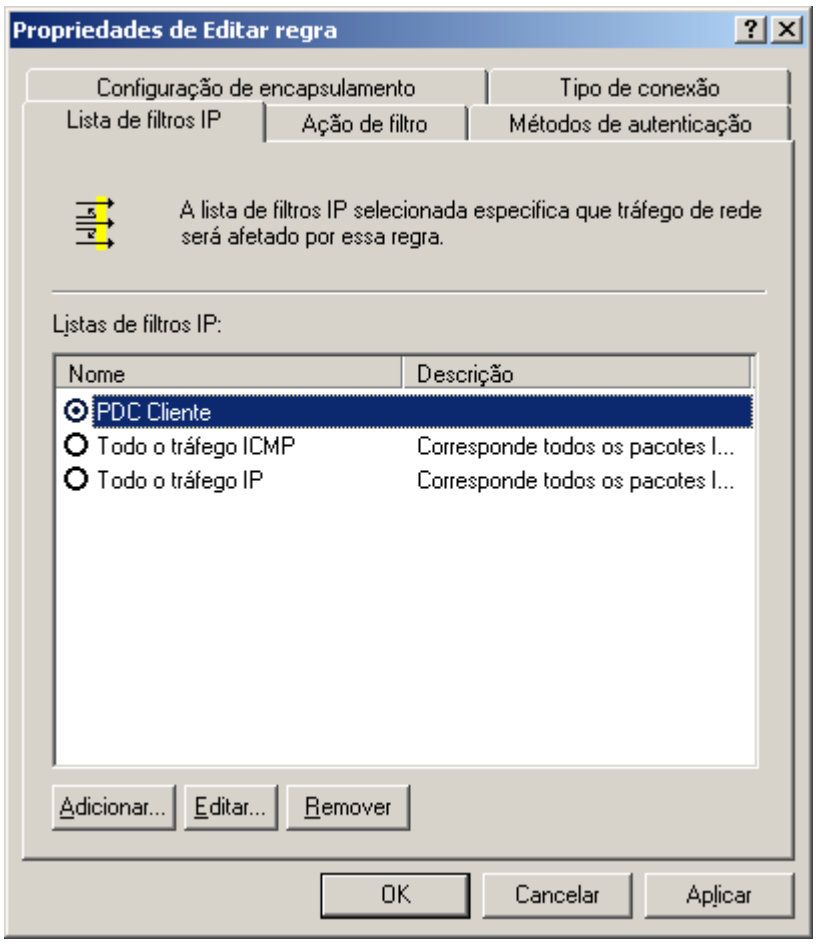

Figura 11: Seleção do filtro a ser utilizado.

Em seguida, na aba Ação de Filtro, seleciona-se a opção Exigir segurança. A autenticação é escolhida na aba Métodos de autenticação, clicando-se no botão Adicionar, e selecionando-se a autoridade de certificação desejada (pdc.exemplo.br), a qual é disponibilizada após a instalação do certificado digital do cliente.

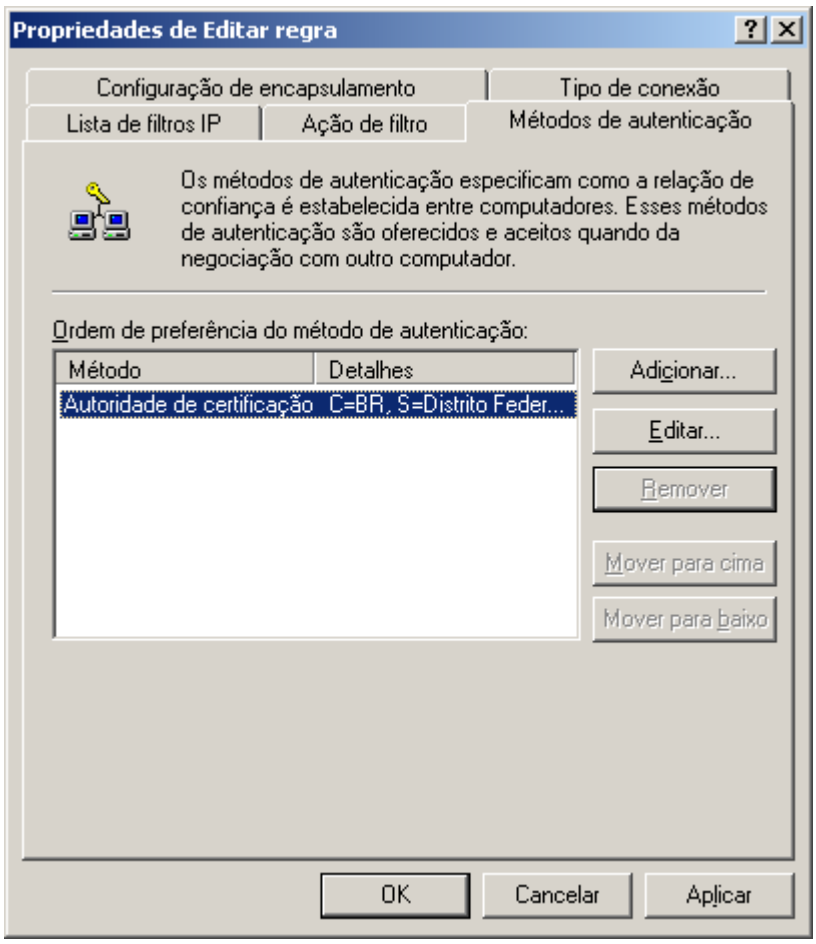

Figura 12: Escolha do método de autenticação.

Com todas essas operações realizadas, confirma-se as seleções, e de volta ao menu principal do mmc, clica-se com o botão direito na opção Tráfego PDC cliente, e seleciona-se a opção Atribuir, para colocar em efeito as configurações de IPsec.

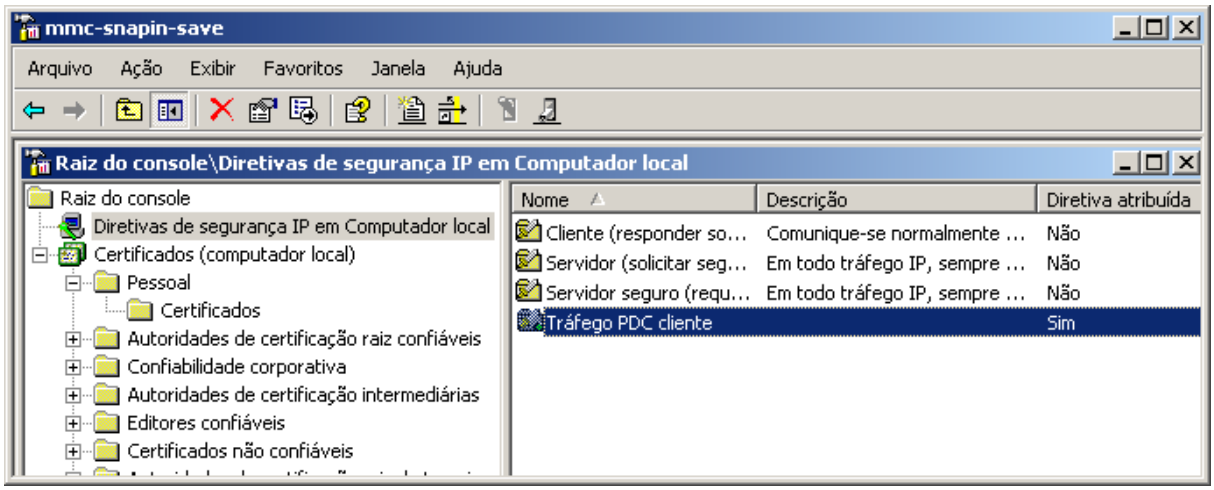

Figura 13: Configuração de tráfego IPsec ativada.

A partir deste momento, toda comunicação feita pelo cliente ao PDC será através de IPsec, incluindo protocolos como o SMB/CIFS. Um teste rápido pode ser efetuado utilizando o comando ping (que utiliza o protocolo ICMP) no *prompt* do Windows para verificar a conexão. Uma mensagem de negociação de segurança IP aparece, seguida da resposta ao comando ping:

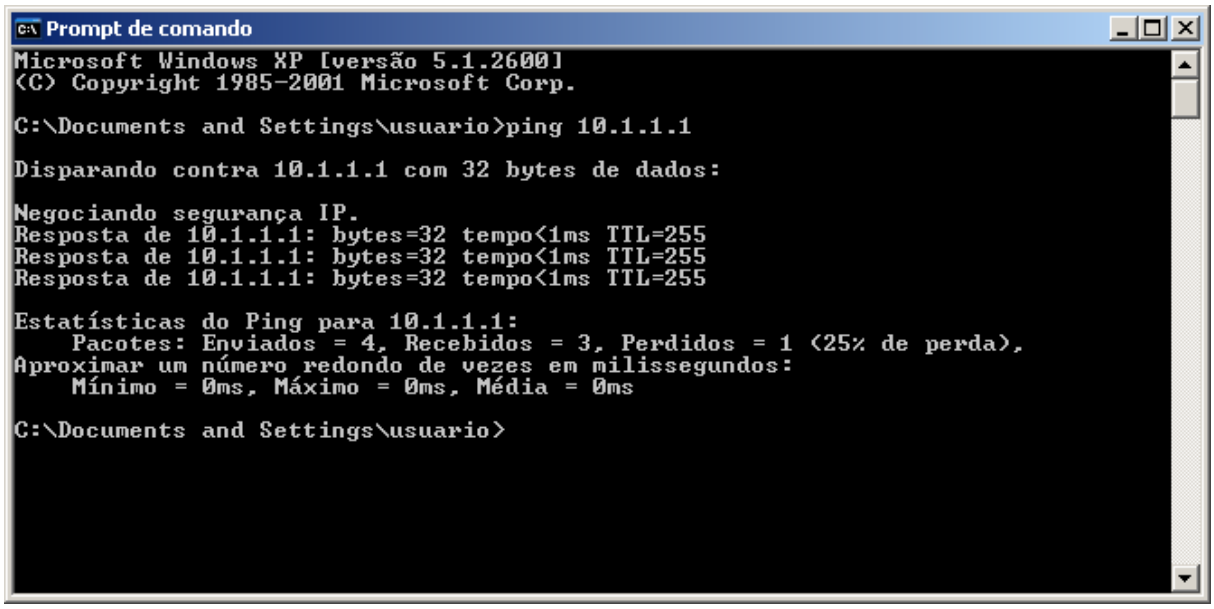

Figura 14: Verificando conexão IPsec.

