Emerson Luis Galeli de Oliveira

Concentrador de VPN com Openswan para conexões em topologia "Road Warrior"

Monografia de Pós-Graduação "*Lato Sensu*" apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em "Administração em Redes Linux"

Orientador
Prof. Joaquim Uchôa

Lavras
Minas Gerais - Brasil
2010

Emerson Luis Galeli de Oliveira

Concentrador de VPN com Openswan para conexões em topologia ”Road Warrior”

Monografia de Pós-Graduação “*Lato Sensu*” apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em “Administração em Redes Linux”

Aprovada em *24 de Abril de 2010*

Prof. Sandro Pereira Melo

Prof. Denilson V. Martins

Prof. Joaquim Uchôa
(Orientador)

Lavras
Minas Gerais - Brasil
2010

Dedico este trabalho a minha família, mas principalmente a minha esposa por toda a ajuda e compreensão durante o processo de elaboração.

Agradecimentos

A minha esposa e meu filho pela compreensão.

Ao professor Joaquim por sua orientação. Aos professores que contribuíram com esta conquista. A Deus por me dar saúde e determinação.

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Este tabalho apresenta o uso do IPSec com Openswan para criação de conexões seguras utilizando certificados X.509 em equipamentos com Linux, onde os clientes não precisão de endereços IPs permanentes ou DNS dinâmico para identificação. E para ajudar no entendimento do conteúdo foi anexado alguns conceitos sobre VPN, algoritmos de criptografia, protocolos e aplicativos usados para criação de túneis VPN.

Palavras-Chave: VPN; Openswan; IPSec; Road Warrior.

Capítulo 1

Introdução

Fornecer recursos de informática para seus funcionários e parceiros, já não é a única tarefa que as empresas tem para agilizar os processos de trabalho e atender mais e melhor seus clientes ou parceiros. Com o crescimento da internet, muitas alternativas foram criadas para fornecer recursos fora dos portões das companhias, antes restritos às redes locais.

Muitas optaram em converter aplicações para o formato Web ou liberaram a troca de arquivos por meio de FTP, sem a devida preocupação com segurança. Mas recursos como Sistemas ERP, Portais Corporativos, Sistema de *Business Intelligence* e Sistemas de digitação de pedidos, possuem dados sigilosos demais para serem abertos à Internet sem nenhum tipo de proteção extra.

Além dos usuários autorizados, pessoas não autorizadas podem ter interesse nos dados das companhias, como os próprios concorrentes. E muitos buscaram no mercado inúmeras soluções para melhorar a segurança na comunicação corporativa, como uso de VPN, SSL/TLS em paginas e E-mails, além de técnicas de criptografia para os dados.

Este trabalho foi motivado pelo desafio de atender usuários que tinham conexões VPN com a empresa, inicialmente com IPSec, e passaram a ter outros computadores compartilhando da mesma conexão internet para atividades extras, onde muitos optaram em colocar um roteador com NAT em seus escritórios. Mas com a exigência do IP dedicado e válido, a solução tradicional de VPN com IPSec passou a ser um entrave na vida dos usuários, porém com o uso de certificados X.509 junto ao IPSec tornou possível as conexões sem endereços IPs fixos ou IPs reservados ao estabelecerem túneis com segurança.

O objetivo deste trabalho é apresentar como realizar conexões seguras com IPSec através de certificados X.509, retirando a obrigatoriedade de faixas de endereços IP aos clientes, uso de DNS dinâmico ou adição de soluções mistas para atender usuários distintos que utilizem NAT em seus roteadores.

Algumas soluções de mercado usadas para criação de VPN entre redes corporativas e seus usuários remotos serão vistas neste trabalho, com os principais protocolos, algoritmos de criptografia, certificados e aplicações. Em destaque estão o IPSec, Openswan e X.509 que completam a solução final apresentada.

O conteúdo foi organizado como descrito. O Capítulo 2 apresenta as formas mais usadas de tunelamento e as principais características dos protocolos e chaves criptográficas. O Capítulo 3 traz os passos para uma implementação bem sucedida. O Capítulo 4 apresenta alguns dos resultados obtidos com a implantação da solução proposta. O Capítulo 5 apresenta as considerações finais para o fechamento do trabalho. No Apêndice A estão descritos os passos para criação dos certificados e no Apêndice B é apresentado uma breve descrição de configuração em ambiente não Linux.

Capítulo 2

VPN

Este capítulo apresenta alguns tipos de VPN descrevendo detalhes sobre protocolos, algoritmos, aplicações e certificados utilizados a fim de facilitar o entendimento do conteúdo.

2.1 Definições

As redes de computadores nasceram da necessidade de compartilhamento de recursos, sendo usadas por inúmeras aplicações com finalidades distintas e, alguns dados transmitidos entre empresas fazem parte de processos confidenciais e não devem ser do conhecimento publico. Com isso, várias empresas buscaram no mercado formas de melhorar a segurança e reduzir os custos de conexões de longas distâncias.

Das técnicas criadas para melhorar a segurança durante a troca de informação, diversas chegaram ao uso da criptografia com o intuito de proteger e garantir a integridade do dados. Mas como citado em (UCHÔA, 2005), é importante que os seguintes elementos da "segurança computacional" sejam observados:

1: Confiança: Trata da disponibilidade do sistema, da acessibilidade aos dados e a garantia de armazenamento com cópias de segurança, evitando a perda ou um destino impróprio.

2: Integridade: A integridade do que está sendo compartilhado nas redes está relacionado a não adulteração do conteúdo transferido da origem ao destino.

3: Confidencialidade: Somente o remetente e o destinatário devem entender o conteúdo transmitido, deve haver algo que certifique a privacidade dos dados para evitar a espionagem.

As VPNs (*Virtual Private Network*s), definido em (TANENBAUM, 2003) como sendo redes sobrepostas às redes públicas, passaram a ter destaque nas conexões entre redes por apresentar baixo custo e nível de segurança muito bom. O baixo custo das VPNs deve-se ao uso das redes públicas (Internet) para estabelecer túneis virtuais entre redes físicas.

Alguns autores definem de maneira genérica o termo VPN, justamente porque além de existirem várias técnicas no mercado para criação de redes privadas, muitas diferem na arquitetura e nos protocolos utilizados. Pode ser visto na Tabela 2.1 algumas soluções encontradas:

Tabela 2.1: Soluções de conexões privadas frente ao modelo OSI

Modelo OSI	Camada	Soluções
Aplicação	7	SSH, Socks
Apresentação	6	
Sessão	5	
Transporte	4	SSL/TLS
Rede	3	IPSec, MPLS, GRE, OpenVPN
Enlace de Dados	2	Frame Relay, ATM, PPTP, L2F, L2TP
Física	1	LPCD, Linha Privativa

Dos exemplos apresentados na Tabela 2.1, alguns merecem um maior destaque e serão comentados a seguir:

1: MPLS: *Multi-Protocol Label Switching*, padronizado pelo IETF, este protocolo tem o repasse rápido de dados com a possibilidade de reserva de banda, pois possui um excelente QoS (*Quality of Service*);

2: GRE: *Generic Routing Encapsulation*, tem como caracteristica o tunelamento de qualquer protocolo dentro de outro protocolo mas não insere criptografia na criação dos túneis;

3: *Frame Relay* é um protocolo criado para transporte otimizado de pacotes entre redes locais com segurança, possui controle de largura de banda, sendo indicado na transmissão de voz e dados. No *Frame Relay*, a criação de circuitos virtuais entre unidades de uma empresa permite a interligação das

LANs e Centrais telefonicas de forma efetiva. Em cada um dos pontos de acesso são implantadas os PVCs (*Permanent Virtual Circuits*), onde são definidos os parâmetros de CIR (Largura de banda permitida), EIR (Largura de banda excedida), além do número do canal que o PVC irá trabalhar. Não há criptografia se aplicado sem outro protocolo que forneça esta funcionalidade, pois o *Frame Relay* cria tuneis virtuais e os gerencia sem opções de cifragem de dados;

4: ATM *Asynchronous Tranfer Mode*, é uma arquitetura baseada em comutação de pacotes por circuitos virtuais (VC), popular entre as empresas de telecomunicações para atender redes geograficamente distribuídas. Assim como o Frame Relay, o ATM cria circuitos virtuais onde a operadora de telecomunicações pode configura-lo para ser um circuito privado e interligar as redes de uma companhia, porém sem o uso de criptografia, mas que pode ser adicionada com o uso de outros protocolos.

2.2 Criptografia

Informações sobre nosso perfil médico ou financeiro, previsões de vendas, estratégias de mercado para um novo produto, projetos de engenharia, todos contém dados confidenciais e devem ser manipulados com cuidado pois pessoas e empresas desonestas podem benficiar-se dessas informações. Por exemplo uma seguradora adoraria saber detalhes da fixa médica de um futuro segurado ou, sabendo do laçamento de um produto da concorrência pode-se patentiar a idéia antes mesmo do criador.

Um caso real que aconteceu em fevereiro de 2008, foi divulgado nos jornais, revistas e internet sobre o desaparecimento de Notebooks da Petrobras, contendo informações sigilosas de pesquisas realizadas durante anos e segundo nota publicada pela Petrobrás "houve vazamento de informações",(G1, 2008). Apesar de serem equipamentos que estavam sob o cuidado de uma empresa terceira, é provável que a Petrobras não deu a devida importância para o conteúdo que seus terceiros transportavam, onde medidas como o uso de criptografia ou trava de acesso ao conteúdo poderia ter evitado tal incidente.

De forma simples a criptografia é o processo de converter dados legítimos em algo sem sentido, com a capacidade de recupera-los a partir desse conteúdo sem sentido - (BURNETT; PAINE, 2002). Para que os dados trafeguem ou sejam armazenados de forma segura, com uma proteção além da que é oferecida pelo

sistema operacional, a criptografia é fundamental para isso, (KUROSE; ROSS, 2007), porque traz aspectos como Confidencialidade, Autenticação, Integridade e Autoria dos dados. As técnicas de criptografia utilizam-se de chaves e algoritmos para codificação e decodificação dos dados, podendo variar o tipo e tamanho de acordo com a aplicação e segurança desejada.

Alguns algoritmos de Criptografia fora criados ao longo dos anos e como descrito por (PETERSON; DAVIE, 2004), existem três tipos: algoritmos simétricos, algoritmos de chave pública e algoritmo de função *hash*.

2.2.1 Algoritmo Simétrico ou de Chave secreta

São algorítimos simétricos porque os participantes da comunicação utilizam a mesma chave para encriptação e decriptação da mensagem compartilhada, podendo ser uma palavra, uma frase ou uma sequência aleatória de números. O tamanho da chave é medido em bits e, quanto maior a chave, mais seguro poderá ser o documento encriptado. Porém esse tipo de algoritmo exige que os usuários mantenham as chaves em segredo, ou seja, a forma mais segura de uso deste algoritmo é quando o usuário que encripta também decripta o conteúdo, como é o caso de proteger arquivos ou pastas presentes em um disco rígido.

Um exemplo de algoritmo muito utilizado é o DES (Data Encryption Standard), que foi o algoritmo padrão do governo americano durante anos e, passou por modificações para acréscimo de segurança chegando a versão 3 (Triple DES ou 3DES), que aplica três vezes o DES sobre os dados. Outro exemplo é o algoritmo IDEA (*International Encryption Algorithm*), com a mesma linha de trabalho do DES porém não é de domínio publico e seu maior uso é no programa PGP. Outro exemplo é o algoritmo Blowfish. Na Figura 2.1 está um esboço de um processo de encriptação de dados onde os dois participantes que compartilham a mesma chave para cifrar ou extrair os dados cifrados de uma mensagem.

2.2.2 Algoritmo de Chave pública

Ao contrário do que é feito pelos algoritmos simétricos, onde há apenas uma chave para todos os participantes, os algoritmos de chave pública possuem um par de chaves. Uma chave pública, usada para cifrar as mensagens e conhecida por todos envolvidos na comunicação e, uma chave privada, usada para decodificar as mensagens enviadas e cada participante possui sua chave exclusiva.

