AVALIAÇÃO DE PROTOCOLOS DE AUTENTICAÇÃO EM REDES SEM FIO

Demetrio de Souza Diaz Carrión

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Luís Felipe Magalhães de Moraes, Ph. D.

Prof. Cláudio Luís de Amorim, Ph. D.

Prof. Noemi de La Rocque Rodriguez, Dr.

RIO DE JANEIRO, RJ - BRASIL

AGOSTO DE 2005

CARRIÓN, DEMETRIO DE SOUZA DIAZ

Avaliação de Protocolos de Autenticação em Redes sem Fio [Rio de Janeiro] 2005

XIX, 145 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2005)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Redes sem Fio
2. Segurança
3. Autenticação
4. Troca de Chaves
5. Detecção de Desligamento de Estação

I. COPPE/UFRJ II. Título (série)

Dedicatória

Aos meus avós Demetrio Diaz Lamas (in memoriam)e Josefa Carrión Alvarez (in memoriam) que me amaram incondicionalmente.

Aos meu fillhos os quais amo incondicionalmente.

Agradecimentos

Aos meus pais, Demetrio Diaz Carrión e Roseli Odete de Souza Diaz Carrión.

Às minhas irmãs Sabrina de Souza Diaz Carrión e Marcela Rodrigues de Souza Diaz Carrión

À grande companheira de meu pai e amiga Fátima Rodrigues Vieira de Souza Diaz Carrión e a todos os meus familiares.

À minha noiva Joana de Castro Caldeira pelo amor e serenidade.

Ao meu orientador, Professor Luis Felipe Magalhães de Moraes pelo acompanhamento não só no mestrado, mas sim desde a graduação no DEL/UFRJ.

Aos membros da banca, Professora Noemi Rodriguez e Professor Cláudio Amorim.

Aos amigos Luciano Albuquerque e Alexandre Pinaffi pela ajuda ténica e não técnica, discussões tecnológicas e sugestões.

Aos demais amigos do Laboratório Ravel.

À Ernst & Young, nas figuras de meu gerente Maurício Saad e diretor Francesco Botino pela compreensão nesta reta final do mestrado.

À FAPERJ por financiar parte da pesquisa.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

AVALIAÇÃO DE PROTOCOLOS DE AUTENTICAÇÃO EM REDES SEM FIO

Demetrio de Souza Diaz Carrión

Agosto/2005

Orientador: Luís Felipe Magalhães de Moraes

Programa: Engenharia de Sistemas e Computação

O objetivo deste trabalho é propor, implementar e avaliar protocolos de autenticação e troca de chaves, além de um mecanismo de Detecção de Desligamento de Estações (DDE), aplicados a redes sem fio; em particular, redes baseadas no protocolo IEEE 802.11b. São analisados diversos protocolos de autenticação e troca de chaves com relação a um conjunto de métricas que foram definidas ao longo da pesquisa para esta dissertação. Inovações são propostas quando da utilização de protocolos considerados o estado da arte como o SRP (Secure Remote Protocol) e novas propostas de segurança para o protocolo DHCP (Dynamic Host Configuration Protocol). A aplicação do protocolo SRP é inovadora nas redes sem fio e a implementação de DHCP com segurança feita é a primeira seguindo a estrutura em (Droms, 2001). O mecanismo de DDE denominado isAlive-2 contém diversos aprimoramentos de segurança, projeto e implementação quando comparado a sua proposta inicial isAlive-1 apresentada em (Carrión, 2003a) e (Nunes, 2004).

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

WIRELESS NETWORKS AUTHENTICATION PROTOCOLOS EVALUATION

Demetrio de Souza Diaz Carrión

August/2005

Advisor: Luís Felipe Magalhães de Moraes

Department: Systems Engineering and Computer Science

The main goal of this work is to propose, implement and evaluate authentication, key exchange and Dead Peer Detection (DPD) protocolos applied to wireless networks; particuraly wireless networks based on IEEE 802.11b. This dissertation analyzes serveral authentication and key exchange protocolos according to a set of metrics defined throughout the research process.The application of state-of-the-art authentication and key exchange protocols like SRP (Secure Remote Protocol) and new proposal regarding secure DHCP (Dynamic Host Configuration Protocol) applied to wireless networks are major contributions outlined in this dissertation. The application of the SRP in wireless networks is innovative and the implementation of the secure DHCP is the first following the framework defined in (Droms, 2001). The Dead Peer Detection mechanism named isAlive-2 has several security, design and implementation improvements compared to the first version isAlive-1 previously presented in (Carrión, 2003a) and (Nunes, 2004).