5 GERENCIAMENTO DOS SISTEMAS

A manutenção de sistemas é um dos pontos que definem o sucesso ou insucesso de um ambiente computacional. De nada adianta utilizar os melhores *softwares* e os melhores *hardwares* se a manutenção dos sistemas for complicada, gerando gastos excessivos de tempo, altas possibilidades de erro, ou até mesmo proibitiva, o que inviabiliza todo o projeto.

Na execução de tarefas de manutenção, é importante ter sempre cópias de segurança dos dados em questão. Atualizações de *softwares*, *hardwares* e sistemas operacionais são pontos críticos do processo de manutenção, e não devem ser executadas sem a realização de *backups* dos dados envolvidos, evitando assim a perda de informações importantes, ou pior, a paralisação dos serviços de TI, em casos de erros, procedimentos mal executados ou manutenções mal planejadas. Uma recomendação importante, porém eventualmente ignorada, é a realização de testes dessas atualizações em ambientes isolados, de preferência com as mesmas características do ambiente de produção. Isso reduz a possibilidade de paralisação dos serviços caso aconteça alguma falha não prevista após o processo de migração.

Os tópicos possuem o objetivo de apresentar aspectos relativos à manutenção do ambiente computacional proposto neste projeto, tais como considerações sobre o sistema operacional e os softwares utilizados, procedimentos de gerenciamento de computadores e usuários, e considerações sobre políticas de segurança.

5.1 Manutenção do sistema operacional (OpenBSD)

O OpenBSD possui um cronograma de lançamento de novas versões sempre a cada seis meses. Essa periodicidade é interessante para o administrador de sistemas, pois possibilita a

realização de um agendamento com antecedência para instalação de novas versões do sistema operacional.

O OpenBSD não é um sistema operacional que possui atualizações automáticas. O administrador deve se preocupar em visitar regularmente o site oficial do projeto para obter as últimas atualizações[17] de segurança ou disponibilidade. Outra opção é visitar regularmente o OpenBSD Journal[18], uma comunidade para usuários do OpenBSD que contém notícias diversas acerca do sistema operacional, tais como novas funcionalidades, problemas de segurança, novos aplicativos, etc.

5.2 Manutenção dos *softwares* utilizados

Os *softwares* utilizados para compor o ambiente computacional estão disponíveis pelo sistema de *ports* do OpenBSD. Uma forma de verificar as atualizações disponíveis para os mesmos, é visitar regularmente os sites referentes à cada *software* (por exemplo, <http://www.squid-cache.org> para o *software* Squid), e verificar se as atualizações já foram transportadas para a estrutura de *ports* do OpenBSD. O site OpenPorts.se[19] possui um repositório com todos os *softwares* disponíveis pela estrutura de *ports*, detalhando as atualizações ou modificações que são feitas regularmente.

É importante frisar que muitas vezes, a atualização do fabricante demora a aparecer no *software* disponível via *ports*. Dessa forma, o administrador não deve confiar plenamente na estrutura de *ports* para obter os softwares atualizados. Caso a atualização seja crítica, é recomendável que o próprio administrador realize a atualização, compilando uma nova versão para o *software*.

---

17 Disponível em: <http://www.openbsd.com/errata.html>. Acesso em 5 de julho de 2008.

18 Disponível em: <http://undeadly.org/>. Acesso em 5 de julho de 2008.

19 Disponível em: <http://openports.se/>. Acesso em 5 de julho de 2008.

5.3 Administração de computadores clientes e contas de usuário

Os tópicos seguintes apresentam as tarefas mais comuns de administração de computadores e usuários para o ambiente computacional deste projeto, assim como considerações sobre a criação e instalação de políticas de segurança.

5.3.1 Procedimentos para inclusão de máquinas clientes no domínio

Com o ambiente Samba configurado, resta ao administrador de sistemas incluir todos os computadores clientes ao domínio criado. Uma ver realizada a inclusão, é possível efetuar um controle centralizado do ambiente computacional para funções como configuração de contas de usuário, perfis e políticas de segurança.

Por padrão, o Windows XP requisita endereços ao servidor DHCP local. Como o servidor de DHCP foi previamente configurado, este enviará automaticamente à máquina cliente os dados básicos de rede, como endereço IP, máscara de rede, endereço de *gateway* e endereços de servidores DNS. É recomendável verificar se os dados foram entregues corretamente, utilizando um *prompt* de comando do Windows, disponível em Iniciar, Programas, Acessórios, Prompt de Comando, e executando o seguinte comando:

```
C:\ ipconfig /all
```

A imagem a seguir demonstra a configuração de uma máquina cliente para a rede descrita neste projeto:

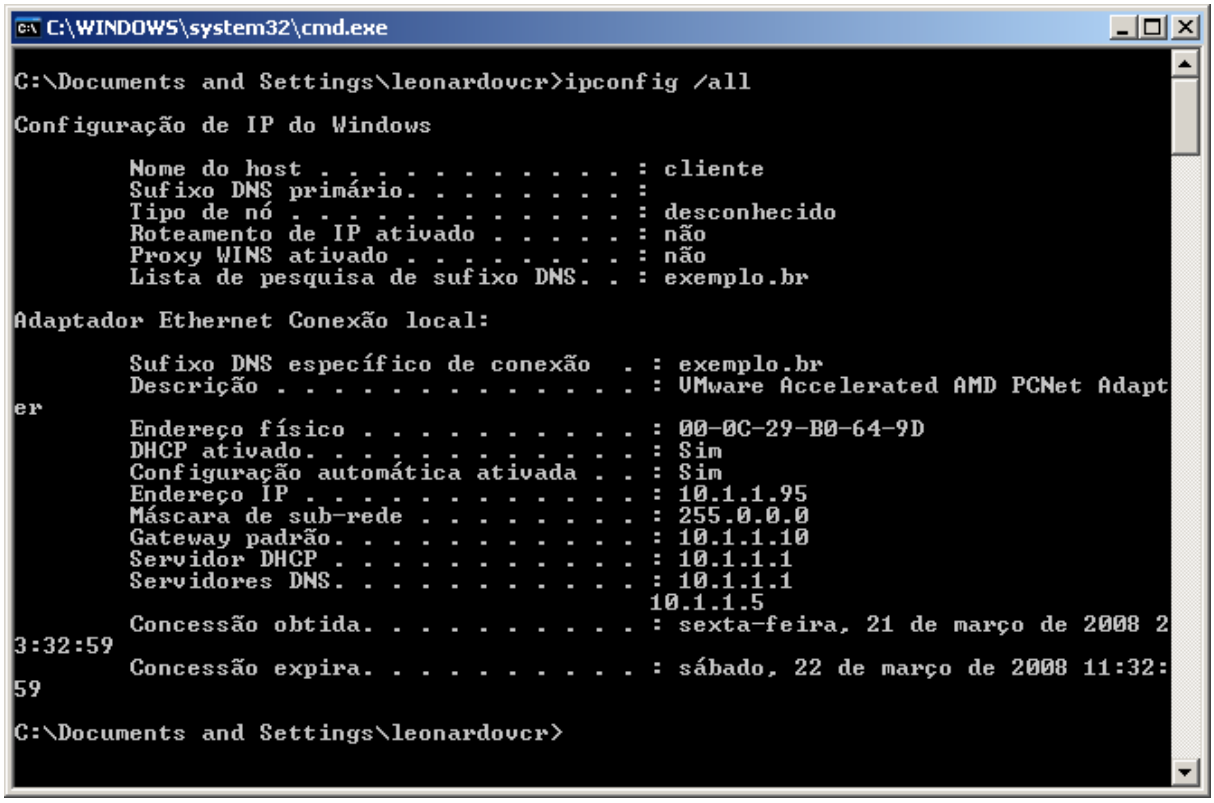

Figura 15: Verificação do DHCP

Com os dados verificados, o administrador pode adicionar o computador ao domínio através do botão "Alterar" localizado em Painel de Controle, Sistemas, aba Nome do Computador. As informações que devem ser preenchidas são o nome do domínio, o nome do computador e os dados do usuário com permissão para efetuar o ingresso do computador no domínio (o administrador da rede por exemplo).

Será então criada uma conta relativa ao computador adicionado, podendo a mesma ser verificada no arquivo /etc/passwd (neste caso, a conta do computador fica no formato nome$) ou usando a ferramenta Server Manager, conforme ilustra a figura a seguir:

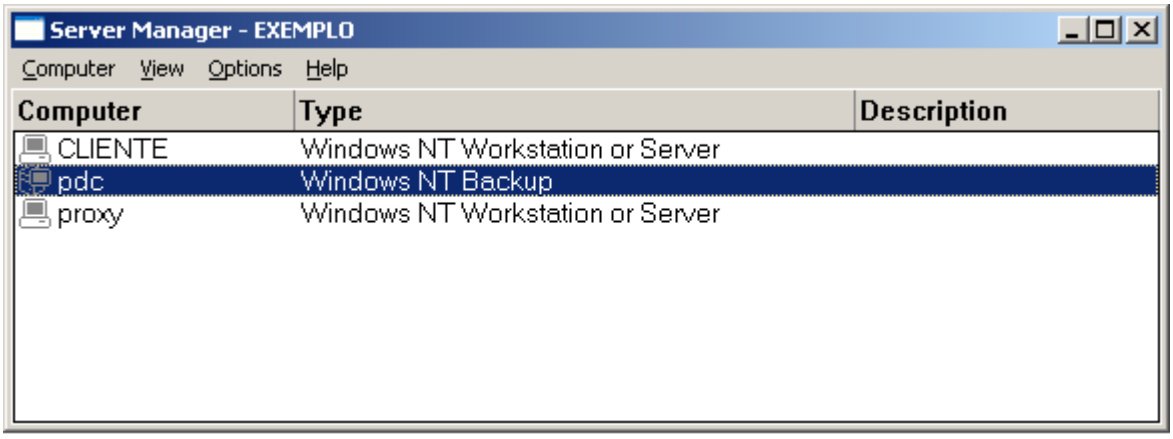

Figura 16: Visualizando os computadores do domínio

É importante lembrar que, caso o computador seja formatado, ou retirado do domínio, deve-se excluir sua conta de domínio antes de adicioná-lo novamente. Para isto, basta deletá-lo utilizando a ferramenta citada acima.