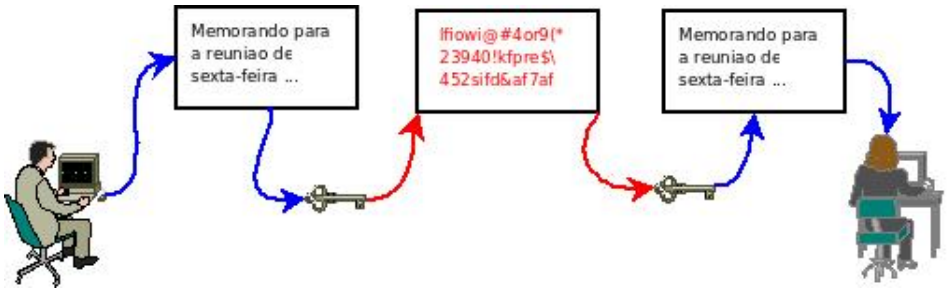

Figura 2.1: Cifragem e Decifragem com a mesma chave

Como apontado em (BURNETT; PAINE, 2002), no início da década de 1970, Whitfield Diffie e seu professor Martin Hellman criaram um esquema onde duas pessoas poderiam comunicar-se criando uma chave secreta compartilhada. Eles publicaram o resultado de seus estudos em 1976, descrevendo a idéia de criptografia de chave pública onde uma chave encripta e a outra decripta. Nesse trabalho, entretanto, informaram que não tinham o algoritmo pronto até aquele momento. Foi quando Ron Rivest, Adi Shamir e Len Adleman resolveram trabalhar na busca por uma solução e, em 1977 os três criaram o que foi publicado em 1978 como sendo o algoritmo de chave pública RSA.

O algoritmo RSA tem como característica a possibilidade de criar chaves de tamanhos variados e principalmente permitir a distribuição de chaves, o que resolve muito problema na troca de informações e garante chaves mais complexas e seguras, características não encontradas em algoritmos de chave simétrica como o DES (56bits) ou o 3DES (168bits). Porém quanto maior forem as chaves mais processamento será exigido dos *hosts* que trocam os dados utilizando este tipo de chave.

Alguns algoritmos de chave pública presentes no mercado são: ElGamal, ECDH e DSS (DIGITAL SIGNATURE STANDARD).

2.2.3 Certificados de Chave pública

Um certificado de chave pública (*public key certificate* – PKC), corresponde a um conjunto de dados à prova de falsificação que garante a associação da chave a um usuário final. Contudo a associação só é fornecida por um terceiro confiável chamado de Autoridade Certificadora ou CA (Certification Authorities), onde esta emite os certificados que contem o nome do usuário, a chave pública e outras informações de identificação.

A necessidade de padronizar os certificados digitais e garantir seu processo de distribuição de chaves, exigiu a criação dos chamados PKI (*Public Key Infrastructures*), que nada mais é do que um processo colaborativo entre as entidades CA, RA (*Registration Authority*), um repositório de certificados, um servidor de recuperação de chaves e o usuário final.

Descrito em (BURNETT; PAINE, 2002), o padrão de certificado X.509 proposto pela ITU-T (*International Telecommunications Union*) e publicado inicialmente em 1988, passou por revisões até chegar na sua versão final em 1999, quando foi oficialmente publicado pelo IETF (*Internet Engineering Task Force*). O X.509 tornou-se um padrão de mercado possuindo a estrutura apresentada na Figura 2.2.

Versão
Número Serial do Certificado
Identificador do algoritmo de assinatura
Nome do Emissor
Validade - Não antes de/Não depois de
Nome do sujeito portador do certificado
Chave pública do sujeito
Identificador único do emissor
Identificador único do sujeito
Extensões
Assinatura

Figura 2.2: Estrutura do certificado X.509. Fonte (BURNETT; PAINE, 2002)

2.2.4 Função Hash ou Message Digest

Também conhecidos como *Resumo de mensagem*, é um algoritmo que recebe qualquer comprimento de entrada e mescla o conteúdo para produzir uma saída pseudo-aleatória de largura fixa. Como exemplo, uma frase com 50 bytes teria como resultado 20 bytes ou, uma mensagem de 15 bytes ao ser convertido por uma função *Hash* teria como resultado algo com 20 bytes.

Outra característica é que os resultados dos resumos parecem aleatórios, com bytes aparentemente sem sentido, mas se utilizar o mesmo algoritmo duas vezes para uma mesma mensagem notará que o resultado será o mesmo, mesmo em equipamentos distintos. E mensagens similares podem ter resultados totalmente

distinto, enquanto resultados onde há uma variação mínima de 1 ou 2 bytes podem ser de mensagens sem nenhuma proximidade uma da outra.

O uso destes algoritmos permite a proteção de senhas ou também garantir a integridade de arquivos, como ocorre com as senhas dos usuários no Linux que são mantidas no arquivo */etc/shadow*. Outro exemplo são em sites onde arquivos de programas são mantidos com seus respectivos resumos cifrados, a fim de garantir aos usuários o *download* de arquivos sem adulteração, funcionando como uma assinatura de autenticidade do arquivo. Fazendo uma analogia simples, os algoritmos *hash* funcionam como o digito verificador de documentos. Os algoritmos mais conhecidos e confiáveis são o MD5 (*Message Digest* 5) e SHA-1 (*Secure Hash Algorithm*).

2.3 Protocolos de Segurança e Tunelamento

Nesta seção serão abordados alguns dos protocolos de tunelamento sobre a plataforma IP, com o intuito de apresentar soluções com pouca ou muita abrangência no mercado, mas que estão disponíveis em diversos sistemas operacionais e roteadores. O destaque maior está para o IPSec, alvo principal deste trabalho e que merece atenção pois trata-se de um protocolo completo para VPN.

2.3.1 SSH

O *Security Shell* oferece um serviço de login remoto seguro e foi criado com o objetivo de substituir os serviços de console remota como Telnet e rlogin, porque oferece autenticação forte e criptografia dos dados (PETERSON; DAVIE, 2004). Também permite a transferência de arquivos ou execução de comandos remotamente como ftp e rsh. Os comandos são suportados por computadores com Unix/Linux, mas não possuem segurança forte para o conteúdo trafegado o que não garante a integridade nem a confidencialidade dos dados.

Na versão 2 do SSH ele possui 3 camadas:

1: SSH-TRANS: oferece um canal criptografado entre o cliente e o servidor, sendo executado sobre uma conexão TCP e sempre que o usuário efetuar o login em um servidor, a primeira etapa é estabelecer um canal onde o servidor solicita ao cliente que se autentique através de uma chave RSA. Após a autenticação, cliente e servidor estabelecem uma chave de sessão

para criptografia de quaisquer dados enviados pelo canal. Nesta camada também é feito a negociação do algoritmo de criptografia que irão utilizar, podendo usar o 3DES. Além disso, é feita a verificação de integridade das mensagens de dados passadas pelo canal;

2: SSH-AUTH: permite ao servidor garantir a identificação do cliente e provar que o cliente é realmente quem ele diz ser. Esta garantia é feita através da chave de criptografia criada na camada de transporte, isso dá ao cliente condições para utilizar outros serviços do servidor;

3: SSH-CONN: Em conjunto com a camada de SSH-AUTH, esta camada torna disponível a execução de aplicações por meio de um túnel seguro, através de um recurso definido como "encaminhamento de porta". Por exemplo, é possível utilizar aplicações presentes no servidor como X Windows, IMAP ou FTP através do túnel SSH criado.

2.3.2 SSL

O *Secure Sockets Layer* (SSL), como definido em (BURNETT; PAINE, 2002), é um protocolo de Internet para a criptografia e autenticação baseado em sessão, que fornece um canal seguro entre duas partes (Cliente e Servidor). O servidor deve ser autenticado enquanto o cliente é dada a autenticação como opção e, uma vez definida a chave secreta entre as partes, o SSL oferece a privacidade necessária para a comunicação entre os envolvidos na comunicação.

O SSL utiliza a camada de transporte e independe do protocolo de aplicativo, com isso protocolos como HTTP, FTP, SMTP, LDAP, entre outros, podem utiliza-lo de forma transparente por estarem na camada superior. como mostrado na Figura 2.3

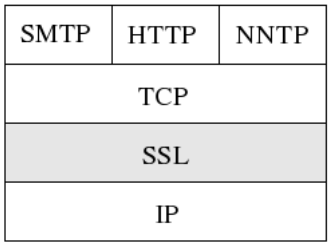

Figura 2.3: SSL na pilha TCP/IP. Fonte (BURNETT; PAINE, 2002)

A Netscape desenvolveu o SSL em 1994, após várias atualizações chegou ‘a versão SSLv3 e, em 1996 foi passada para o IETF as especificações e o grupo de trabalho rebatizou-o como TLS (*Transport Layer Security*). O IETF inseriu pequenas alterações e passou a distribui-lo como TLSv1 desde 1999. Alguns recursos importantes do SSLv3/TLSv1 são: Permitir que Cliente e Servidor solicitem um novo *handshake* a qualquer momento e chaves ou cifras podem ser renegociadas; Compactação de dados; Uso de chaves Diffie-Hellman e chaves não RSA; Certificados em cadeia.

O SSL/TLS é formado por duas partes: seu estado e as transições de estado. O estado descreve o sistema em um ponto específico do tempo. As transições de estado são os processos de alteração de um estado para outro. Todos estados e transições de estado referente a um objeto são chamados de máquina de estado, onde é possível ter duas máquinas de estado: uma para o cliente e outra para o servidor. As interações entre as máquinas de estado são chamadas de *handshake*.

O SSL/TLS é composto por: *Record protocol, Handshake protocol, Alert protocol, ChangeCipherSpec protocol*.

1: *Record protocol* ou Protocolo de Camada de Registro: interage com os protocolos das camadas superiores para realizar a criptografia, decriptografia e autenticação dos dados;

2: *Handshake protocol*: estabelece os parâmetros do estado de sessão atual, tais como versão do SSL, valor aleatório, ID de sessão, cifras aceitáveis e métodos de compactação;

3: *Alert protocol*: o protocolo de alerta transporta as mensagens de alerta e a severidade delas para as partes de uma sessão de SSL. As mensagens são compactadas e cifradas pela especificação do estado atual da conexão;

4: *ChangeCipherSpec protocol*: existe apenas para sinalizar uma transição em estratégias de criptografia. Consiste em uma única mensagem, cifrada e compactada pela camada de registro. Antes de terminar o protocolo de *handshake* envia a mensagem para notificar que os registros subseqüentes serão protegidos com a especifica criptografia e chave, ocorrendo em ambos os comunicadores.

2.3.3 L2F

Foi um dos primeiros protocolos criado pela CISCO para VPN, este protocolo sempre assume que a rede privada encontra-se atrás de um *gateway*. Como citado em (LEWIS, 2004), seu tunelamento independente do protocolo IP, o que lhe permite trabalhar com redes ATM e Frame Relay.

O protocolo L2F estabelece uma conexão do usuário remoto com um Servidor de Acesso ou NAS (*Network Access Server*) à rede, utilizando na autenticação um dos protocolos: PPP, RADIUS, TACACS ou TACACS+. Depois o NAS requisita uma conexão com o *gateway* da rede privada, liberando o tráfego de pacotes do usuário remoto através do túnel criado entre o NAS e o *gateway*.

Descrito em (STEWART JAMES MICHAEL; CHAPPLE, 2008), este protocolo tem como deficiência não suportar a criptografia e não possuir um padrão de tunelamento, tendo grandes desvantagens se comparado a outros protocolos de tunelamento para VPN, pois deixa os dados vulneráveis a todo tipo de ataque.

2.3.4 PPTP

Este protocolo foi desenvolvido pelo consórcio de empresas composto por US Robotics, 3Com Primary Access, Ascend, Microsoft e ECI Telematics e entregue em 1996, dois anos antes do IPSec e L2TP. O PPTP é um protocolo baseado e considerado uma extensão do PPP pois faz uso de mecanismos de autenticação, compactação e criptografia do PPP.

De acordo com o (TECHNET, 2010b), este protocolo foi designado para fornecer autenticação e criptografia durante a comunicação entre um cliente e um *gateway*, ou entre dois *gateways*, sem a necessidade de uma chave pública, utilizando apenas uma identificação de usuário e uma senha.

Seu objetivo era ser simples, ter suporte a múltiplos protocolos e ter a capacidade de percorrer uma ampla gama de redes IP. O *Point-to-Point Tunneling Protocol* (PPTP) utiliza uma conexão TCP para manutenção do tunel, o *Generic Routing Encapsulation* (GRE) para o encapsulamento dos dados dentro dos frames PPP, (TECHNET, 2010b) podendo ser encriptados utilizando MPPE e EAP-TLS, ou compactados e, para autenticação dos túneis também se utiliza dos protocolos PAP, CHAP, MS-CHAPv1, MS-CHAPv2, EAP. A Figura 2.4 apresenta como é feito o encapsulamento do PPTP.