Lista de Acrônimos

AAA : Authentication, Authorization and Accounting;

AC : Autoridade Certificadora;

ACK : Acknowledged;

ACM : Association of Computer Machinery;

ADSL : Assynchronous Subscriber Digital Line;

AKE : Asymmetric Key Exchange;

AP : Access Point;

API : Application Programming Interface;

ARP : Address Request Protocol;

ATCCID : Autenticação, Troca de Chaves, Controle de Acesso, Confidencialidade, Integridade e Detecção de Desligamento de Estação ;

BRAN : Broadband Radio Access Networks;

BSS : Basic Service Set;

BSSID : Basic Service Set Identifier;

CA : Certification Authority;

CLD : Cálculo do Logarítmo Discreto;

CR : Challenge-Response;

CRC : Cyclical Redundancy Check;

CRL : Certification Revocation List;

CSMA : Carrier Sense Multiple Access;

CSMA/CA : Carrier Sense Multiple Access with Collision Avoidance;

CSMA/CD : Carrier Sense Multiple Access with Collision Detection;

CVS : Concurrent Version System;

DDE : Detecção de Desligamento de Estação;

DHC : Dynamic Host Configuration;

DHCP : Dynamic Host Configuration Protocol;

DMZ : Demilitarized Zone;

DNS : Domain Name Service;

DoS : Denial of Service;

DPD : Dead Peer Detection;

DTPT : Disclosure Time to Patch Time;

EAP : Extensible Authentication Protocol;

EKE : Exponencial Key Exchange;

EP : Equivalent Plaintext;

ERB : Estação Rádio Base:

ESS : Extended Service Set;

ESSID : Extended Service Set Identifier;

ETC : Equivalência ao Texto Claro;

ETSI : European Telecommunications Standards Institute;

FNP : Fatoração de Números Primos;

GUI : Graphical User Interface;

HDLC : High Level Data Link Control;

HIPERLAN : High Performance Radio Local Area Network;

HMAC : Hashed Message Authentication Code;

HomeRF : Home Radio Frequency;

HTTP : Hypertext Transfer Protocol;

HTTPS : Hypertext Transfer Protocol Secure;

IAPP : Inter-Access Point Protocol;

ICMP : Internet Control Message Protocol ;

ICP : Infra-estrutura de Chave Pública;

IEEE : Institute of Eletric and Eletronic Engineers;

IKE : Internet Key Exchange;

IETF : Internet Engineering Task Force;

IFH : Inversão de Função de Hash;

IP : Internet Protocol;

IPSec : Internet Protocol Secure;

ISM : Industrial, Scientific and Medical;

IV : Initialization Vector;

LAN : Local Area Network;

MAC : Medium Access Control;

MAC : Message Authentication Code;

MD5 : Message Digest Version 5;

NACK : Not Acknowledged;

NIS : Network Information System;

OCSP : Online Certificate Status Protocol;

OSI : Open System Interconnection;

PA : Ponto de Acesso;

PFS : Perfect Forward Secrecy;

PKI : Public Key Infrastructure;

PRNG : Pseudo Random Number Generator;

PSK : Pre-Shared Key;

QoS : Quality of Service;

RADIUS : Remote Authentication Dial In User Service;

RFC : Request for Comments;

S-ALOHA : Sloted-ALOHA;

SPEKE : Simple Password-Authenticated Exponential Key Exchange;

SRP : Secure Remote Protocol;

SSH : Secure Shell;

SSID : Service Set Identifier;

SSL : Secure Socket Layer;

STA : Station (Estação);

TACACS+ : Terminal Access Controller Access Control System ;

TCP : Transmission Control Protocol;

TLS : Transport Layer Security;

TKIP : Temporary Key Integrity Protocol;

VI : Vetor de Inicialização;

VPN : Virtual Private Network;

UDP : User Datagram Protocol;

URL : Universal Resource Locator;

WAVE : Wireless Ability in Vehicular Environments;

WEP : Wired Equivalent Privacy;

WINS : Windows Internet Naming Service;

WPA : Wi-Fi Protected Access;

WLAN : Wireless Local Area Network;

WiMax : Wireless Metropolitan Area Extended

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A Segurança da informação é primariamente composta por três pilares fundamentais, sendo eles a confidencialidade, a integridade e a disponibilidade. Em suma a confidencialidade garante que somente as entidades permitidas poderão entender o significado da informação; a integridade garante que a informação não foi modificada desde que foi gerada; a disponibilidade garante que somente as entidades autorizadas terão acesso à informação e que estas sempre estarão disponíveis quando solicitadas (STALLINGS, 1999).