5.3.2 Gerenciamento de contas e grupos de usuários

Conforme demonstrado no capítulo 4.2.1.1, é possível utilizar a ferramenta User Manager para gerenciar usuários e grupos de usuários no domínio. A seguinte figura demonstra a principal tela disponível para o administrador:

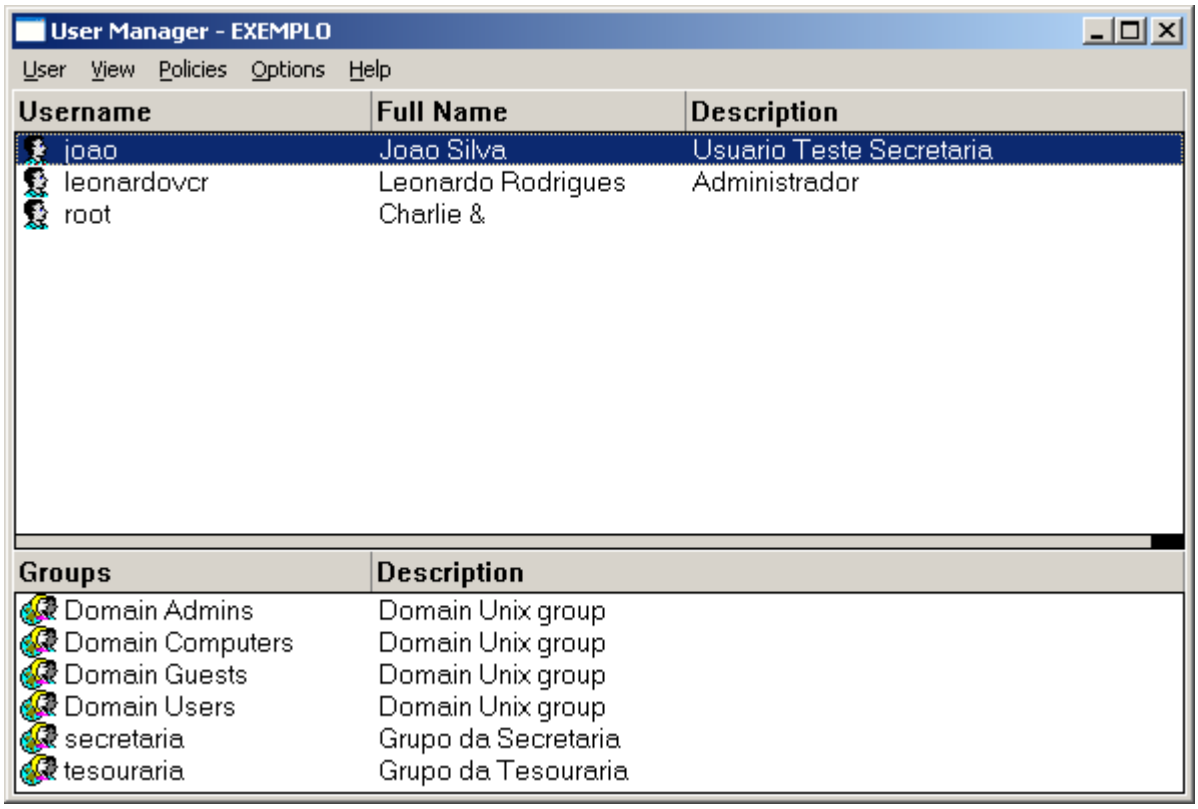

Figura 17: Visualizando os usuários e grupos do domínio.

É possível visualizar graficamente os usuários e os grupos do domínio, e renomear, remover ou criar usuários e/ou grupos. Cada usuário possui também uma tela própria, onde é possível editar características como nome de usuário, senha, nome completo, descrição, horários de logon, perfil a ser utilizado, grupos pertencentes e computadores onde o mesmo tem permissão para efetuar logon. A seguinte figura demonstra a utilização desta tela de propriedades para um usuário:

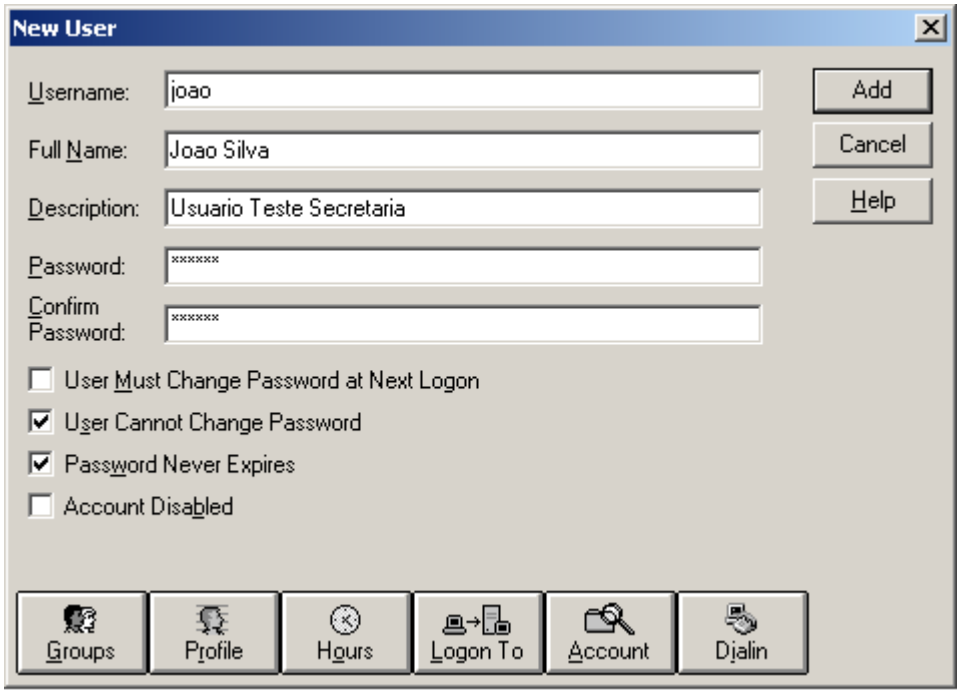

Figura 18: Inclusão de um novo usuário no domínio.

5.3.3 Gerenciamento de políticas de segurança

As políticas de segurança, ou diretivas de sistema, são uma das principais ferramentas disponíveis para efetuar um controle refinado sobre as mais diversas variáveis do sistema operacional Windows. Sua utilização permite especificar quais serviços do sistema devem ou não estar disponíveis para o usuário, tais como o Painel de Controle, ou as Propriedades de Vídeo. As políticas de segurança também permitem a ativação de configurações pré-estabelecidas pelo administrador, tal como a configuração de *proxy* padrão elaborada para este projeto, mencionada no capítulo 4.2.1.3 e descrita nos apêndices desta monografia.

A elaboração de uma boa política de segurança deve levar em conta o perfil da empresa e de seus funcionários. Deve-se levantar as necessidades e o perfil de cada usuário, tais como professores, diretores ou atendentes, e idealizar uma política para ser então implementada no sistema. Professores provavelmente não precisam ter acesso ao *prompt* de comando, no entanto, este acesso é necessário para a maioria dos funcionários que efetuam o suporte técnico. O administrador deve se preocupar em criar políticas separadas para cada grupo funcional, ou até mesmo políticas individuais para certos funcionários caso necessário. Um bom material de consulta seria o livro “Microsoft Windows Group Policy Guide”, escrito por

Darren Mar-Elia, Derek Melber, William Stanek, et al. Vale ressaltar que este livro cobre a tecnologia de Group Policy presente no Active Directory do Windows 2003 Server. Este projeto implementa um modelo de políticas de segurança anterior, utilizado pelo Windows NT. As duas tecnologias possuem diferenças, porém, os princípios que devem ser levados em conta na elaboração de uma política de segurança pouco se alteram. Neste projeto, o administrador é livre para implementar políticas mesmo estas não estando disponíveis por padrão no computador Windows cliente, sendo possível criá-las conforme explicado no capítulo 4.2.1.3 desta monografia. Este procedimento permite criar muitas das políticas entregues por um servidor Windows 2003 Server, porém indisponíveis em clientes Windows.

A ferramenta utilizada para o gerenciamento de políticas de segurança é o POLEDIT.EXE. A figura abaixo ilustra a tela principal dessa ferramenta:

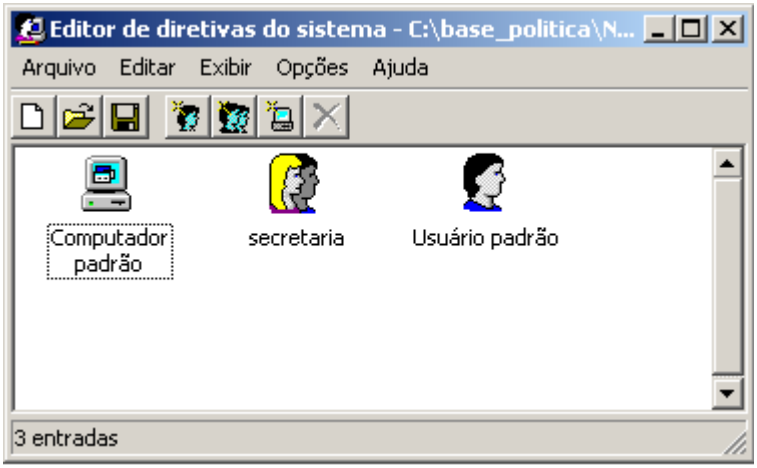

Figura 19: Gerenciador de políticas.