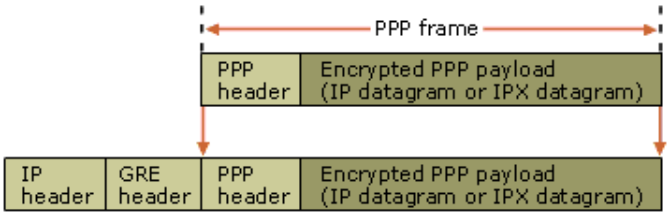

Figura 2.4: Encapsulamento PPTP. Fonte (TECHNET, 2010a)

O PPTP tem como suas principais características permitir negociar a autenticação, a forma de encriptar os dados e os serviços de atribuição de endereços IPs ao clientes. Mais detalhes podem ser vistos em (Network Working Group, 1999) ou em (TECHNET, 2010b).

2.3.5 L2TP

É um protocolo desenvolvido pela IETF (*Internet Engineering Task Force*) que combina as melhores características dos protocolos L2F e o PPTP - (LEWIS, 2004).

Definido na RFC2661 - (TOWNSLEY *et al.*, 2010), o L2TP estende as funcionalidades do PPP permitindo que L2TP e PPP estejam em equipamentos diferentes e interligados pela rede. As conexões feitas pelos clientes remotos chegam até um *L2TP Access Concentrator* (LAC), mas as conexões PPP se conclui em um *L2TP Network Server* (LNS). Com isso as conexões PPP passam por um tunelamento via LAC até chegar ao LNS.

Uma das diferenças entre o L2TP e o PPTP está no protocolo utilizado na camada inferior. Enquanto o PPTP deve ser sempre utilizado acima do IP, o L2TP pode ser utilizado sobre redes IP, X.25, Frame Relay ou ATM.

Este protocolo trabalha em dois modos de tunelamento, **voluntário** e **compulsório**. No modo **voluntário** o túnel é solicitado pelo computador cliente, que pode discar para qualquer provedor de acesso (LAC) e, este pode localizar o melhor servidor para criação do túnel (LNS). No modo **compulsório** o túnel é iniciado pelo servidor de acesso, de forma automática, onde todas as configurações sobre os clientes são mantidas pelo servidor de acesso (LAC). Na Figura 2.5, é apresentada como uma conexão L2TP se realiza.

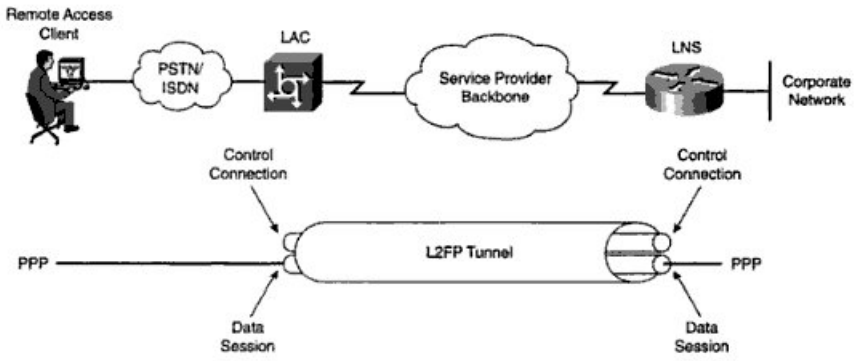

Figura 2.5: Conexão L2TP. Fonte (LEWIS, 2004)

O controle de mensagens é similar ao L2F, utiliza-se o processo chamado *in-band* através do UDP/IP, enquanto no PPTP o processo é chamado de *out-of-band* com uma conexão TCP separada.

O protocolo permite a autenticação dos túneis nas extremidades, durante a configuração, através de protocolos inclusos no PPP como PAP, CHAP ou MS-CHAP (v1 e v2). Os serviços de autenticação como Radius e Tacacs também são suportados, porém não possuem processos para gerenciamento de chaves encriptadas. O L2TP em conjunto apenas com o PPP permite a criação de túneis porém em modo texto simples, não havendo a encriptação do conteúdo, expondo os pacotes de dados e os pacotes de controle a algumas formas de ataque.

Para adicionar mais segurança ao L2TP pode-se configura-lo para trabalhar em conjunto com IPSec, promovendo-lhe um nível de segurança satisfatório, pois permite a implementação de algoritmos como DES ou 3DES nas mensagens L2TP trocadas entre os computadores cliente e servidor. O encapsulamento L2TP e IPSec de um datagrama PPP é ilustrado na Figura 2.6.

2.3.6 SOCKS

Citado em (FOWLER, 1999), uma outra abordagem de tunelamento é através do protocolo SOCKS, trata-se de um protocolo de *proxy* para aplicações baseadas em TCP/IP. O SOCKS foi desenvolvido para permitir que servidores *proxy* fechem conexões seguras com seus clientes, mas a aprovação do IETF só ocorreu em sua versão 5 onde foi registrado na RFC1928, tendo como pessoas chaves do projeto:

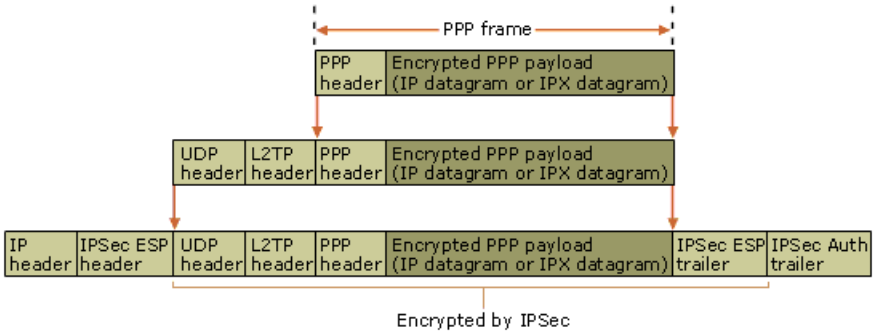

Figura 2.6: Frame PPP com L2TP e IPSec. Fonte (TECHNET, 2010b)

Marcus Leech: Bell-Northern Research; David Koblas: Independent Consultant; Ying-Da Lee: NEC Systems Laboratory; LaMont Jones: Hewlett-Packard Company; Ron Kuris: Unify Corporation; Matt Ganis: International Business Machines.

O SOCKS inclui dois componentes: o *SOCKS server* e o *SOCKS client*, onde no servidor o SOCKS é implementado na camada de aplicação e no cliente entre as camadas de aplicação e transporte, tendo com propósito permitir que os hosts conectados ao servidor tenham acesso às aplicações de internet sem a necessidade de um IP válido.

O funcionamento é simples, quando o cliente requisita um serviço de rede o cliente SOCKS intercepta e envia a solicitação para o servidor SOCKS. Se a requisição for aceita, o server estabelece uma sessão com o client e passa a atender as solicitações de serviço, ou seja, o servidor SOCKS conecta ao servidor de aplicação em nome do cliente SOCKS e repassa os dados da aplicação para o cliente. Para o servidor de aplicação quem fez a requisição foi o servidor SOCKS, similar ao que ocorre com o uso do mascaramento de rede através do NAT.

O SOCKS permite conexões privadas, mas é preciso ter atenção pois algumas soluções utilizam o SOCKSv4 que possui algumas restrições com protocolos, como ICMP, FTP-data e toda a base UDP, impedindo o funcionamento de aplicações como bind udp, traceroute, ping, entre outras. Uma solução de VPN via socks encontrada é o Aventail VPN - (SONICWALL, 2009).

2.3.7 IPSec

Descrito em (BURNETT; PAINE, 2002), esta *suite* de protocolos tem sua estrutura desenvolvida em padrões abertos para assegurar na rede IP uma comunicação privada segura, garantindo confidencialidade, integridade e autenticidade através da criação de túneis que permitem a comunicação de dados por uma rede pública.

O IPSec implementa a criptografia e a autenticação na camada de rede, onde sistemas e aplicativos finais podem aproveitar a vantagem de uma segurança forte sem quaisquer alterações ou configurações adicionais, a Figura 2.7 apresenta a estrutura de um pacote TCP/IP - IPSec onde é possível observar a semelhança com pacotes TCP/IP simples, o que lhes permitem também ser roteados entre redes IP sem nenhuma alteração nos equipamentos de rede, evitando custos extras de implementação e de gerenciamento.

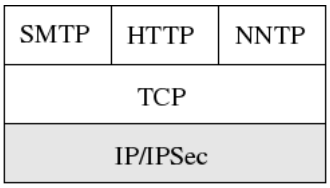

Figura 2.7: Segurança na camada de Rede. Fonte (BURNETT; PAINE, 2002)

O IPSec combina diversas técnicas para proteção dos dados dentro de um pacote IP, definindo quais informações são adicionadas aos pacotes e também como criptografar os dados inseridos neles, e para negociar a associação de segurança o IPSec pode utiliza o IKE (*Internet Key Exchange*), que permite às duas entidades efetuarem as trocas de chaves de forma automática e fácil.

O IKE integra as principais funcionalidades dos protocolos ISAKMP (*Internet Security Association and Key Management Protocol*), Oakley e SKEME, onde o IKEv1 é especificado na RFC4109 e o IKEv2 nas RFC4306. Apesar do IKEv1 e o IKEv2 possuirem cabeçalhos com o mesmo formato e executarem sobre a mesma porta 500 do protocolo UDP, ambos os participantes de uma comunicação segura precisam ter a mesma versão do IKE, pois versões diferentes não trabalham em conjunto. Outra diferença importante é o fato do IKEv2 possuir suporte ao NAT-T, a autenticação extensiva, a aquisição remota do endereço e permitir o uso da porta 4500 do UDP para o IKE SA.

Segue algumas das características que o IKE adotou dos protocolos citados:

- **ISAKMP**: - possui uma estrutura para autenticação e troca de chaves mas não define como realizar isto, sendo este desenvolvido para ser um protocolo de troca de chaves independente e com suporte a diversas trocas de chaves;

- **Oakley**: - descreve uma sequência de troca de chaves conhecidos como modos e, detalha os serviços definidos por cada um (exemplo: troca de chaves, proteção de identidade e autenticação);

- **SKEME**: - descreve uma técnica versátil de troca de chaves que fornece anonimato, rejeição e atualização rápida de chaves. Tanto o SKEME como o Oakley definem um metodo de intercâmbio de chave autenticada, isto inclui a construção das cargas, que informações serão carregadas e qual a ordem que cada uma delas será processada e usada.

O IKE realiza seus serviços em duas fases, detalhadas como:

Primeira: O IKE estabelece um canal de comunicação seguro e autenticado chamado de associação de segurança do IKE, ou ISAKMP SA ou simplesmente IKE SA. Essa troca é baseada no algoritmo Diffie/Hellman e no símbolo cifrado de identificação, mas o processo de autenticação pode ser através de uma chave pré-compartilhada (Pre Shared Key) ou uma chave RSA. Nesta fase o IKE permite apenas 2 modos de troca de informações para configuração das SAs, o modo Principal (*Main Mode*) e o modo Agressivo (*Agressive Mode*);

Segunda: As SAs são negociadas em nome dos serviços como IPSec e seus componentes (AH e ESP), podendo ser trocadas as chaves e outros parâmetros através do canal seguro estabelecido na primeira fase. Na segunda fase o modo Rápido ou (*Quick Mode*) é utilizado e aplicado apenas nesta fase, sendo criado nesta fase o IPsec SA, que define os serviços de segurança a serem utilizados e protegendo o tráfego que por ele transitar. Também são recriadas novas chaves criptográficas periodicamente para melhorar a segurança, sendo comum a renovação mais frequente no IPsec SAs do que no ISAKMP SAs. Múltiplos IPSec SAs podem ser negociados através do canal seguro criado na primeira fase, permitindo que serviços de segurança mais fragmentados e mais flexíveis sejam negociados.

Os serviços de segurança fornecidos pelo IPSec são baseados em dois componentes, o AH (*Athentication Header*) e o ESP (*Encapsulating Security Payload*). Tais serviços de segurança são conhecidos como: controle de acesso, integridade sem conexão, autenticação da origem dos dados, rejeição de pacotes repetidos,

confidencialidade (criptografia dos dados) e confidencialidade limitada do fluxo de tráfego.

O AH fornece os serviços de controle de acesso, integridade sem conexão, autenticação da origem dos dados, rejeição de pacotes repetidos (evita ataques do tipo *replay*). O AH utiliza funções *hash* ao invés de assinaturas digitais, garantindo o serviço de integridade do que é trafegado e evitando problemas de performance, uma vez que o AH não oferece o serviço de proteção aos dados ou confidencialidade. A Figura 2.8 mostra a estrutura do Authentication Header no IPSec.