Estes três pilares são aplicados na segurança de redes de computadores como um todo e em particular na segurança das redes sem fio, encontrando nestas a aplicação da pesquisa desta dissertação.

Desde o surgimento da Internet, em especial com a grande disseminação comercial no Brasil em 1995 (MAZZEO, 2000), verificou-se um grande aumento nos ataques virtuais, que têm por alvo os pilares da segurança acima descritos.

Estes ataques são direcionados na exploração de falhas de definição de protocolos, falhas de implementação, erros de operação de sistemas entre outros. Isto se aplicou largamente em casos da Internet como falhas no SSL (WAGNER, 1995), ataques de negação de serviço (UnixInsider, 2001), distribuição indiscriminada de *worms* (Blaster, 2003), (Sasser, 2004) e outros.

Estes ataques tinham como principal infra-estrutura física as redes locais ou a

grande rede que é a Internet as quais eram quase em toda sua totalidade formadas por sistemas cabeados. As redes cabeadas possuem uma particularidade que é a transmissão de dados através de um meio confinado.

Em contrapartida, as redes locais sem fio tomaram um grande vulto a partir de sua disseminação comercial em 2000. Não que este meio de comunicação tenha surgido somente neste ano, outros projetos pioneiros como o ALOHA (TANENBAUM, 2002), já utilizavam este tipo de rede como alternativa de transmissão desde 1970 na AlohaNet instalada na Universidade do Havaí (AlohaNet, 2004). Ressalta-se no entanto que os estudos e aplicações das redes locais sem fio estavam baseados na operacionalização deste tipo de rede, com pouco foco na segurança da informação.

Grandes esforços acadêmicos e industriais foram concentrados no sentido de tornar as redes locais sem fio mais confiáveis e com maior capacidade de transmissão, como pode ser visto através do grande número de protocolos que foram estudados e implementados como o ALOHA, S-ALOHA, CSMA/CA e MACAW (TANENBAUM, 2002).

Em um segundo momento, a partir do qual as redes sem fio passaram a contar com uma maior confiabilidade e taxas de transmissão da ordem de 11 Mbps, com propostas atuais beirando os 100 Mbps, houve a adoção comercial desta tecnologia, com milhares de equipamentos sendo vendidos anualmente (Veja, 2004).

Pode-se considerar a operacionalização das redes sem fio como um desafio vencido, mas outros requisitos surgiram com alto grau de criticidade dentre eles QoS e segurança.

Esta dissertação aborda os aspectos de segurança das redes sem fio, em especial aspectos relacionados ao procedimento de autenticação e troca de chaves entre dois nós[1] das redes sem fio infra-estruturadas[2].

[1]Nó corresponde a uma entidade participante da rede sem fio com capacidade de recepção e transmissão de dados através de ondas eletromagnéticas e que juntamente com outras entidades utiliza um protocolo de comunicação comum aos demais.

[2]As redes sem fio podem ser classificadas inicialmente em dois tipos, redes ad hoc e redes infra-estruturadas como será visto adiante.

Diversas falhas de segurança em redes sem fio foram reportadas, como por exemplo as encontradas no WEP (WALKER, 2000), (BORISOV, 2001), (FLUHRER, 2001) e (STUBBEEFIELD, 2001), falhas no 802.1X (MISHRA, 2003), no WPA (MOSKOVITZ, 2004) e (Truesecure, 2004), entre outras. Todas estas falhas serão abordadas em capítulos posteriores, mas neste momento este grande número de propostas e suas sucessivas falhas de segurança caracterizam o grande desafio de se prover segurança para as redes locais sem fio.

Estas falhas abalam os aspectos de confidencialidade, integridade e disponibilidade de uma infra-estrutura baseada em rede sem fio, apesar de diversas propostas já terem sido elaboradas, como serão vistas adiante.

Estas falhas e os desafios ainda não respondidos apresentam-se como as principais motivações para o desenvolvimento desta dissertação. Neste trabalho abordam-se protocolos de autenticação, troca de chaves e detecção de desligamento de estações (DDE), que serão responsáveis por garantir:

- A autenticação dos nós na rede sem fio;
- A provisão de uma chave de sessão que poderá ser utilizada em um processo de criptografia dos dados que trafegam na rede ;
- Detecção automática do desligamento de uma estação, servindo como mecanismo para revogação de acesso.