É possível especificar políticas aplicáveis à computadores, grupos de usuário ou usuários. A figura acima ilustra que o modelo de política criado possui, além de políticas para computador e usuário padrão, que são aplicáveis à todos os usuários e computadores, uma política específica para usuários do grupo secretaria. Um exemplo de configuração seria a especificação de um endereço de *proxy* padrão para os usuários do grupo secretaria, evitando que os usuários mudem o *proxy* de seus computadores (e saim do controle de acesso à Internet). A figura a seguir ilustra esta opção:

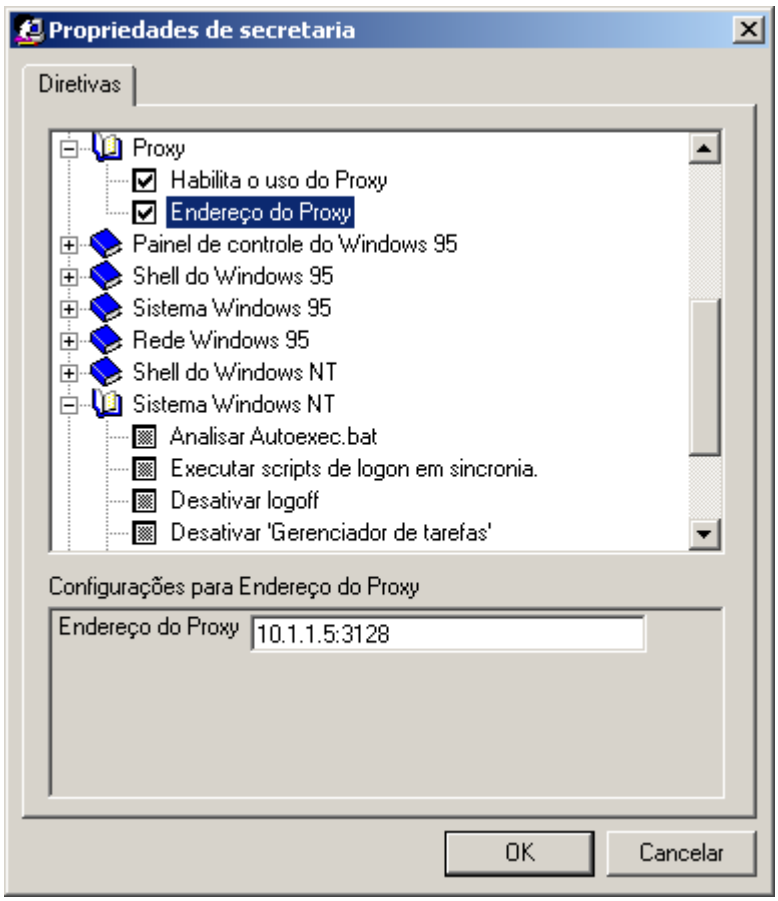

Figura 20: Política para ativação de um endereço de *proxy*.

As configurações disponíveis para grupos de usuário são as mesmas que estão disponíveis para usuários em específico. Já as configurações de políticas para computadores diferem das configurações de usuários e grupos por possuírem opções maior abrangência, que serão aplicáveis à todos os usuários que utilizarem aquele computador em específico. A figura a seguir ilustra uma opção de configuração relativa à um computador:

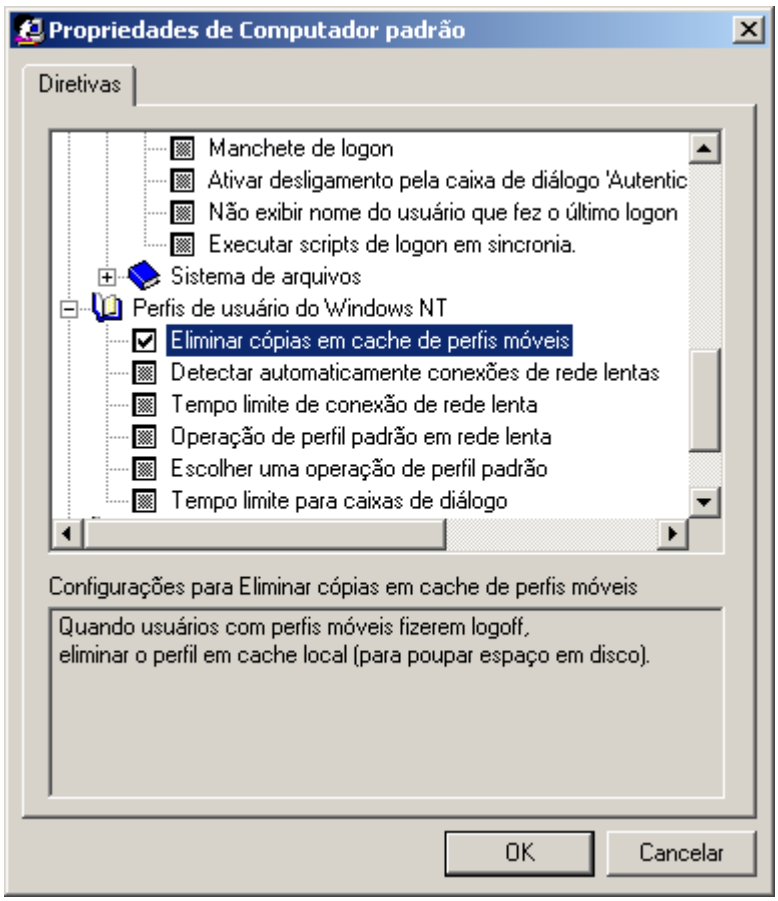

Figura 21: Exemplo de política aplicável à um computador.

6 ANÁLISE DO CENÁRIO PROPOSTO

O objetivo deste capítulo é apresentar as atividades que foram necessárias para a realização deste projeto, as dificuldades encontradas, como estas foram vencidas, e as funcionalidades que foram obtidas na solução final.

Financeiramente, vale ressaltar que devido à utilização exclusiva de *softwares* livres na solução final, caso esta seja escolhida para substituir servidores Windows 2003 Server, uma instituição de médio porte com cerca de 50 máquinas e um servidor pode economizar cerca de US$ 2.800,00 em licenças de produtos, conforme lista a tabela de preços para licenciamento disponível na página da Microsoft[20]. Esse preço aumentaria ainda mais caso fosse incluído também uma licença para servidor *proxy* Microsoft ISA Server. No caso específico da instituição de ensino apresentada neste projeto, que possui cerca de 180 máquinas, a economia financeira é ainda maior.

6.1 Dificuldades encontradas

A implementação do modelo computacional proposto apresentou vários obstáculos. Primeiramente, a escolha do OpenBSD como sistema operacional implica em alguns problemas de compatibilidade e integração. Devido à grande popularidade do sistema operacional GNU/Linux no meio dos *softwares* livres, alguns destes projetos de *software* possuem seu desenvolvimento voltado à este sistema, ou o tem como referência. Dessa forma, alguns *softwares* não funcionam corretamente em outros sistemas tipo Unix, como o OpenBSD, sendo necessário então algum trabalho por parte do administrador para integrá-los com sucesso.

---

20 Disponível em: <http://www.microsoft.com/windowsserver2003/howtobuy/licensing/pricing.mspx>.

O principal obstáculo de integração deste projeto veio com a utilização do *software* Samba junto ao OpenBSD. Apesar de pacotes de instalação do Samba estarem disponíveis para este sistema operacional, os mesmos não possuem habilitadas algumas extensões necessárias (como o winbind, necessário para o processo de autenticação do Proxy). Além disso, algumas configurações específicas do Samba fazem referências a comandos utilizados em sistemas GNU/Linux, sendo necessário uma modificação destas por parte do administrador.

Com relação ao Samba, os principais obstáculos de integração podem ser listados como:

- Criação de um pacote customizado, através da compilação de extensões de *software* pelo mecanismo de *ports* do OpenBSD, visando a utilização de ferramentas que não estão disponíveis por padrão (como o winbind) no sistema de *packages* e *ports* deste sistema operacional.
- Criação de *scripts* de manuseio de dados de contas de usuário, adequados ao sistema OpenBSD, referenciando-os nas configurações do Samba para que o mesmo permita a utilização de interfaces gráficas de gerenciamento, objetivando a facilidade de administração.
- Uma das ferramentas referenciadas pelo Samba em sua configuração (*delete user from group script*) não possui semelhantes no OpenBSD. Logo, uma ferramenta para desempenhar tal tarefa precisava ser criado.
- Edição geral do arquivo de configuração visando refletir o cenário proposto, criando entradas para a utilização de impressoras, políticas de segurança e diretórios de arquivos.
- Criação de uma política de segurança utilizável pelo Samba, para especificação de servidor *proxy* a ser utilizado no computador cliente, visto que o Windows XP não dispõe de uma, diferentemente do Windows 2003 Server, que possui essa política por padrão.

O servidor Proxy por sua vez, utiliza o *software* Squid para realizar as funções de *web* e *caching proxy*, juntamente com o *software* SquidGuard para prover controle de acesso. A autenticação é feita pelo Squid, em conjunto com a extensão ntlm_auth e winbind, fornecidas através do *software* Samba. Para que todas essas ferramentas sejam integradas, e funcionem a contento, as seguintes atividades tiveram que ser realizadas:

- Edição das configurações do *software* Squid, visando a utilização do SquidGuard como motor de controle de acesso, e a autenticação de usuários a partir da base de dados no servidor Proxy.
- Criação de uma configuração para o SquidGuard para obter controle de acesso, utilizando o princípio da liberação ou do bloqueio de endereços na Internet a partir de uma lista de sites especificados pela instituição, e por uma lista geral obtida na Internet, para filtrar sites maliciosos ou propagandas.
- Criação de um *script* para gerar listas de grupos de usuários para o Proxy, a partir dos grupos listados pelo winbind na base de usuários do servidor PDC, viabilizando o controle de acesso à Internet por grupos de usuários no SquidGuard.
- Criação de um *script* para buscar e atualizar automaticamente a lista geral de endereços na Internet.

Por último, uma estrutura de chaves públicas (PKI) teve que ser implementada para a utilização do IPsec, visando mitigar o problema de segurança existente no protocolo SMB/CIFS. As seguintes atividades tiveram que ser realizadas:

- Criação de uma PKI, utilizando o servidor PDC como autoridade certificadora.
- Criação de certificados digitais X.509 para o servidor PDC e para um computador cliente.
- Exportação de um certificado digital tipo PKCS-12 específico para utilização no Windows XP do computador cliente.
- Criação de regras para os fluxos de IPsec, tanto no OpenBSD (servidor PDC), quanto no Windows XP (computador cliente).