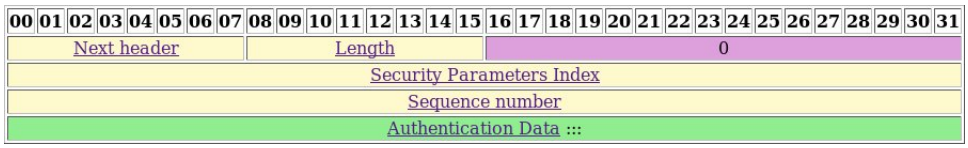

Figura 2.8: Authentication Header

Na Figura 2.8, tem-se:

- **Next Header**: Especifica o próximo protocolo da camada superior a ser encapsulado, como TCP ou UDP;
- **Length**: Indica o comprimento do conteúdo AH;
- **SPI**: O *Security Parameters Index* indica um conjunto de parâmetros usados para identificar uma associação de segurança na conexão vigente;
- **Sequence number**: Fornece um número sequencial crescente para cada pacote enviado com um determinado SPI, isso garante proteção contra ataques do tipo *replay*;
- **Authentication Data**: Contém o *Integrity Check Value* (ICV) ou valor de verificação da integridade do pacote, sendo nele inserido o resuldado da função *hash* que é calculada utilizando campos do cabeçalho IP, do AH e de dados dos protocolos de camadas superiores;

O ESP fornece os serviços de confidencialidade, controle de acesso, confidencialidade limitada do fluxo de tráfego e a rejeição de pacotes do tipo *replay*, sendo que sozinho o ESP consegue atender a maioria dos serviços de segurança oferecidos pelo IPSec. A criptografia dos dados presentes no pacote IP garante a confidencialidade oferece mais um nível de autenticação. Seu formato varia de

acordo com o tipo e modo de criptografia que esta sendo utilizado, mas em todos os casos a chave utilizada para criptografia é selecionada através do campo SPI presente no ESP e AH. Na Figura 2.9 é apresentado o ESP, para uma melhor compreensão.

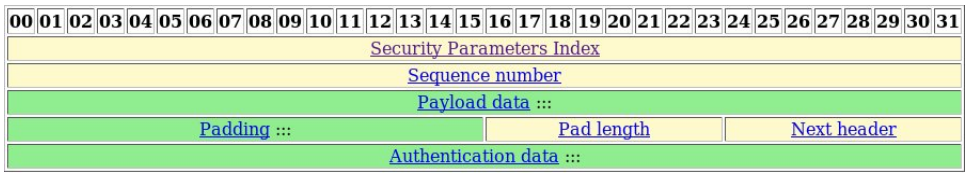

Figura 2.9: Encapsulating Security Payload

Tem-se na Figura 2.9, os seguintes elementos:

- **SPI**: O Security Parameters Index indica um conjunto de parâmetros usados para identificar uma associação de segurança na conexão vigente, assim como no AH;
- **Sequence number**: Fornece um número sequencial crescente para cada pacote enviado com um determinado SPI, permitindo ao destinatário monitorar a ordem do pacote e evitar ataques como o já citado no AH;
- **Payload Data**: Contém o conteúdo dos dados cifrados e transportados pelo pacote IP;
- **Padding**: Fornece espaço para a adição de bytes requeridos por alguns algoritmos de criptografia, isso confunde invasores que podem tentar obter informações sobre a criptografia usada no pacote de dados em trânsito;
- **Next Header**: Identifica o tipo de dado transportado no campo Payload Data;
- **Authentication Data**: Contém o valor do ICV calculado no pacote do ESP menos o campo Authentication Data, sendo este opcional pois é incluído apenas se o serviço de autenticação for definido na associação de segurança;

Os cabeçalhos AH e ESP podem ser combinados de diversas formas, onde a RFC2401 descreve as combinações das associações de segurança que devem ser suportadas pelo protocolo IPSec.

Tanto o AH como o ESP podem trabalhar de duas formas: em modo *transporte* ou em modo *túnel*. Esses modos diferem de acordo com a topologia empregada na VPN.

No modo *transporte*, o AH é usado para hospedar implementações e fornecer proteção para protocolos de camada superior, além dos campos no cabeçalho IP, sendo inserido depois do cabeçalho IP, antes de qualquer protocolo de camada superior e antes de outro cabeçalho IPSec. Já o ESP é utilizado para suportar implementações de *hosts* e fornecer proteção para os protocolos de camada superior, mas não para o próprio cabeçalho IP e assim como no AH, o cabeçalho ESP é inserido depois do cabeçalho IP, antes de qualquer protocolo de camada superior e antes de outro cabeçalho IPSec. Este modo é aplicado em conexões *host-to-host*.

No modo *túnel*, tanto o AH como o ESP operam de forma semelhante com uma distinção, o ESP adiciona ao final de cada pacote alguns dados chamados de *trailers*, mas ambos permitem a proteção ao pacote de IP interno, incluindo o cabeçalho do IP interno. Estando nos cabeçalhos de IP original os endereços de destino finais e no novo cabeçalho de IP externo um endereço distinto. Esse modo pode ser empregado em conexões *gateways-to-gateways* ou *gateways-to-hosts*. Na Figura 2.10 estão dispostos os cabeçalhos com a aplicação do AH, na Figura 2.11 o uso do ESP, para ambos os modos de trabalho.

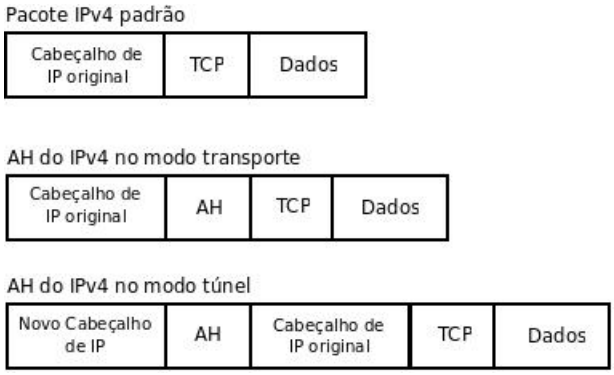

Figura 2.10: IPv4 com AH em modo Túnel e Transporte. Fonte (BURNETT; PAINE, 2002)

2.4 Aplicações para VPN em Linux

2.4.1 Openswan e strongSwan

Como citado em (PIAT, 2009), uma implementação IPsec pode ser dividida em duas partes:

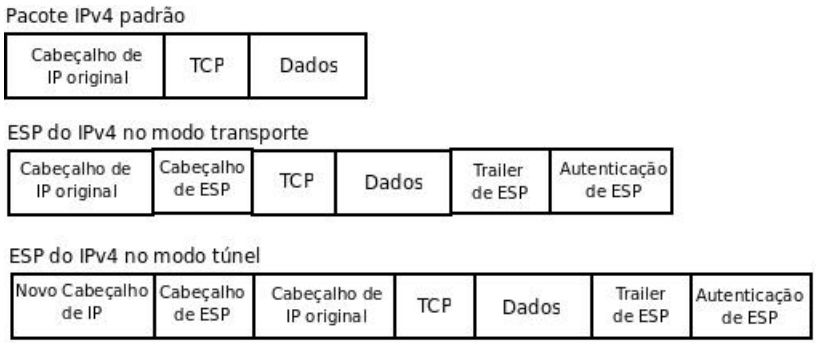

Figura 2.11: IPv4 com ESP em modo Túnel e Transporte. Fonte (BURNETT; PAINE, 2002)

- **kernel**: cuida de tudo, uma vez que a criptografia ou chaves de assinatura são conhecidos;

- **user-level-program**: negocia com antecedência para definição de chaves e retorna a parte do *kernel* através de uma *IPSec-specific kernel API*. O programa em nível de usuário também pode ser configurado para renegociar as chaves periodicamente para chaves que não são usadas há um longo tempo. Uma vez que o *user-level-program* solicita um suporte para o *kernel-level* é preciso ter certeza de que o *kernel* suporta a API que o pacote espera.

Antes do Linux ter suporte nativo ao IPSec, houve os projetos do FreeS/WAN e do KAME. Ambos incluíram um patch ao kernel que se comunicava com um servidor de troca de chaves. Assim era necessário instalar pacotes da aplicação e recompilar um novo *kernel* com o *patch* para IPSec.

O experimental Linux 2.5 aprovou o *kernel* API do KAME que passou a ser implementado no Linux 2.6, e as novas versões do FreeS/WAN (2.01 ou superior) passaram a utilizar a API também. Em seguida, algumas distribuições GNU/Linux, incluiram em seus pacotes um *backport* do KAME IPSec API. O projeto FreeS/WAN foi descontinuado em 22/04/2003, na versão 2.06, após sua API original não ter sido escolhida para a adoção no Linux, acarretando a criação dos *forks* Openswan e strongSwan, (FREES/WAN, 2010), (OPENSWAN, 2009), (STRONGSWAN, 2009).

O objetivo inicial desses projetos foi dar continuidade ao projeto do FreeS/WAN mantendo a implementação de novos recursos desenvolvidos para o IPSec em Linux, nos *kernels* 2.4 e 2.6 com IPv4 ou IPv6, sendo possível utiliza-los para conexão com outras aplicações ou outras plataformas, desde que possuam suporte

ao IPSec, como por exemplo o Concentrador de VPN Cisco, o aplicativo SSH Sentinel, entre outras existentes no mercado.

O Openswan manteve-se mais próximo às características de configuração do FreeS/WAN, utilizando o arquivo */etc/ipsec.conf* para definição das regras de carregamento e parâmetros de conexão. Já no strongSwan este arquivo foi mantido apenas para compatibilidade, pois o mantenedor do projeto indica o uso do arquivo */etc/strongswan.conf* a fim de obter todos os recursos deste fork, porém este não será detalhado neste trabalho.

As seções definidas no arquivo */etc/ipsec.conf* são:

- config: mantém os parâmetros de carregamento do IPSec como nível de log, interface de rede, suporte ao NAT-T, entre outros fatores necessários para o aplicativo;
- conn: contém as características da conexão como rotas, endereçamento de rede, certificados digitais, tempo de expiração das chaves, tipo de autenticação, forma de iniciar a conexão e é claro o nome da conexão que deve ser declarado na primeira linha desta seção.

Outro arquivo também importante para estes aplicativos é o */etc/ipsec.secrets* que mantem as chaves usadas nas conexões ou caminhos para os certificados privados do host membro da conexão. Um exemplo de um arquivo contendo um chave PSK e o caminho para um certificado RSA pode ser visto na Figura 2.12

```
200.200.100.100: PSK "exemplo12233355555"
0.0.0.0: RSA /etc/ipsec.d/private/servidor.pem "senhacert2010"
```

Figura 2.12: Conteúdo de um arquivo ipsec.secrets

Nos dois certificados apresentados na Figura 2.12 foram especificados quais endereços IPs podem estabelecer uma conexão com o respectivo segredo, sendo possível adicionar mais de um tipo de certificado para ser usado em conexões IPSec distintas. E no diretório */etc/ipsec.d/private/* foi adicionado o certificado privado, pois tanto o Openswan com o strongSwan definem como padrão o local */etc/ipsec.d/* para armazenamento de arquivos como certificados e políticas de revogação ou autorização de conexões.

A automatização do processo de intercâmbio de chaves e criação das associações de segurança entre duas entidades é feito por meio do protocolo IKE que, em ambos os aplicativos é controlado pelo *daemon Pluto*, mas este *daemon* possui

suporte apenas ao IKEv1. Apenas o strongSwan possui suporte ao IKEv2, controlado pelo *daemon Charon* e permite realizar as associações de segurança com qualquer aplicativo IPSec que também utilize o IKEv2.

As principais características que distingue os dois aplicativos são apresentadas na Tabela 2.2. Onde os mais relevantes para este trabalho e que podem ser decisivos na escolha de uma das soluções para Linux são:

- **X.509 support**: ambos os aplicativos possuem suporte a este tipo de certificado que já lhes concede a criação de conexões com outras plataformas de sistemas que rodam IPSec;
- **L2TP multiple clients behind same NAT router**: o suporte a este recurso só é dado pelo Openswan e isso lhe dá vantagens em ambientes onde é comum receber conexões de diversos clientes estando estes por trás do mesmo roteador realizando NAT, sendo estes clientes estações Windows que conectão através do conjunto L2TP/IPSec;
- **Full FreeBSD, NetBSD, MacOSX support**: Para um concentrador de VPN quanto mais plataformas forem suportadas maior é a gama de clientes atendido pelo mesmo *gateway* IPSec, evitando o uso de outras soluções o que acarretaria mais trabalho para manter o ambiente funcional;
- **IKEv2**: O suporte a este protocolo dá recursos ao aplicativo para trabalhar com o NAT-T e só é encontrado no strongSwan, porém ambos os participantes da VPN precisam ter o mesmo suporte ou não será possível realizar conexões e, apesar do Openswan não possuir este recursos o aplicativo aplicativo permite o NAT-T através do ESP;
- **XAUTH via PAM**: O uso do PAM para permitir autenticação é um recurso importante que adiciona várias possibilidades de validação de acesso.