Estes tipos de protocolos não são específicos para redes sem fio, sendo amplamente aplicados nas redes cabeadas. Na verdade, a tendência tem se mostrado oposta, ou seja, muitos protocolos de segurança aplicados há anos nas redes cabeadas têm encontrado um terreno fértil de aplicação nas redes sem fio.

Não obstante, a aplicação destes protocolos inicialmente voltados para redes cabeadas em redes sem fio sem uma análise inicial, pode levar a inconsistências. Um exemplo de uma inconsistência que possibilita exploração de falhas de segurança é aparente na aplicação do protocolo 802.1X nas redes 802.11 como será visto mais adiante.

Em contrapartida muitas das falhas de desenho do WEP são apenas repetições de falhas já corrigidas quando dá proposta e implementação do conjunto de protocolos que forma o IPSec (IPSec, 2004).

Para tanto serão analisados protocolos de autenticação, troca de chaves e DDE existentes não de forma exaustiva, mas fazendo-se um apanhado dos protocolos mais utilizados comercialmente, que são objeto de estudos acadêmicos e que representem classes distintas, como serão vistos adiante.

Esta análise será dividida em descrever sucintamente as principais características destes protocolos, apontar pontos fortes e fracos e desta forma categorizá-los de maneira que se verifique a significância de cada um quanto a aplicabilidade nas redes locais sem fio.

Através desta análise serão definidas métricas para comparação destes protocolos, de forma que alguns destes sejam implementados para posteriormente serem examinados em detalhes, culminando com um discussão comparativa contendo os principais protocolos de autenticação e troca de chaves.

Em resumo, um procedimento de segurança para uma WLAN conta com:

1. Autenticação mútua dos nós;
2. Troca de chaves de sessão que serão utilizadas nos passos 4 e 5;
3. Permissão de acesso aos recursos oferecidos por um nó da rede (Controle de acesso);
4. Confidencialidade dos dados que trafegam entre dois nós;
5. Integridade dos dados que trafegam entre dois nós;
6. DDE (Controle de acesso)

O processo de autorização e bloqueio (passos 3 e 6) correspondem ao controle de acesso, mas não se deve simplificar o controle de acesso como apenas um método de liberar e revogar o acesso irrestrito aos recursos oferecidos por uma determinada entidade.

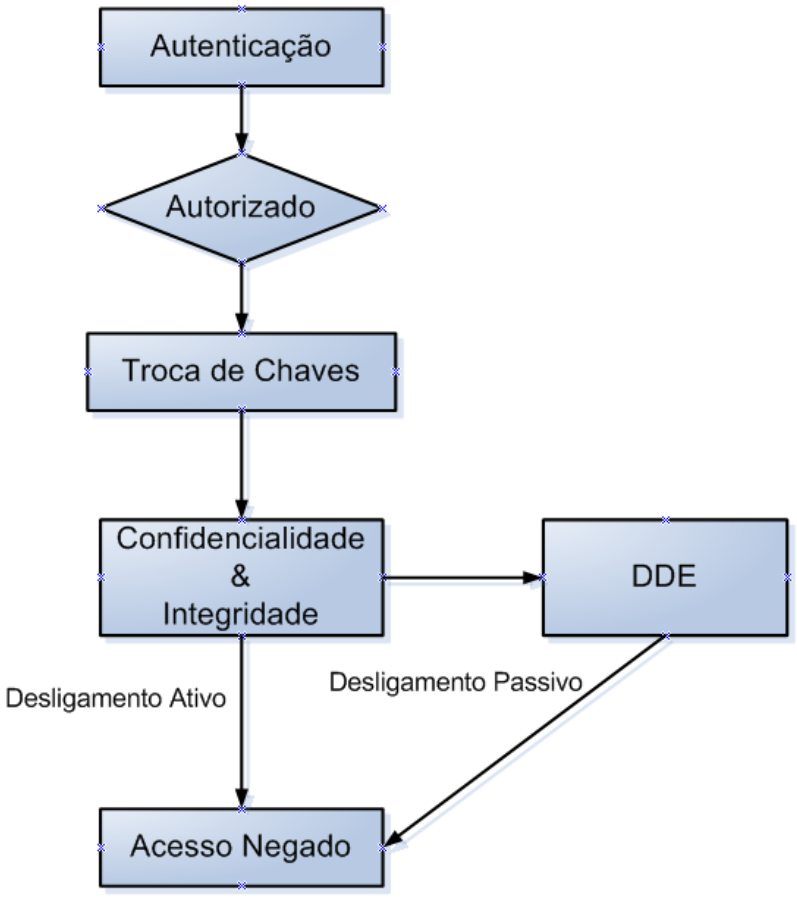

Figura 1.1: Fluxograma de segurança em redes

O controle de acesso pode ser granularizado de forma que baseado em um conjunto de informações como credenciais, horário do dia e quotas, por exemplo, ele permita que se habilitem determinadas categorias de acesso na entidade que provê os recursos de rede.