6.2 Funcionalidades obtidas

A solução final apresenta duas vertentes de funcionalidades obtidas, que podem ser classificadas em funcionalidades de administração e funcionalidades de segurança.

6.2.1 Administração

Conforme apresentado nos capítulos 4 e 5 deste projeto, a solução final apresenta funcionalidades de administração similares às existentes em um servidor Windows 2003 Server com *Active Directory*. As seguintes funcionalidades administrativas foram obtidas:

- Relação automática de nomes de rede de clientes à endereços IP distribuídos via DHCP, atualizando o servidor DNS de forma dinâmica.
- Administração de contas, máquinas, e grupos de usuários, através de uma interface gráfica.
- Gerenciamento centralizado de arquivos e de arquivos de impressão.
- Gerenciamento de perfis de usuários.
- Gerenciamento de políticas de segurança, com a possibilidade de se criar políticas customizadas (conforme demonstrado com a criação de uma para definição de servidor Proxy).
- Controle de acesso à Internet baseado em usuários ou grupo de usuários, além de *caching* de dados para otimizar a banda utilizada.

As funcionalidades acima podem ser consideradas suficientes para as necessidades de Tecnologia de Informação de várias empresas ou organizações, visto que cobrem serviços básicos como controle de usuários, arquivos, impressão e controle de acesso à Internet.

6.2.2 Segurança

Com relação ao cenário apresentado na seção 2.1, os problemas de segurança encontrados foram tratados. A autenticação de usuários pelo servidor Proxy é feita de modo seguro, utilizando-se do protocolo NTLM para realizar o tráfego de dados de autenticação de forma criptografada.

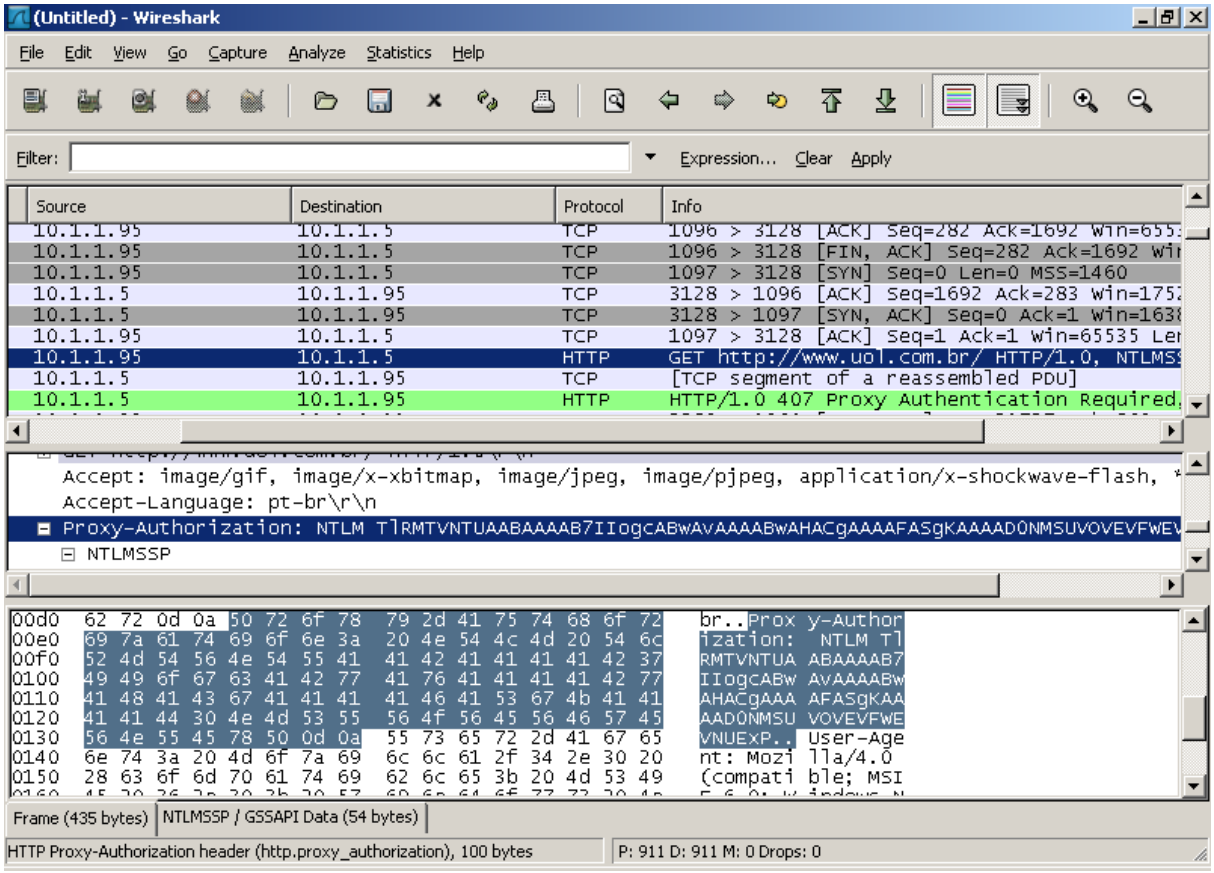

Figura 22: Servidor Proxy utilizando NTLM.

O teste demonstrado acima foi realizado efetuando-se *login* em um computador cliente, com o nome de usuário “joao” e a senha “123456”. Estes dados foram então automaticamente enviados ao servidor Proxy com a abertura de uma sessão em um navegador de Internet, requisitando o endereço <http://www.uol.com.br>. Com essa automatização, resolve-se o problema de segurança anteriormente encontrado, que era a possibilidade de se utilizar uma senha diferente da senha de logon para o acesso à Internet. O outro problema de segurança tratado foi o tráfego inseguro de informações de autenticação, apresentado na seção 3.3. Conforme ilustra a figura apresentada, as informações de autenticação agora trafegam de

forma criptografada, utilizando o protocolo NTLM, aumentando consideravelmente a segurança com relação ao método de autenticação anteriormente utilizado.

A questão da confidencialidade do protocolo SMB/CIFS foi resolvida utilizando-se a estrutura IPsec para encapsular o tráfego de rede. Para fins de comparação, as duas figuras seguintes ilustram um mesmo processo de transferência de arquivos. Na primeira o IPsec não é utilizado, sendo este ativado na segunda figura.

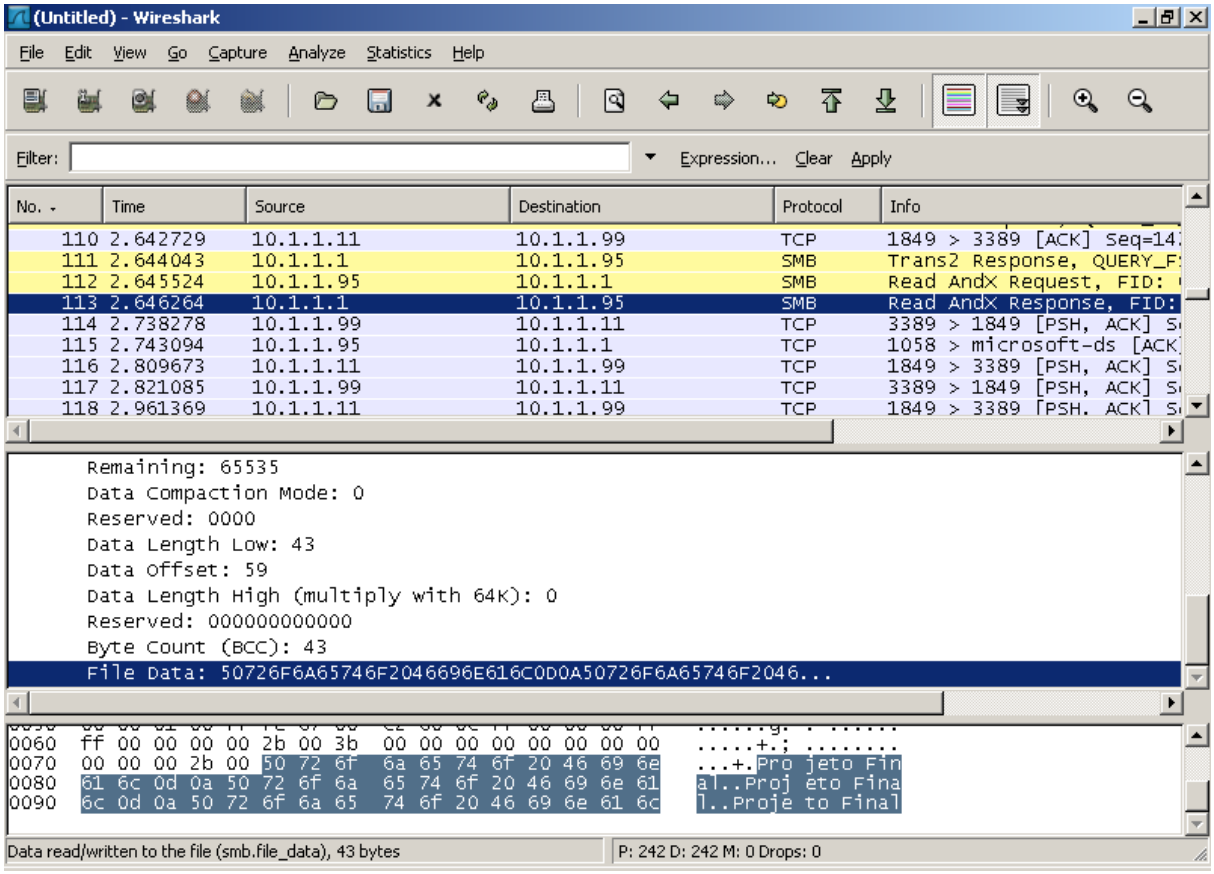

Figura 23: Dados trafegando sem IPsec.

Na figura acima, as informações trafegam sem IPsec pela rede. Nota-se que o conteúdo do arquivo sendo transferido do servidor (10.1.1.1) para o cliente (10.1.1.95) pode ser facilmente visualizado (texto grafado na cor azul). Este é o método de tráfego de dados utilizado pelo cenário apresentado na seção 2.1.