2.4.2 OpenVPN

Esta ferramenta foi descrita no trabalho (FIGUEIREDO, 2006), cabendo a este trabalho elucidar apenas alguns pontos importantes da ferramenta.

É uma ferramenta que permite a criação de VPNs com base no protocolo SSL/TLS, mas segundo o fabricante da ferramenta - (OPENVPN, 2010), a mesma trabalha nas camadas 2 ou 3 do modelo OSI e, não precisa de modificações no

Tabela 2.2: Comparativo entre Openswan e strongSwan. Fonte (OPENSWAN, 2009)

Feature	strongswan-4.1	openswan-3.0
X.509 support	yes	yes
Raw RSA key support	IKEv1 only	yes
RSA keys from DNS support	IKEv1 only	yes
KLIPS	Linux 2.4	Linux 2.4 + 2.6
MAST / Merged stack	no	yes
Fast ipsec starter	fast helper	integrated
Smartcard Interface	PKCS #11	OpenSC
Local CRL Caching	yes	no
CA Management	yes	no
SCEP client	yes	no
Attribute Certificates	yes	no
L2TP multiple clients behind same NAT router	no	yes
L2TP multiple clients on identical internal IP	no	yes
L2TP IPsec SA ref tracking	no	yes
IKEv2	yes	no
MOBIKE	yes	no
Windows IKE support	no	yes
Full FreeBSD support	no	yes
Full NetBSD support	no	yes
Full OpenBSD support	no	untested
Full Mac OSX support	no	yes
IPsec verify support	no	yes
IPsec livetest support	no	no
XAUTH via passwd file	yes	yes
XAUTH via PAM	no	yes
Aggressive Mode	no	yes
DNSSEC support	no	yes
Pluto DNS helper processes	no	yes
Pluto crypto helper processes	no	yes
Cryptographic Offload	no	yes (OCF)
Hardware offloading	no	yes (hifn,intel)
Hardware RNG support	no	yes (hifn)
TAProom support	no	yes
NAT-T forcing for ESP filter circumvention	IKEv2 only	yes

kernel para fazer complexas alterações na camada IP, pois roda sobre a *user-space*. Garantindo-lhe algumas características importantes:

- Permite a criação de túneis utilizando diversos sistemas operacionais, como Linux, Sun Solaris, *BSD, Mac OS X, Windows;
- Tunelamento em qualquer *subnet* IP ou adaptador virtual de *ethernet*, através de uma porta TCP ou UDP. Utilizando dispositivos virtuais TUN/TAP, mais detalhes em (KRASNYANSKY, 2009);
- Não necessita de conhecimento aprofundado no S.O., uma vez que não exige ajustes de *kernel* do sistema, pois roda na área de usuário e não na de sistema;
- Bibliotecas OpenSSL são usadas para autenticação e criptografia do túnel, podendo trabalhar com vários tipos de cifras, tamanhos de chaves e HMAC (*Keyed-Hashing for Message Authentication*). Mantém compatibilidade com certificados RSA e X.509 PKI;
- Túneis construídos por meio de chaves fixas précompartilhadas (*pre-shared keys*) ou chaves dinamicamente trocadas com base na troca de chaves do TLS.
- Não há dificuldades de criar túneis por trás de dispositivos com NAT, uma vez que trabalha acima da camada de rede;

Em resumo o OpenVPN é uma ferramenta com características importantes e que cria dois tipos de túneis: IPs Roteáveis (*Routing*) e *Ethernet* em Ponte (*Bridging Ethernet*). Mais detalhes sobre estes dois modos de trabalho podem ser vistos no site da comunidade OpenVPN, (OPENVPN, 2010).

2.4.3 POPTOP

Esta é a primeira e mais utilizada (ESSEBIER, 2009) ferramenta que implementa o serviço PPTPD a servidores Linux, permitindo que clientes PPTP estabeleçam conexões VPN sem dificuldade. Clientes com Windows 95/98/Me/NT/2000/XP/Vista/Win7 e Linux com algum cliente PPTP podem estabelecer conexões com as versões mais recentes da ferramenta.

Algumas de suas características são:

- Suporte os métodos de autenticação e criptografia usados pela Microsoft: MSCHAPv2 – evolução do método de autenticação MSCHAP; MPPE 40 – 128 bits RC4 para criptografia;

- Suporte para múltiplas conexões simultâneas;
- Não há necessidade de aplicações integradoras em ambientes com redes Microsoft com Radius;
- Suporte aos clientes Windows 95/98/Me/NT/2000/XP/Vista/Win7;
- Suporte aos clientes Linux.

Capítulo 3

Implantando VPN com IPSec

O uso do IPSec com Openswan como apresentado neste trabalho se deve às necessidades surgidas dentro da empresa onde o autor é um dos colaboradores, tais necessidades refere-se à conexões de rede seguras para outros colaboradores e fornecedores junto a companhia. Para ser mais específico, segue os dois principais pontos levantados:

1: Uma conexão VPN IPSec com o concentrador de VPN CISCO pertencente à IBM, com o intuito de prover acesso aos medidores de energia IP à rede da Companhia Paulista de Força e Luz, e que teve com orientação da IBM a compra ou aluguel de um equipamento com CISCO-PIX para evitar quaisquer dificuldades no acesso aos medidores;

2: Padronizar a conexão VPN dos vendedores remotos para facilitar o processo de manutenção do ambiente e reduzir variáveis de incidentes em ambos os pontos, pois diversos usuários tiveram como solução inicial o IPSec e por conta de mudanças no layout dos escritórios passaram a utilizar roteadores com NAT que impediam as conexões, onde foi dado como solução temporária o uso do OpenVPN e o *gateway* da empresa passou a suportar dois tipos de VPN;

Para trabalhar com uma única solução que atendesse ambos os casos foi necessário descartar o OpenVPN e seguir com uma solução homologada pela IBM, o IPSec. Os aplicativos Openswan e strongSwan foram avaliadas, mas por apresentar diversas falhas consecutivas de conexão com o concentrador CISCO o strongSwan foi

descartado e, o Openswan por ter sucesso na criação da VPN com o Concentrador CISCO atendeu à primeira necessidade da empresa.

Após o primeiro ponto ter sido resolvido, o passo seguinte foi a busca por uma solução de VPN com IPSec para atender clientes com acesso à internet atrás através de NAT, ou que possuam conexões ADSL de IPs dinâmicos, podendo ocorrer mais de uma conexão através do mesmo *gateway* do cliente. Para atender a este segundo ponto, a solução a ser aplicada é combinando o aplicativo Openswan com certificados digitais X.509, onde os certificados são udados na identificação dos usuários sem aplicação de DNS dinâmico ou IPs permanentes.

Então o foco deste trabalho concentra-se na apresentação de uma solução que atende aos usuários com perfil de *Road Warrior*, ou viajantes que não possuem um ponto permanente de acesso à internet mas que precisam ter acesso à rede da companhia e, a empresa necessita fornecer este acesso para manter seus recursos e dados disponíveis aos usuários remotos.

Para facilitar o entendimento do conteúdo a ser apresentado, foi criado um modelo de laboratório, onde o diagrama com os componentes usados na configuração do circuito com Openswan com Linux pode ser visto na Figura 3.1

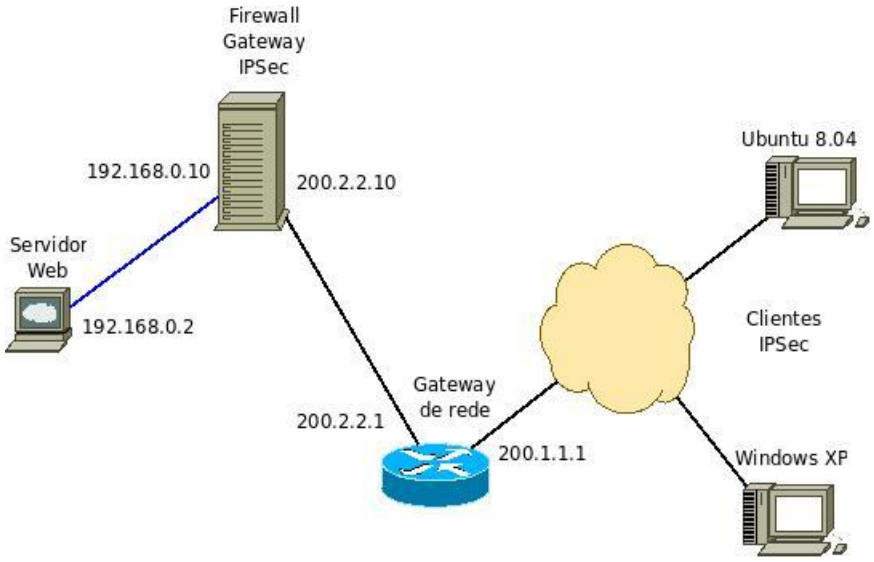

Figura 3.1: Diagrama de Testes

Os componentes apresentados no diagrama são:

- **Firewall/Gateway IPSec**: Openswan sobre o Debian 4.0 Etch com duas interfaces de rede (eth0 e eth1), tendo um importante papel no circuito por ser o concentrador dos túneis VPN;
- **Servidor Web**: Servidor Apache preparado sobre um Ubuntu Server 8.04, com apenas uma placa de rede conectada ao servidor IPSec;
- **Gateway de rede**: Um gateway com interfaces interligando a placa externa do servidor IPSec e a rede pública;
- **Clientes IPSec**: Dois equipamentos clientes de internet com endereços aleatórios, que criarão túneis com o servidor IPSec. Clientes: Ubuntu 8.04 e Windows XP.

3.1 Instalação

A forma utilizada de instalação de programas no Debian e no Ubuntu é através da ferramente APT (*Advanced Packaging Tool*), o que torna simples o processo. Então, como na maioria das instalações feitas por APT ou por *aptitude* e recomendado pelo manual *Referência Debian* - (AOKI, 2007), faz-se necessário a atualização do sistema e dos arquivos de índice presentes nos repositórios.

3.1.1 Instalação do Openswan

Tanto no cliente Linux como no servidor a instalação é feita através do mesmo comando, como está na Figura 3.2.

```
# apt-get update
# aptitude safe-upgrade
# apt-get install openswan
```

Figura 3.2: Atualiza APT

Caso seja solicitado a confirmação de instalação de novos pacotes para atender aos prérequisitos do Openswan, recomenda-se aceitar as recomendações para o funcionamento correto do IPSec no Linux.

Ao completar a instalação do Openswan a estrutura de diretório e arquivos a ser criada é apresentada na Tabela 3.1.

Tabela 3.1: Estrutura de pastas e arquivos do IPSec

/etc/ipsec.d	Contém certificados, configurações e referenciadas do ipsec.conf
/etc/ipsec.conf	Arquivo com a configuração inicial e parâmetro das Conexões
/etc/ipsec.secrets	Arquivo com as chaves usadas nas conexões

3.2 Criação dos Certificados

Os certificados a serem criados estão no formato X.509, por ser um padrão de mercado e pelo suporte dado pela ferramenta Openswan, sendo assim, todo o processo de criação deve ser executado no diretório it /etc/ipsec.d/.

No Apêndice A estão descritos os passos para criação de um novo conjunto de certificados para o novo CA e os demais certificados para os participantes da VPN. Durante a criação dos certificados são solicitados dados como Nome ou Sigla do país, assim como outras informações usadas na criação do mesmo. Para mais detalhes, ver (SPENNEBERG, 2009).

Em resumo, para criação dos certificados da CA e das estações participantes basta utilizar os comandos apresentados na Figura 3.3

```
Gera Certificado para a CA:

# perl /usr/lib/ssl/misc/CA.pl -newca

Gera certificado para as estações:

# perl /usr/lib/ssl/misc/CA.pl -newreq
# perl /usr/lib/ssl/misc/CA.pl -sign
```

Figura 3.3: Geração dos certificados X.509

3.3 Configuração do Openswan

A configuração do Openswan no servidor Concentrador IPSec e de um cliente Linux com Ubuntu 8.04 serão apresentadas nesta seção. Algumas das referências usadas na elaboração da melhor configuração podem ser encontradas em (LEEUW, 2009) e (OPENSWAN, 2009).