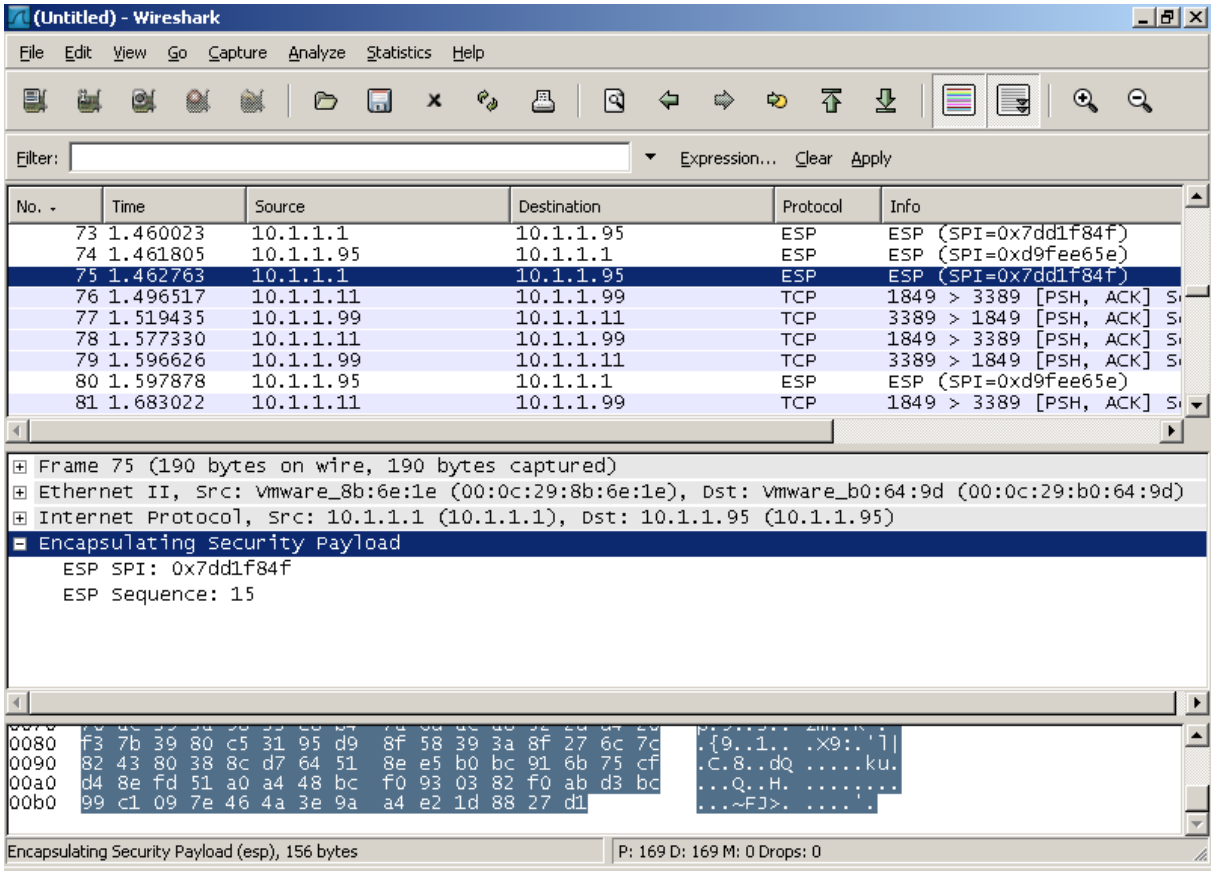

Figura 24: Dados trafegando com IPsec.

Esta segunda figura apresenta o mesmo tráfego de dados da figura anterior, porém com o IPsec ativado. Nota-se que o tráfego SMB/CIFS sequer consta na lista de pacotes capturados pelo wireshark, aparecendo no lugar os pacotes ESP, que encapsulam o tráfego entre o cliente (IP 10.1.1.95) e o servidor (IP 10.1.1.1). O conteúdo dos pacotes ESP é criptografado, protegendo assim todo o tráfego por ele encapsulado.

CONCLUSÃO

O presente trabalho demonstra a possibilidade de se utilizar um ambiente de servidores tipo Unix para gerenciar clientes Microsoft, com implementações para melhorar a integração e a segurança do ambiente computacional. A solução final pode ser utilizada pelas mais variadas empresas que gostariam de obter uma solução isenta de custos de licenciamento para o gerenciamento de suas informações.

Um dos maiores desafios do projeto é a integração entre os diferentes *softwares* e o sistema operacional escolhido. Algumas ferramentas tiveram que ser criadas para suprir certas faltas de funcionalidade apresentadas por alguns *softwares* e pelo sistema operacional. Por fim, quanto à segurança, é demonstrado a possibilidade de criptografar o tráfego de rede entre clientes e servidores, além de uma melhor integração, transparente e segura, entre clientes e servidor *proxy*. Boas práticas de administração de sistemas também foram abordadas durante a implementação, tais como permissões de usuários e procedimentos administrativos. Com relação aos aspectos administrativos, foi apresentado que o ambiente de servidores tipo Unix implementado fornece funcionalidades básicas para clientes Windows, como gerenciamento de arquivos, contas, impressão, perfis, políticas de segurança, e inclusive controle de acesso à Internet.

Vale lembrar que, financeiramente, este projeto se preocupou apenas com o custo inicial de implementação, que equivale à aquisição de licenças de *softwares*. Custos adicionais como manutenção e suporte, tais como avaliados por um estudo de *Total Cost of Ownership*, não foram considerados.

REFERÊNCIAS BIBLIOGRÁFICAS

BARR, David. **RFC 1912, Common DNS Operational and Configuration Errors**. Disponível em: <http://rfc.net/rfc1912.html>. Acesso em 20 abr. 2008.

BATTISTI, Júlio. **Windows Server 2003 Curso Completo**. Rio de Janeiro: Axcel Books, 2003.

CYBERSOURCE. **Total Cost of Ownership**. Documento digital disponível em: <http://www.cybersource.com.au/about/linux_vs_windows_tco_comparison.pdf>. Acesso em 5 jul. 2008.

HERTEL, Christopher. **Implementing CIFS: The Common Internet File System**. Upper Saddle River: Prentice Hall, 2003.

IBM. **IBM Rational Performance Tester on Windows 2000 and Linux**. Disponível em: <http://www-128.ibm.com/developerworks/lotus/library/ls-Rational_Win32_Linux>. Acesso em 5 jul. 2008.

JOHANSSON, Jesper. **How to Shoot Yourself in the Foot with Security, Part 1**. Disponível em: <http://www.microsoft.com/technet/community/columns/secmgmt/sm0905.mspx>. Acesso em 13 mar. 2008.

JONES, JEFF. **Windows vs Linux**. Disponível em: <http://blogs.technet.com/security/archive/2006/07/13/441386.aspx>. Acesso em 5 jul. 2008.

KLEIN, Amit. **BIND 9 DNS Cache Poisoning.** Documento digital disponível em: <http://www.trusteer.com/docs/BIND_9_DNS_Cache_Poisoning.pdf>. Acesso em 17 jan. 2008.

LEACH, Paul. **CIFS Security Considerations, Author's Draft 4**. Documento digital disponível em: <ftp://ftp.microsoft.com/developr/drg/CIFS/CIFS-Security-Considerations.txt>. Acesso em 28 fev. 2008.

LUCAS, Michael. **Absolute OpenBSD: Unix for the Practical Paranoid**. San Francisco: No Starch Press, 2003.

MAZZOCCHIO, Daniele. **OPENBSD as a Domain Name Server**. Documento digital disponível em: <http://www.kernel-panic.it/openbsd/dns/OpenBSD_dns_server.pdf>. Acesso em 28 fev. 2008.

MAZZOCCHIO, Daniele. **OPENBSD as a Primary Domain Controller**. Documento digital disponível em: <http://www.kernel-panic.it/openbsd/pdc/OpenBSD_pdc.pdf>. Acesso em 28 fev. 2008.

MICROSOFT. **Windows Server 2003 R2 Pricing**. Disponível em: <http://www.microsoft.com/windowsserver2003/howtobuy/licensing/pricing.mspx>. Acesso em 5 jul. 2008.

MICROSOFT. **Visão Geral da Assinatura do Bloco de Mensagens de Servidor (SMB)**. Disponível em: <http://support.microsoft.com/kb/887429>. Acesso em 5 jul. 2008.

NETAPPLICATIONS. **Operating System Market Share**. Disponível em: <http://marketshare.hitslink.com/report.aspx?qprid=10&qpmr=15&qpdt=1&qpct=3&qptimeframe=M&qpsp=108>. Acesso em 5 jul. 2008.

ONESTAT. **Website Statistics and Website Metrics**. Disponível em: <http://www.onestat.com/html/aboutus_pressbox46-operating-systems-market-share.html>. Acesso em 5 jul. 2008.

OPENBSD. **OpenBSD Security**. Disponível em: <http://www.openbsd.org/security.html>. Acesso em 28 fev. 2008.

OPENBSD. **The OpenBSD Project Man Pages**. Disponível em: <http://www.openbsd.org/cgi-bin/man.cgi>. Acesso em 28 fev. 2008.

OPENBSDJOURNAL. **OpenBSD Journal – A Resource for the OpenBSD Community**. Disponível em: <http://undeadly.org/>. Acesso em 28 fev. 2008.

OPENGROUP. **The Open Group – Making Standards Work - NTLM**. Disponível em <http://www.opengroup.org/comsource/techref2/NCH1222X.HTM>. Acesso em 5 jul. 2008.

OPENPORTS. **The OpenBSD Package Collection**. Disponível em: <http://openports.se/>. Acesso em 29 fev. 2008.

PALMER, Brandon, NAZARIO, Jose. **Secure Architectures with OpenBSD - 1st Edition**. Boston: Addison-Wesley, 2004.

PCWORLD. **Study: Linux is Still Cheaper than Windows**. Disponível em: <http://www.pcworld.com/news/article/0,aid,118937,00.asp>. Acesso em 5 jul. 2008.

PETRELEY, Nicholas. **Security Report: Windows vs Linux**. Disponível em: <http://www.theregister.co.uk/security/security_report_windows_vs_linux/>. Acesso em 5 jul. 2008.

ROCHA, José Antonio Meira da. **Modelo de Trabalho de Conclusão de Curso (TCC).** Modelo de documento digital do programa OpenOffice 2.0 disponível em: <http://www.meiradarocha.jor.br>. Acesso em 12 jun. 2006.