No Apêndice B está descrito como proceder para criar uma conexão em máquinas Windows XP utilizando o mesmo concentrador Openswan.

3.3.1 Configuração do Servidor IPSec

O primeiro arquivo a ser detalhado é o ipsec.conf, onde estão os parâmetros de carregamento do IPSec e das conexões, ilustrado na Figura 3.4.

```
# Config do Servidor IPSec
config setup
   interfaces=%defaultroute
   nat_traversal=no
   virtual_private=%v4:10.0.0.0/8,%v4:192.168.0.0/16,%v4:172.16.0.0/12

conn %default
   keyingtries=1
   compress=yes
   disablearrivalcheck=no
   authby=rsasig
   leftrsasigkey=%cert
   rightrsasigkey=%cert

conn roadwarrior
   pfs=no
   auto=add
   #
   left=200.2.2.10
   leftcert=/etc/ipsec.d/certs/debian-32.pem
   leftprotoport=17/1701
   #
   right=%any
   rightsubnet=vhost:%priv,%no
   rightprotoport=17/1701

#Disable Opportunistic Encryption
include /etc/ipsec.d/examples/no_oe.conf
```

Figura 3.4: Arquivo /etc/ipsec.conf no Servidor IPSec

Neste caso, tem-se: **config setup**: nesta seção foram alterados apenas dois parâmetros e os demais mantidos com os valores padrões, predefinidos pelo mantenedor do pacote openswan-2.4.6. Sendo eles:

- *interfaces*: Define as interfaces físicas e virtuais que o IPSec utilizará. O valor *%defaultroute*, força o programa procurar pela interface ligada ao roteador padrão.

- *nat_traversal*: É um parâmetro novo para contornar a falha do IPSec de não trabalhar atrás de mascaramento de IPs ou NAT. O valor "no" desativa este recurso no servidor.

- *virtual_private*: Parâmetro usado para definir subnets virtuais e é utilizado em conjunto com o valor vhost na definição da subnet, onde máquinas clientes poderão ter endereços dentro desta faixa atrás do NAT.

conn default: esta seção cria uma conexão com valores padrões usados em outras conexões, como a *roadwarrior*, os parâmetros definidos estão descritos:

- *keyingtries* – define o número de tentativas para concluir a conexão.

- *Compress* – este parâmetro permite que o servidor proponha o uso de uma compressão dos dados se ativo, devendo ocorrer antes da criptografia. Se o valor for "yes" ele estará ativo. Se o valor for "no" o protocolo não vai recomendar o uso da compressão, mas poderá aceitar se sugerido pela outra ponta.

- *disablearrivalcheck* – desativa a verificação de endereços no cabeçalho dos pacotes vindos do túnel. O valor "no" mantém a verificação ativa.

- *authby* – define como as duas pontas da conexão, servidor e cliente, se autenticarão. Podendo ser "rsasig" com uso de chaves RSA, "secret" para chaves précompartilhadas, e "never" para não haver tentativas de conexão ou nunca sejam aceitas.

- *leftrsasigkey* – contém a chave para o equipamento definido como lado esquerdo da conexão.

- *rightrsasigkey* – contém a chave para o equipamento definido como lado direito da conexão.

conn roadwarrior: esta seção cria a conexão que será utilizada pelos clientes IPSec com IPs dinâmicos, ou conhecidos como *Road Warrior*, tendo os seguintes parâmetros de conexão:

- *pfs* – sigla de “Perfect Forward Secrecy” é usado em conexões com chaves précompartilhadas e, por isso exige um sigilo no controle das chaves. O valor padrão desse parâmetro é “yes”.

- *auto* – define para o Openswan o que deve ser feito com esta conexão durante o carregamento do serviço. Os valores aceitos são *start* para a conexão ser iniciada no carregamento do IPSec, conexões fixas como ocorre entre dois *gateways* e, “add” para a conexão ser adicionada no ipsec mas ser iniciada manualmente, sendo usada nas conexões *roadwarrior* porque o cliente não permanece todo tempo conectado.

- *left* – endereço do host definido como sendo o lado esquerdo da conexão. Neste caso foi definido o IP do concentrador de VPN elucidado neste trabalho.

- *leftcert* – indica o nome e o caminho do arquivo com o certificado X.509 do host esquerdo, é o certificado gerado para o servidor IPSec, concentrador de VPN.

- *leftprotoport* – define o protocolo e a porta usados na autenticação da conexão para o lado do servidor. O número 17 representa o protocolo UDP e a porta 1701.

- *right* – endereço do host definido como sendo o lado direito da conexão. O valor “%any” representa qualquer endereço de internet.

- *rightsubnet* – define qual subnet o cliente deve ter, o uso do valor vhost:%no,%priv permite IPs reservados, definidos em virtual_private, ou endereços privados.

- *rightprotoport* – define o protocolo e a porta usados na autenticação da conexão. O número 17 representa o protocolo UDP e a porta 1701.

3.3.2 Configuração do Cliente IPSec

Seguem dois exemplos de configuração de cliente IPSec, Figura 3.5, um configurado para uma conexão sem NAT e outra com NAT, ou seja, quando o cliente precisa usar a internet através de um *firewall* ou *gateway* de rede. O Servidor não precisa de nenhuma alteração adicional para receber ambos os acessos.

A diferença entre as duas configurações está apenas em duas linhas. A primeira é o parâmetro de setup do IPSec nat_traversal que ativa suporte ao NAT. A outra

é no parâmetro de conexão right, que indica o endereço IP do cliente, devendo ser alterado sempre que o cliente obtiver um novo endereço.

Os parâmetros descritos na seção de configuração do servidor valem para as do cliente e por isso não serão repedidos nesta parte do documento.

```
# Cliente SEM NAT
config setup
   interfaces=%defaultroute
   nat_traversal=no

conn roadwarrior
   pfs=no
   auto=add
   #
   left=200.2.2.10
   leftcert=/etc/ipsec.d/certs/debian-32.pem
   leftprotoport=17/1701
   #
   right=200.1.1.2
   rightcert=/etc/ipsec.d/certs/laptop.pem
   rightprotoport=17/1701

# Cliente COM NAT
config setup
   interfaces=%defaultroute
   nat_traversal=yes

conn roadwarrior
   pfs=no
   auto=add
   #
   left=200.2.2.10
   leftcert=/etc/ipsec.d/certs/debian-32.pem
   leftprotoport=17/1701
   #
   right=192.168.10.2
   rightcert=/etc/ipsec.d/certs/laptop.pem
   rightprotoport=17/1701
```

Figura 3.5: Arquivo /etc/ipsec.conf no Cliente IPSec

3.4 Autenticação do IPSec

A forma de autenticação utilizada é através dos certificados X.509 criados para os clientes e para o servidor.

3.4.1 Configuração das chaves RSA

Outro arquivo importante na configuração do Openswan é o */etc/ipsec.secrets*. Neste arquivo podem ser definidas os tipos de chaves, endereços IPs ou nome do host completo, a chave ou o caminho para ela com a respectiva senha se possuir. Na Figura 3.6 e Figura 3.7 há o conteúdo dos arquivos de senhas para o servidor e no cliente utilizados neste trabalho.

```
# Arquivo ipsec.secrets para o servidor IPSec
# Chave privada RSA para o servidor, usada na autenticação de
# outros hosts que tenham a chave publica

%any: RSA /etc/ipsec.d/private/debian-32.key "ufla2009"
```

Figura 3.6: Primeiro exemplo de arquivo /etc/ipsec.secrets

```
# Arquivo ipsec.secrets para o Cliente Cliente
# Chave privada RSA para o servidor, usada na autenticação de
# outros hosts que tenham a chave publica

: RSA /etc/ipsec.d/private/laptop.key "ufla2009"
```

Figura 3.7: Segundo exemplo de arquivo /etc/ipsec.secrets

Onde:

- %any – Valor informado antes da chave privada do servidor e permite que um cliente com qualquer IP faça autenticação fornecendo a respectiva chave pública. Também pode ser usado valores 0.0.0.0 ou deixar sem preenchimento antes do ":" para ter o mesmo efeito. Outros valores aceitos são endereços IPs dos hosts que podem fazer autenticação com chaves ao servidor.
- RSA – Indica o tipo de chave, podem ser definidas dois tipos: RSA e PSK;
- /etc/ipsec.d/private/debian-32.key – caminho completo do arquivo com a chave privada do servidor;

- “ufla2009” - senha para autenticação da chave primária do servidor.

3.5 Riscos e Medidas preventivas

Alguns pontos importantes sobre segurança do servidor IPSec devem ser levados em consideração na implementação das VPNs, como:

- Permissões de arquivos e pastas: para o Openswan os arquivos que devem ser protegidos são os que contém as chaves pública e privada, assim como as senhas dos certificados não devem ser de conhecimento público e por esse motivo devem receber permissões especiais a fim de impedir acesso aos arquivos por usuários não autorizados. Na Figura 3.8 são definidas as permissões para os arquivos com as chaves do servidor, mas esse mesmo processo deve ser aplicado nas chaves do cliente.

```
# chmod 600 /etc/ipsec.secrets
# chmod 400 /etc/ipsec.d/private/debian-32.key
# chmod 700 /etc/ipsec.d/private
```

Figura 3.8: Permissões para arquivos Openswan

- Controle de acesso ao servidor, como exemplo o iptables pode ser usado para controlar o acesso ao servidor vindos da internet. Na Figura 3.10 estão algumas regras para serem usadas na liberação das portas usadas durtante a ativação de um circuito VPN;

- Controle de acesso à LAN privada: Uso do firewall para controlar os pacotes que entram e saem da rede Privada. Como exemplo, para para clientes VPN que tenham disponível apenas os serviços de HTTP e RDP então a liberação deve ser restrita aos serviços:

- Cuidados no transporte das chaves: Esse cuidado é devido ao certificado do servidor IPSec que deve ser levado ao cliente de uma forma segura. Para os clientes Linux/Unix os arquivo “debian-32.pem e laptop.pem” ou, para clientes Windows o arquivo “cliente-ipsec.p12” , poderiam ser transferidos via SCP ou SFTP.

```
# Negociação IKE
$IPT -A OUTPUT -p udp --sport 500 -j ACCEPT
$IPT -A INPUT -p udp --dport 500 -j ACCEPT

# Autenticação e Criptografia - ESP
$IPT -A INPUT -p 50 -j ACCEPT
$IPT -A OUTPUT -p 50 -j ACCEPT

# Autenticação de cabeçalho - AH
$IPT -A INPUT -p 51 -j ACCEPT
$IPT -A OUTPUT -p 51 -j ACCEPT

# Conexões L2TP para Windows
$IPT -A OUTPUT -p udp --sport 1701 -j ACCEPT
$IPT -A INPUT -p udp --dport 1701 -j ACCEPT

# Usado para o NAT-T: Aplicado quando o IPSec está atrás do NAT
$IPT -A OUTPUT -p udp --sport 4500 -j ACCEPT
$IPT -A INPUT -p udp --dport 4500 -j ACCEPT
```

Figura 3.9: Controle de acesso ao Servidor IPSec

```
# ETH0 - Placa externa
# ETH1 - Placa interna
iptables -P FORWARD DROP
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
iptables -A FORWARD -p tcp -dport 80 -o eth1 -d 192.168.0.2 -j ACCEPT
iptables -A FORWARD -p tcp -dport 3389 -o eth1 -d 192.168.0.3 -j ACCEPT
iptables -A FORWARD -m limit --limit 3/minute --limit-burst 3 -j LOG \
 --log-prefix "Pacote FORWARD perdido: "
```

Figura 3.10: Controle de acesso aos serviços que passam pela VPN

Capítulo 4

Resultados Obtidos

Esta solução de VPN com IPSec permite que qualquer empresa que tenha funcionários trabalhando remotamente, viajantes ou escritórios de representantes com usuários que utilizam os recursos de dentro da rede da companhia. Isso é conseguido sem a preocupação com faixas de endereços IPs dos usuários remotos, com versões do aplicativo de VPN ou se o recurso disponibilizado para estes usuários garante a segurança no tráfego dos dados.

A solução foi aplicada em uma empresa de 1000 funcionários, sendo 5 vendedores com equipamentos portáteis utilizando outros sistemas operacionais e 100 escritórios de representações espalhados por todo o território nacional.

Todos os escritórios de representações possuiam inicialmente conexões internet em um único computador ou link dedicado para criar a VPN com a empresa, e no lado da companhia era mantido em um gateway Linux com FreeSwan sem o suporte a NAT-T. Sendo que para atender os vendedores foi instalação o OpenVPN, por utilizarem conexões com a internet em hoteis, shopping center ou aeroportos, sempre por trás de um NAT.

A Figura 4.1 mostra um diagrama sobre as conexões VPN realizas pela empresa.

Os ganhos obtidos com esta solução foram:

- Uso de uma única solução de VPN, reduzindo a necessidade de mais recursos para o Gateway de VPN, menos tempo gasto com a manutenção do servidor e dos clientes, como também a redução no número de portas abertas para o *Gateway/Firewall*;

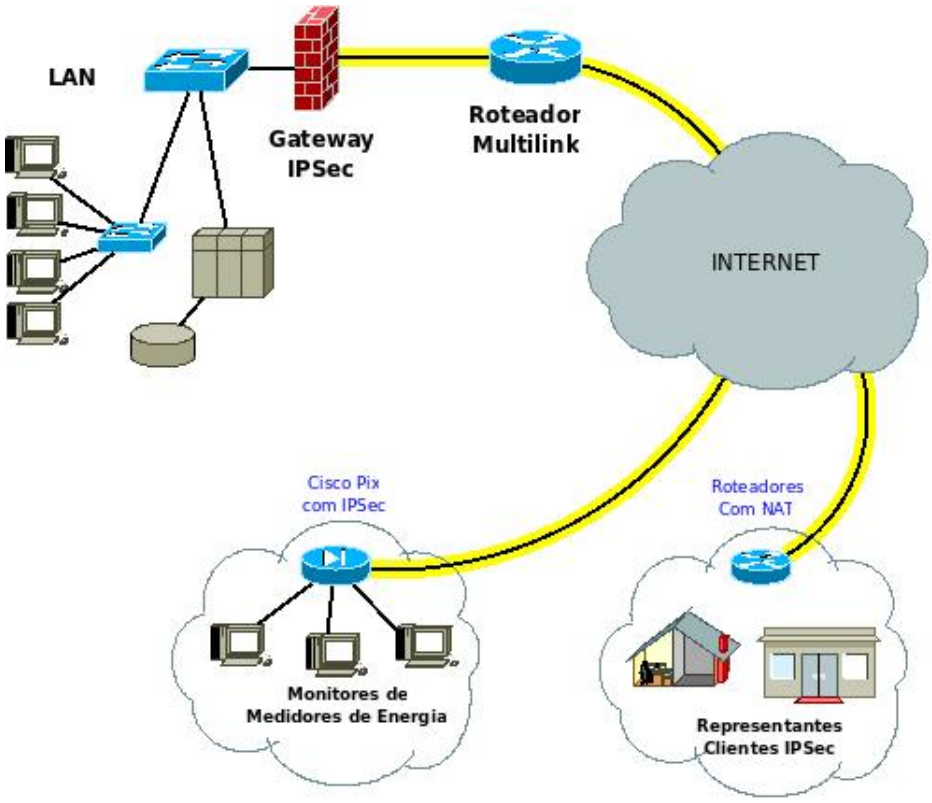

Figura 4.1: Circuito de uma empresa com a solução demonstrada

- As conexões podem ser feitas de qualquer ponto da Internet, independente do endereço IP utilizado ou se há ou não NAT entre o Cliente e o Servidor;
- Ao usar o OpenVPN era comum ter clientes com versões diferentes do aplicativo, a solução eliminou a necessidade de controlar versões do servidor e dos clientes com OpenVPN;
- Como a empresa já dispunha de uma VPN IPSec com a Companhia de Energia e outra com o Gateway da Filial, foi adicionado uma nova conexão para atender os clientes *Road Warrior*;
- Não há a necessidade de manter um DNS dinâmico para aumentar a segurança das conexões;
- redução no tráfego de dados com o uso da compressão de pacotes presente no IPSec;

Para demonstrar o nível de segurança e volume dos pacotes trafegados pelo gateway, foram capturados todos os pacotes que passaram pelo primeiro nós da rede, após o Gateway IPSec, através das ferramentas TCPDUMP e Wireshark. Onde o resultado da análise dos pacotes coletados estão apresentados nas figuras seguintes. A Figura 4.2 exibe o conteúdo transidado durante uma consulta HTTP sem nenhum tipo de segurança. A Figura 4.3 exibe o resultado da mesma consulta, porém passando através do circuito VPN presente entre o Gateway IPSec e o cliente remoto.

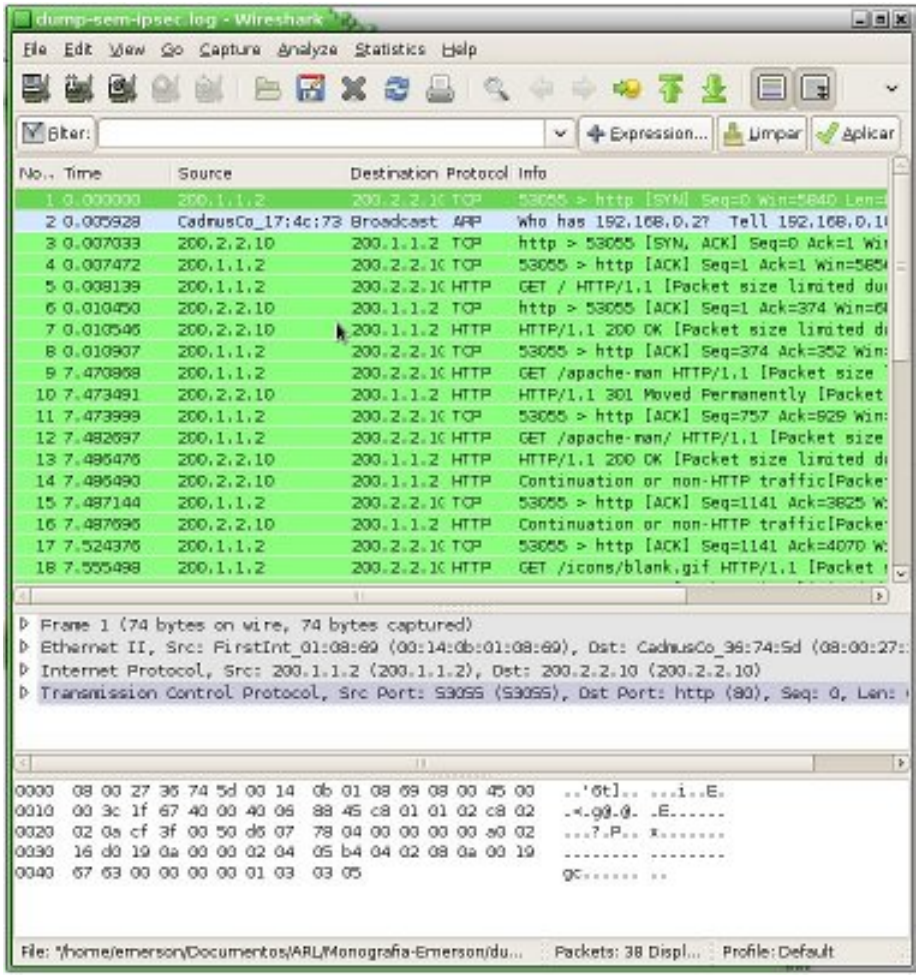

Figura 4.2: Pacotes coletados com TCPDUMP sem VPN

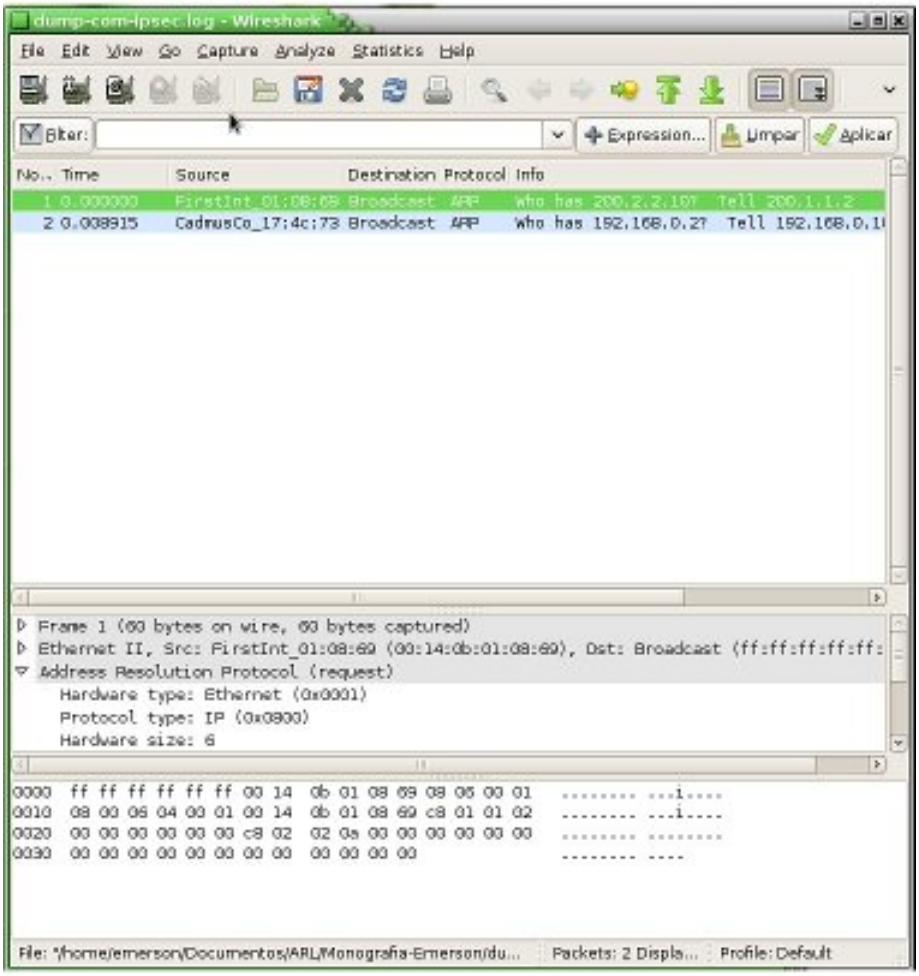

Figura 4.3: Pacotes coletados com TCPDUMP com VPN

Capítulo 5

Conclusão

O objetivo deste trabalho foi apresentar uma solução de VPN através do protocolo IPSec, utilizando a ferramenta Openswan, para clientes que fazem uso da internet através de conexões com NAT ou com endereços IPs variados, uma conexão conhecida como "Road Warrior".

O trabalho descreveu alguns dos principais métodos de criptografia existentes no mercado para contrução de Túneis VPN, como é o caso do SSL/TLS. E também algoritmos de criptografia que são usados em chaves de autenticação de túneis ou de cifragem dos pacotes de dados traferidos através dos túneis.

Outros protocolos além do IPSec foram apresentados para mostrar algumas das características de outras soluções existentes no mercado, com o intuito de enfatizar as vantagens do IPSec.

No IPSec alguns ajustes foram feitos ao longo do tempo e no Linux a aplicação Openswan acompanhou as melhorias do protocolo, ganhando recursos para atender as necessidades dos usuários que utilizam formas de conexões variadas como o uso de NAT ou viajam constantemente e não podem depender de um ponto de acesso fixo para ter segurança na traferência de dados confidenciais.

Os maiores ganhos obtidos e destacado neste trabalho, foi a unificação de uma solução através do protocolo IPSec e do aplicativo Openswan, que permite a conexão de redes virtuais privadas com gateways CISCO, Linux, Unix ou estações de trabalho Linux ou Windows, com segurança e sem restrições para as formas de acesso dos usuários.

Como projeto futuro pretendo integrar circuitos VPN com DMVPN (*Dynamic Multipoint VPN*), isso consiste na interligação de várias redes privadas através da combinação dos protocolos IPSec, GRE e NHRP (*Next Hop Resolution Protocol*), onde para Linux há o projeto (OPENNHRP, 2010) que está em desenvolvimento, mas trata-se de uma solução indicada pela CISCO para interligar várias unidades de negócio de uma companhia aos datacenters de forma redundante.

Referências Bibliográficas

AOKI, O. *Referência Debian*. http://qref.sourceforge.net/Debian/reference/reference.pt-br.pdf, 2007.

BURNETT, S.; PAINE, S. *Criptografia e Segurança, O Guia Oficial RSA*. 1. ed. Rio de Janeiro: Campus, 2002.