SAMBA. **The Samba Project**. Disponível em: <http://www.samba.org>. Acesso em 29 fev. 2008.

SAPRONOV, Konstantin. **2005: *nix Malware Evolution**. Disponível em: <http://www.viruslist.com/en/analysis?pubid=184625030>. Acesso em 5 jul. 2008.

SECURECOMPUTING. **Malware Threats & Trends Alert**. Disponível em: <http://www.securecomputing.com/index.cfm?skey=1739>. Acesso em 5 jul. 2008.

SHALLASECURESERVICES. **Shalla's Blacklist Status Overview**. Disponível em: <http://www.shallalist.de/cgi-bin/stat.cgi>. Acesso em 5 jul. 2008.

SNI. Secure Networks Inc. and CORE Seguridad de la Información. **BIND Vulnerabilities and Solutions.** Documento digital disponível em: <http://www.openbsd.org/advisories/res_random.txt>. Acesso em 17 jan. 2008.

SNIA. Storage Networking Industry Association. **Common Internet File System Technical Rerefence Rev. 1.0.** Documento digital disponível em: <http://www.snia.org/tech_activities/CIFS/CIFS-TR-1p00_FINAL.pdf>. Acesso em 22 fev. 2008.

SQUID. **Squid, Optimizing Web Delivery**. Disponível em: <http://www.squid-cache.org>. Acesso em 29 fev. 2008.

SQUIDGUARD. **SquidGuard, a Combined Filter, Redirector and Access Controller for Squid**. Disponível em: <http://www.squidguard.org>. Acesso em 29 fev. 2008.

TANENBAUM, Andrew. **Redes de Computadores – 4ª Edição**. Rio de Janeiro: Elsevier, 2003.

TERPSTRA, John. **Samba-3 by Example - 2**nd **Edition**. Stoughton: Prentice Hall, 2005.

APÊNDICES

A – *Scripts* de gerenciamento criados para utilização no Samba:

addusergroup.sh:

```
#!/bin/sh
## Script para adicionar um usuário à um grupo
## Sintaxe: $1 = grupo
##          $2 = usuário
/usr/sbin/usermod -G $1 $2
exit 0
```

groupadd.sh:

```
#!/bin/sh
## Script para criação de grupo
/usr/sbin/groupadd $1
exit 0
```

groupdel.sh:

```
#!/bin/sh
## Script para remover um grupo
/usr/sbin/groupdel $1
exit 0
```

machineadd.sh:

```
#!/bin/sh
## Script para criação de contas de computador
/usr/sbin/useradd -g DomainComputers -d /dev/null -s /sbin/nologin
$1
exit 0
```

renameuser.sh:

```
#!/bin/sh
## Script para renomear um usuário
## Sintaxe: $1 = nome novo
##          $2 = nome antigo
/usr/sbin/usermod -l $1 $2
exit 0
```

setprimarygroup.sh:

```
#!/bin/sh
## Script para alterar o grupo primário de um usuário
## Sintaxe: $1 = grupo
##          $2 = usuário
/usr/sbin/usermod -g $1 $2
exit 0
```

useradd.sh:

```
#!/bin/sh
## Script para criação de usuários
/usr/sbin/useradd -g DomainUsers -s /sbin/nologin -m $1
exit 0
```

userdel.sh:

```
#!/bin/sh
## Script para remoção de usuários
/usr/sbin/userdel -r $1
exit 0
```

rmusergroup.sh

```
#!/bin/sh
## Script para remover um usuário de um grupo
## Sintaxe: $1 = grupo
##          $2 = usuário
## Utilização: rmusergroup.sh <grupo> <usuário>

# 1o sed -> remove o usuário da linha do grupo
# 2o sed -> troca ,, por ,
# 3o sed -> remove , no fim da linha
# 4o sed -> troca :, por :

cd /etc
cat ./group \
        | sed '/'$1'/ s/'$2'//' \
        | sed '/'$1'/ s/,,/,/' \
        | sed '/'$1'/ s/,$//' \
        | sed '/'$1'/ s/:,/:/' > group.new

# Substitui o arquivo /etc/group pelo modificado, group.new
rm /etc/group
mv /etc/group.new /etc/group

# Ajusta permissões
chown root.wheel /etc/group
chmod 644 /etc/group
exit 0
```

B – Modelo de política criado para escolha de servidor Proxy:

proxy.adm:

```
CLASS USER

CATEGORY "Proxy"

	POLICY "Habilita o uso do Proxy"
	KEYNAME "Software\Microsoft\Windows\CurrentVersion\Internet
Settings"
		VALUENAME "ProxyEnable"
			VALUEON 1
			VALUEOFF 0
	END POLICY

	POLICY "Endereço do Proxy"
	KEYNAME "Software\Microsoft\Windows\CurrentVersion\Internet
Settings"
		PART "Endereço do Proxy" EDITTEXT
		VALUENAME "ProxyServer"
		END PART
	END POLICY

	END CATEGORY
```

C – Conteúdo final do arquivo smb.conf para o servidor PDC:

/etc/samba/smb.conf:

```
[global]
workgroup = EXEMPLO
netbios name = pdc
server string = Servidor Administrativo
passdb backend = tdbsam
passwd program = /usr/bin/passwd '%u'
username map = /etc/samba/smbusers
hosts allow = 10. 127.
domain logons = Yes
time server = Yes

name resolve order = host
idmap uid = 2000-4000
idmap gid = 2000-4000

# Logs (registros)
log level = 4
log file = /var/log/samba/smbd.%m

# Referências para os scripts criados
add user script = /etc/samba/scripts/useradd.sh '%u'
add user to group script = /etc/samba/scripts/addusergroup.sh '%g'
'%u'
add group script = /etc/samba/scripts/groupadd.sh '%g'
```

```
add machine script = /etc/samba/scripts/machineadd.sh %u
delete user script = /etc/samba/scripts/userdel.sh '%u'
delete user from group script = /etc/samba/scripts/rmusergroup.sh
'%g' '%u'
delete group script = /etc/samba/scripts/groupdel.sh '%g'
rename user script = /etc/samba/scripts/renameuser.sh '%unew'
'%uold'
set primary group script = /etc/samba/scripts/setprimarygroup.sh
'%g' '%u'

# Impressão
 load printers = yes
 printing = cups
 printcap name = cups


[NETLOGON]
    comment = Servico de Logon de Rede
    path = /home/NETLOGON
    guest ok = yes
    writable = no
    share modes = no

[profiles]
     comment = Perfil de Usuarios
     path = /home/profiles
     browseable = no
     guest ok = yes

[psc1310-1]
     comment = HP PSC 1310
     valid users = root @DomainUsers
     path = /var/spool/samba/printing
     printer = psc1310-1
     public = no
     writable = no
     printable = yes

[print$]
     comment = Drivers de Impressao
     path = /etc/samba/drivers
     browseable = no
     guest ok = no
     read only = yes
            write list = root
```

D – Conteúdo final do arquivo rc.local para o servidor PDC:

/etc/rc.local:

```
#       $OpenBSD: rc.local,v 1.39 2006/07/28 20:19:46 sturm Exp $

# Site-specific startup actions, daemons, and other things which
# can be done AFTER your system goes into securemode.  For actions
# which should be done BEFORE your system has gone into securemode
# please see /etc/rc.securelevel.

echo -n 'starting local daemons:'

# Add your local startup actions here.

# BIND Daemon
if [ -x /usr/sbin/named ]; then
        echo -n ' named'
        /usr/sbin/named
fi

# DHCP Daemon
if [ -x /usr/local/sbin/dhcpd ]; then
        echo -n ' dhcpd'
        /usr/local/sbin/dhcpd
fi

# NetBIOS Name Server Daemon
if [ -x /usr/local/libexec/nmbd ]; then
        echo -n ' nmbd'
        /usr/local/libexec/nmbd
fi

# Samba Server Daemon
if [ -x /usr/local/libexec/smbd ]; then
        echo -n ' smbd'
        /usr/local/libexec/smbd
fi

# CUPS Daemon
if [ -x /usr/local/sbin/cupsd ]; then
     /usr/local/sbin/cupsd
     echo -n ' cupsd'
fi

       echo '.'
```

E – Conteúdo final do arquivo smb.conf para o servidor Proxy:

/etc/samba/smb.conf:

```
[global]
workgroup = EXEMPLO
netbios name = proxy
server string = Servidor de Autenticação Proxy
security = domain
bind interfaces only = Yes
smb ports = 445 139
hosts allow = 10. 127.
encrypt passwords = Yes
name resolve order = host wins
idmap uid = 2000-4000
idmap gid = 2000-4000


## Logging
log level = 2
syslog = 0
log file = /var/log/samba/smbd.%m
max log size = 100

## Winbind
password server = pdc.exemplo.br
winbind separator = /
winbind use default domain = yes
```

F – Conteúdo final do arquivo rc.local para o servidor Proxy:

/etc/rc.local:

```
#       $OpenBSD: rc.local,v 1.39 2006/07/28 20:19:46 sturm Exp $
# Site-specific startup actions, daemons, and other things which
# can be done AFTER your system goes into securemode.  For actions
# which should be done BEFORE your system has gone into securemode
# please see /etc/rc.securelevel.

echo -n 'starting local daemons:'

# Add your local startup actions here.

 BIND Daemon
if [ -x /usr/sbin/named ]; then
        echo -n ' named'
        /usr/sbin/named
fi

# NetBIOS Name Server Daemon
if [ -x /usr/local/libexec/nmbd ]; then
        echo -n ' nmbd'
        /usr/local/libexec/nmbd
```

```
fi

# Samba Server Daemon
if [ -x /usr/local/libexec/smbd ]; then
        echo -n ' smbd'
        /usr/local/libexec/smbd
fi

# Winbind Daemon
if [ -x /usr/local/libexec/winbindd ]; then
        echo -n ' winbindd'
        /usr/local/libexec/winbindd
fi


# Squid Daemon
if [ -x /usr/local/sbin/squid ]; then
       echo -n ' squid'
        /usr/local/sbin/squid
fi
      echo '.'
```

G – Conteúdo final do arquivo squid.conf para o servidor Proxy:

/etc/squid/squid.conf:

```
http_port 3128
hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
cache deny QUERY
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache
access_log /var/squid/logs/access.log squid

# Declaração SquidGuard
url_rewrite_program /usr/local/bin/squidGuard -c \
/etc/squidguard/squidguard.conf
 url_rewrite_children 10

# Acesso via NTLM transparente
auth_param ntlm program /usr/local/bin/ntlm_auth \
 --helper-protocol=squid-2.5-ntlmssp
auth_param ntlm children 10

refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern .               0       20%     4320

acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
acl SSL_ports port 443
acl Safe_ports port 80          # http
```

```
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl CONNECT method CONNECT

# Cria uma ACL para permitir somente usuários com autenticação
acl autenticacao proxy_auth REQUIRED

#Recommended minimum configuration:
#
# Only allow cachemgr access from localhost
http_access allow manager localhost
http_access deny manager
# Deny requests to unknown ports
http_access deny !Safe_ports
# Deny CONNECT to other than SSL ports
http_access deny CONNECT !SSL_ports

# Libera acesso apenas aos usuários que realizaram autenticação
http_access allow autenticacao

# Nega o resto dos acessos
http_access deny all

http_reply_access allow all
icp_access allow all
coredump_dir /var/squid/cache
```

H – Conteúdo final do arquivo squidguard.conf para o servidor Proxy:

/etc/squidguard/squidguard.conf:

```
logdir /var/squidguard/log
dbhome /var/squidguard/db/BL

### Definição dos Grupos de Origem
src secretaria {
ip 10.0.0.0/8 127.0.0.1
userlist /etc/squidguard/listas/secretaria
log secretaria.log
}

src tesouraria {
ip 10.0.0.0/8 127.0.0.1
userlist /etc/squidguard/listas/tesouraria
log tesouraria.log
```

```
}

### Definicao dos Grupos de Destino (listas de endereços)
## Listas próprias:
dest lista1 {
domainlist /etc/squidguard/sites_liberados
}

dest lista2 {
domainlist /etc/squidguard/sites_bloqueados
}

## Listas disponíveis em http://www.shallalist.de:
dest adv {
domainlist adv/domains
urllist adv/urls
}

acl {

 secretaria {
 pass lista1 !lista2 !adv all
 redirect http://proxy.exemplo.br/bloqueio.htm
 }

 tesouraria {
 pass lista1 none
 redirect http://proxy.exemplo.br/bloqueio.htm
 }

 default {
 pass none
 redirect http://proxy.exemplo.br/bloqueio.htm
 log default.log
 }
      }
```

I – Arquivo para atualização do banco de dados do SquidGuard:

/etc/squidguard/squidguard-listas.conf:

```
# Este arquivo é utilizado apenas para atualizar o Banco de Dados
# de sites do SquidGuard.
# Listas disponíveis em http://www.shallalist.de

logdir /var/squidguard/log
dbhome /var/squidguard/db/BL

dest adv {
domainlist adv/domains
urllist adv/urls
}
```

J – *Script* para criação de listas de usuários para o SquidGuard:

/home/suporte/scripts/criar_listas.sh:

```
#!/bin/ksh
## Script para criar uma lista com usuários de grupos do sistema.

# Cria um diretório temporário para as listas.

mkdir -p /tmp/listas
cd /tmp/listas

# Cria listas para cada grupo, realizando um parse em /etc/group.

getent group "secretaria" | cut -d: -f4 | tr "," "\n" > secretaria
getent group "tesouraria" | cut -d: -f4 | tr "," "\n" > tesouraria

# Ajusta as permissões das listas, liberando permissão de leitura
# para o usuário _squid, utilizado pelo SquidGuard.

chmod * 640
chgrp _squid *

# Envia as listas para o servidor proxy como o usuário suporte.
# É necessário que a chave pública do usuário esteja no servidor
# proxy (em /home/suporte/.ssh/authorized_keys).

if [ $(whoami) = "root" ];
 then # Efetua a cópia fingindo ser o usuário suporte.
 su suporte -c 'scp /tmp/listas/* 10.1.1.5:/etc/squidguard/listas/'
fi

if [ $(whoami) = "suporte" ];
 then # Efetua a cópia como o usuário em questão (suporte).
 scp /tmp/listas/* 10.1.1.5:/etc/squidguard/listas/
fi
```

K – *Script* para atualização automática de listas de *sites* para o SquidGuard:

/home/suporte/scripts/atualizar- listas.sh:

```
#!/bin/sh
## Script para atualização de listas do SquidGuard

# Parar Squid
/usr/local/sbin/squid -k shutdown

# Mudar diretório padrão
cd /var/squidguard/db

# Remove .tar.gz de alguma atualização anterior, caso exista
if [ -x /var/squidguard/db/shallalist.tar.gz ]; then
```

```
 rm shallalist.tar.gz
fi


# Baixar as listas

if wget -T 60 http://www.shallalist.de/Downloads/shallalist.tar.gz
-a /var/squidguard/log/blacklists.log
 then
  echo "Arquivo baixado com sucesso!"
 else
  echo "Erro ao baixar listas de arquivos. Finalizando processo."
  exit 0
fi


# Remover listas antigas
rm -R BL/

# Descompactar lista nova
tar -xzf shallalist.tar.gz

# Criar Banco de Dados
/usr/local/bin/squidGuard -C all -c /etc/squidguard/squidguard-
listas.conf

# Ajustar permissões de arquivos
chown -R _squid BL/

# Iniciar o Squid
/usr/local/sbin/squid
```

L – Arquivo de configuração para o servidor DNS no PDC:

/var/named/etc/named.conf:

```
acl clients {
        localnets;
};

options {
        version "";
        listen-on { any; };
        allow-recursion { clients; };
};

logging {
        category lame-servers { null; };
};

zone "localhost" {
        type master;
        file "standard/localhost";
        allow-transfer { localhost; };
```

```
};

zone "127.in-addr.arpa" {
        type master;
        file "standard/loopback";
        allow-transfer { localhost; };
};

zone "com" {
        type delegation-only;
};

zone "net" {
        type delegation-only;
};

controls {
        inet      127.0.0.1 port 953
        allow     { 127.0.0.1; };
};

// Zonas mestre

zone "exemplo.br" {
        type master;
        file "master/db.exemplo.br";
        allow-update { 10.0.0.0/8; }; //Atualizações Dinâmicas
};


// Resolução de Nomes Inversa

zone "1.1.10.in-addr.arpa" {
        type master;
        file "master/db.10.1.1";
        allow-update { 10.0.0.0/8; };
};

// RFC 1912

zone "255.in-addr.arpa" {
        type master;
        file "master/db.255";
};

zone "0.in-addr.arpa" {
        type master;
        file "master/db.0";
};
```

M – Arquivos de configuração de zonas para o servidor DNS do PDC:

/var/named/master/db.0:

```
$TTL  3h

@ IN  SOA pdc.exemplo.br.  suporte.exemplo.br.  (
          2007080901  ; serial
          3h          ; refresh  after  3 hours
          1h          ; retry  after  1 hour
          1w          ; expire  after  1 week
          1h  )       ; negative  caching  TTL  of  1 hour


; Name  Servers

          IN   NS   pdc.exemplo.br.
```

/var/named/master/db.10.1.1:

```
$ORIGIN  .
$TTL  10800       ; 3 hours
1.1.10.in- addr.arpa       IN  SOA   pdc.exemplo.br.  suporte.exemplo.br.
(
                                     2007080944  ; serial
                                     10800       ; refresh  (3 hours)
                                     3600        ; retry  (1 hour)
                                     604800      ; expire  (1 week)
                                     3600        ; minimum  (1 hour)
                                     )
                           NS        pdc.exemplo.br.
$ORIGIN  1.1.10.in- addr.arpa.
1                          PTR       ca.exemplo.br.
5                          PTR       proxy.exemplo.br.
```

/var/named/master/db.255:

```
$TTL  3h

@ IN  SOA pdc.exemplo.br.  suporte.exemplo.br.  (
          2007080901  ; serial
          3h          ; refresh  after  3 hours
          1h          ; retry  after  1 hour
          1w          ; expire  after  1 week
          1h  )       ; negative  caching  TTL  of  1 hour

; Name  Servers

          IN  NS    suporte.exemplo.br.
```

/var/named/master/db.exemplo.br:

```
$ORIGIN  .
$TTL  10800       ; 3 hours
```

```
exemplo.br                IN  SOA   pdc.exemplo.br.  suporte.exemplo.br.
(
                                    2007080960  ; serial
                                    10800       ; refresh  (3 hours)
                                    3600        ; retry  (1 hour)
                                    604800      ; expire  (1 week)
                                    3600        ; minimum  (1 hour)
                                    )
                          NS        pdc.exemplo.br.
$ORIGIN  exemplo.br.
ca                        A         10.1.1.1
pdc                       A         10.1.1.1
proxy                     A         10.1.1.5
```

N – Arquivo de configuração para o servidor DNS no Proxy:

/var/named/etc/named.conf:

```
acl clients {
        localnets;
};

options {
        version "";       // remove this to allow version queries
        listen-on     { any; };
        allow-recursion  { clients;  };
};

logging {
        category lame-servers  { null;  };
};

// Standard zones
//
zone "." {
        type hint;
        file "standard/root.hint";
};
zone "localhost"  {
        type master;
        file "standard/localhost";
        allow-transfer  { localhost;  };
};

zone "127.in-addr.arpa"  {
        type master;
        file "standard/loopback";
        allow-transfer  { localhost;  };
};

zone "com" {
        type delegation-only;
};
```

```
zone "net" {
        type delegation-only;
};
```

O – Arquivo de configuração para o servidor DHCP no PDC:

/etc/dhcpd.conf:

```
option domain-name "exemplo.br";
option domain-name-servers 10.1.1.1;

authoritative;

ddns-updates on;
ddns-update-style interim;
ddns-domainname "exemplo.br";
allow client-updates;

subnet 10.0.0.0 netmask 255.0.0.0 {
 range 10.1.1.50 10.1.1.150;
 option routers 10.1.1.10;
 option domain-name-servers 10.1.1.1, 10.1.1.5;
}

zone exemplo.br {
primary 10.1.1.1;
}

zone 1.1.10.in-addr.arpa {
primary 10.1.1.1;
```