ESSEBIER, T. *Poptop - The PPTP Server for Linux*. http://www.poptop.org/, 10 dez. 2009.

FIGUEIREDO, D. A. *VPN'S DE BAIXO CUSTO COM DNS DINÂMICO E ADSL*. Dissertação (Monografia ARL) — UFLA, 2006.

FOWLER, D. *Virtual Private Networks: making the right connections*. San Francisco: Morgan Kaufmann, 1999.

FREES/WAN. *Linux FreeS/WAN*. http://www.freeswan.org/: [s.n.], jun. 2010.

G1. Petrobras afirma que informações sigilosas foram furtadas. *G1*, http://g1.globo.com/Noticias/Economia_Negocios/0,,MUL298824-9356,00-INFORMACOES+SIGILOSAS+SAO+FURTADAS+DA+PETROBRAS.html, 14 fev. 2008.

KRASNYANSKY, M. *Virtual Tunnel*. http://vtun.sourceforge.net/, 14 dez. 2009.

KUROSE, J. F.; ROSS, K. W. *Redes de Computadores e a Internet*. 3. ed. São Paulo: Pearson, 2007.

LEEUW, J. d. *Using a Linux L2TP/IPsec VPN server*. http://www.jacco2.dds.nl/networking/openswan-l2tp.html, 5 abr. 2009.

LEWIS, M. *Troubleshooting virtual private networks*. 1. ed. Indianapolis,USA: Cisco Press, 2004.

Network Working Group. *Point-to-Point Tunneling Protocol (PPTP)*. http://www.rfc-editor.org/rfc/rfc2637.txt, jul. 1999.

OPENNHRP. *Projeto OpenNHRP*. http://sourceforge.net/projects/opennhrp/, mar. 2010.

OPENSWAN. *Openswan Documentation*. http://www.openswan.org/docs/, 10 abr. 2009.

OPENVPN. *OpenVPN Documentation*. http://www.openvpn.net/index.php/open-source/documentation.html, 10 jan. 2010.

PETERSON, L. L.; DAVIE, B. S. *Redes de Computadores*. 1. ed. Rio de Janeiro: Campus, 2004.

PIAT, F. *IPSec no Debian*. 20. ed. http://wiki.debian.org/IPsec, 16 mar. 2009.

SONICWALL. *Aventail*. www.aventail.com: [s.n.], ago. 2009.

SPENNEBERG, R. *IPSec HOWTO*. http://www.ipsec-howto.org/, 9 jul. 2009.

STEWART JAMES MICHAEL, T. E.; CHAPPLE, M. *CISSP: Certified Information Systems Security Professional Study Guide*. 4. ed. Indianapolis, USA: Wiley, 2008.

STRONGSWAN. *the OpenSource IPsec-based VPN Solution for Linux*. http://www.strongswan.org/: [s.n.], jun. 2009.

TANENBAUM, A. S. *Redes de Computadores*. 4a. ed. Rio de Janeiro: Campus, 2003.

TECHNET. *Protocolo de encapsulamento ponto a ponto*. http://technet.microsoft.com/pt-br/library/cc738852%28WS.10%29.aspx: [s.n.], 5 jan. 2010.

TECHNET. *Virtual private network and intranet security*. http://technet.microsoft.com/pt-br/library/bb742458%28en-us%29.aspx: [s.n.], 5 jan. 2010.

TOWNSLEY, W.; VALENCIA, A.; RUBENS, A.; PALL, G.; ZORN, G.; PALTER, B. *Layer two tunneling protocol l2tp*. http://www.ietf.org/rfc/rfc2661.txt, 15 jan. 2010.

UCHÔA, J. Q. *Segurança computacional*. 2. ed. Lavras: UFLA/FAEPE, 2005.

Apêndice A

Criação dos Certificados Digitais

A.1 Criação da Autoridade Certificadora - CA

A Figura 1A.1 apresenta os passos para criação de um novo certificado para uma Autoridade Certificadora - CA, com este certificado serão criados os certificados dos equipamentos que poderão estabelecer uma VPN. Durante a criação dos certificado são solicitados dados como Nome ou Sigla do país, asim como outras informações que usadas pelo certificado.

É possível prolongar o tempo de validade do certificado, para isso, execute manualmente como na Figura A.2. Neste exemplo o tempo definido é de 10anos e, o padrão de criação é de 1 ano. Outra atividade apresentada nesta figura é a criação do arquivo CRL que será usado no gateway:

A.2 Criação do certificado para o servidor

Assim com no certificado CA também é solicitado informações sobre o local e o responsável pelo certificado do servidor. Os passos de criação estão descritos na Figura A.3.

Após o certificado do servidor ser criado ele precisa ser assinado pelo CA, para isso outras informações são importantes afim de garantir a segurança do certificado. Os passos deste processo podem ser vistos na Figura A.4.

```
# perl /usr/lib/ssl/misc/CA.pl -newca

CA certificate filename (or enter to create)
<enter>
Making CA certificate ...
Using configuration from /usr/lib/ssl/openssl.cnf
Generating a 1024 bit RSA private key
.....................................................................+++
.....................................+++
writing new private key to './demoCA/private/./cakey.pem'

Enter PEM pass phrase: < informe a senha para criar o certificado >

Verifying password - Enter PEM pass phrase: < repita a senha >

You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:BR <Use a sigla do país>
State or Province Name (full name) [Some-State]: <Use a sigla do estado>
Locality Name (eg, city) []: <Informe a Cidade>
Organization Name (eg, company) [Internet Widgits Pty Ltd]: ARL107 <Nome da Empresa>
Organizational Unit Name (eg, section) []:<enter>
Common Name (eg, YOUR name) []:Emerson Galeli
Email Address []:ca@servidor.com.br <enter>
```

Figura A.1: Criação de um certificado CA

```
Prolongar a tempo da chave:
# cd demoCA
# openssl x509 -in cacert.pem -days 3650 -out cacert.pem -signkey ./private/cakey.pem

Criar arquivo CRL para o Gateway:
# openssl ca -gencrl -out crl.pem
```

Figura A.2: Alterar validade do CA e criar arquivo CRL

Com os certificados prontos, basta envia-los para os devidos diretórios como pode ser visto na Figura A.5:

```
# perl /usr/lib/ssl/misc/CA.pl -newreq

Using configuration from /usr/lib/ssl/openssl.cnf
Generating a 1024 bit RSA private key
.....................................+++
.............................+++
writing new private key to 'newreq.pem'
Enter PEM pass phrase:<entre com a senha do certificado>
Verifying password - Enter PEM pass phrase:<repita a senha>
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:BR <enter>
State or Province Name (full name) [Some-State]:SP <enter>
Locality Name (eg, city) []:Bauru <enter>
Organization Name (eg, company) [Internet Widgits Pty Ltd]: ARL107 <enter>
Organizational Unit Name (eg, section) []: <enter>
Common Name (eg, YOUR name) []: servidor.com.br
Email Address []:admin@servidor.com.br

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: <enter>
An optional company name []: <enter>
Request (and private key) is in newreq.pem
```

Figura A.3: Criação do certificado para o Servidor

A.3 Criação do certificado para o clientes

Utilize o mesmo comando usado no processo de criação do certificado do servidor, diferindo apenas no preenchimento dos campos *commonName, emailAddress e password* solicitados pelo script de criação.

```
# perl /usr/lib/ssl/misc/CA.pl -sign

Using configuration from /usr/lib/ssl/openssl.cnf
Enter PEM pass phrase: <senha usada na criação do CA>
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName :PRINTABLE:'BR'
stateOrProvinceName :PRINTABLE:'SP'
localityName :PRINTABLE:'City'
organizationName :PRINTABLE:'ARL107'
commonName :PRINTABLE:'servidor.com.br'
emailAddress :IA5STRING:'admin@servidor.com.br'
Certificate is to be certified until Feb 13 16:28:40 2010 GMT (365 days)
Sign the certificate? [y/n]:y <enter>

1 out of 1 certificate requests certified, commit? [y/n]y <enter>
Write out database with 1 new entries
Data Base Updated
(certificate snipped)
Signed certificate is in newcert.pem
```

Figura A.4: Validação do certificado do Servidor com o CA

```
# mv newcert.pem servidor.pem
# mv newkey.pem servidor.key
# cp servidor.key private/
# cp servidor.pem certs/
# cp demoCA/cacert.pem cacerts/
```

Figura A.5: Mover Certificados para os locais definitivos

A.3.1 Criação do Certificado para clientes Windows

Para o Cliente com Windows, faz-se necessário criar um certificado no formato .p12, para cria-lo use o comando apresentado na figura A.6:

```
# openssl pkcs12 -export -in newcert.pem -inkey newkey.pem -certifile \
demoCA/cacert.pem -out newcert-client.p12
```

Figura A.6: Cria certificado no formato P12

Apêndice B

Conexão VPN para Clientes "Road Warrior" Windows

B.1 Instalação do L2TP e PPP no Debian

L2TP será utilizado em conjunto com o PPP para fazer a autenticação do túnel para os clientes com Windows. Na Figura B.1 está o comando usado para instalar o servidor L2TP no Gateway com Debian:

```
# apt-get install ppp l2tpd
```

Figura B.1: Instalar PPP e L2TP no servidor Linux

As pastas com os arquivos de configuração do servidor são /etc/ppp/ e /etc/l2tpd/.

B.1.1 Configuração do L2TP e do PPP no Debian

O primeiro arquivo a ser configurado é o /etc/l2tpd/l2tpd.conf, Figura B.2, onde são definidos os parâmetros de carregamento do L2tp, tais como: faixa de ip dos clientes Windows herdados na ativação do túnel, a porta de conexão, protocolos que podem ou não ser usados na autenticação, entre outros.

As demais configurações devem ser feitas para o PPP, onde estão apresentados na Figura B.3, Figura B.4 e Figura B.5

```
; Arquivo /etc/l2tpd/l2tpd.conf
[global]
port = 1701
[lns default]
ip range = 192.168.1.15-192.168.1.30
local ip = 200.2.2.10
length bit = yes
require chap = yes
refuse pap = yes
require authentication = yes
name = Debian-32
ppp debug = yes
pppoptfile = /etc/ppp/options.l2tpd
```

Figura B.2: Arquivo l2tpd.conf

```
# Arquivo /etc/ppp/options.l2tpd
ipcp-accept-local
ipcp-accept-remote
auth
crtscts
idle 1800
mtu 1410
mru 1410
debug
lock
proxyarp
connect-delay 5000
noccp
nodefaultroute
refuse-pap
require-mschap-v2
```

Figura B.3: Arquivo de Opções do L2TPD

Após tudo configurado basta carregar o serviço do L2TPD e o servidor já estará pronto para receber conexões dos clientes Winsows.

```
# Arquivo /etc/ppp/options
asyncmap 0
auth
crtscts
lock
hide-password
modem
+mschap-v2
proxyarp
lcp-echo-interval 30
lcp-echo-failure 4
noipx
```

Figura B.4: Arquivo de Opções do PPP

```
# Arquivo /etc/ppp/chap-secrets
# Secrets for authentication using CHAP
# client         server             secret           IP addresses
usuario        Debian-32        "senha-cliente"            *
```

Figura B.5: Arquivo de senhas dos usuários VPN

B.1.2 Importar Chave X.509 no Windows

A chave a ser importada deve estar no formato P12, veja detalhes de criação no Apêndice B. Com o certificado em mãos faça a importação no Windows utilizando a ferramenta Certimport para facilitar o processo, podendo seguir os passos:

1:Caso não tenho no computador o aplicativo Certimport faça download do site: `ftp://ftp.openswan.org/openswan/windows/certimport/'`

2:Copie o arquivo no formato .P12, contendo a chave assinada pela CA do servidor com IPSec, para a pasta onde está o aplicativo Certimport no micro Windows;

3:Faça a importação do certificado com o comando apresentado na Figura B.6.

```
C:\certimport> certimport.exe -p senha-da-chave arquivo.p12
```

Figura B.6: Comando usado para importar chave no Windows

B.1.3 Configuração do L2TP e do PPP no Windows XP

Crie uma nova conexão VPN definindo um nome para esta, com o endereço do host sendo o do servidor IPSec apresentado neste trabalho, 200.2.2.10, depois confira as seguintes propriedades:

- Defina em Tipo de VPN, na aba Rede das propriedades da conexão, como sendo “L2TP IPSec VPN”;
- Na aba Segurança habilite a opção Avançada, e nas configurações de segurança avançadas permita apenas o protocolo MS-CHAP v2.