Um exemplo simples e retirado do ambiente de redes sem fio é liberar o acesso a serviços HTTP (BERNERS-LEE, 1996) para usuários que sejam do grupo CONVIDADO e acesso SSH (OpenSSH, 2004) para usuários que sejam do grupo ADMINISTRADORES.

O importante é destacar que perfis de acesso podem ser estabelecidos, como neste exemplo em que dois grupos distintos têm acesso a serviços distintos.

Nem todos os protocolos implementam o conjunto Autenticação, Troca de Chaves, Controle de Acesso, Confidencialidade, Integridade e Detecção de Desligamento

de Estação (ATCCID), sendo mais comum observar sistemas que provêm ATCCIDD através de vários protocolos trabalhando em conjunto.

Isto não ocorre no WEP que implementa Autenticação, Controle de Acesso, Confidencialidade e Integridade, enquanto que o WPA implementa somente Confidencialidade e Integridade.

Esta dissertação abrange protocolos de autenticação, troca de chaves e detecção de desligamento de estação e contribui com:

- Definição dos termos utilizados para se estabelecer o nível de representatividade dos protocolos para as WLANs;
- Definição de um grupo de protocolos que sejam representativos nas WLANs;
- Análise e implementação dos protocolos considerados representativos;
- Definição de métricas para avaliação dos protocolos de autenticação e troca de chaves em redes sem fio;
- Proposta de novos protocolos de autenticação e troca de chaves para WLANs, baseados na análise dos protocolos já existentes propondo melhorias e inovações;
- Avaliação e proposição de mecanismo de detecção de desligamento de estação;
- Análise comparativa dos protocolos considerados representativos e dos protocolos propostos.

Capítulo 2

Protocolos de rede sem fio

2.1 Introdução às Redes sem Fio

AS redes sem fio caracterizam-se pela transmissão de dados através de ondas eletromagnéticas em um meio intrinsecamente não confinado.

Os protocolos mais utilizados últimos anos incluem:

1. **Bluetooth** (HAARTESEN, 2000): Dedicado a redes locais pessoais com alcance de até 10 m, tem como principal objetivo substituir os cabos para comunicação entre equipamentos. O Bluetooth possui taxa de transmissão de até 1 Mbps. Um exemplo de uso do Bluetooth é a conexão de um celular a um *headset*;

2. **HomeRF** (MYERS, 2004): Utilizado para a comunicação em redes locais domésticas a fim de interconectar eletrodomésticos em uma rede local. As taxas de transmissão são de 1.6 Mbps. Um exemplo do HomeRF é a interconexão de um Modem-Router-ADSL a uma TV, um computador e impressora através de uma rede sem fio. O grupo de desenvolvimento do HomeRF (*Home Radio Frequency Working Group - HomeRF WG*) foi dissolvido em Janeiro de 2003;

3. **Hiperlan** (ETSI, 2004): Hiperlan significa *High Performance Radio Local Area Network* e corresponde a um padrão europeu para comunicação de dados

em redes sem fio com taxas de transmissão de até 54 Mbps, mas que ainda não encontrou penetração no mercado. Este padrão é promovido pela ETSI BRAN, *European Telecommunications Standards Institute Broadband Radio Access Networks*;

4. **802.11** (802.11, 1999): Padrão desenvolvido pelo IEEE baseado no CSMA/CA (*Carrier Sense Medium Access with Colision Avoidance*) (TANENBAUM, 2002) com taxas de transmissão atuais que variam de 1 Mbps a 54 Mbps e dedicado às redes locais sem fio.

Destes protocolos o 802.11 é o que possui o maior número de referências acadêmicas e sucesso no mercado. Este padrão foi desenvolvido com dois objetivos principais, as redes locais sem fio infra-estruturadas e as redes ponto-a-ponto (redes ad hoc). O primeiro tipo é o de maior sucesso e uso atualmente.

Devido à grande disseminação deste protocolo e dos desafios em aberto, adotou-se o 802.11como foco principal de aplicação dos mecanismos de autenticação e troca de chaves avaliados e propostos nesta dissertação.

O 802.11 é subdivido em diversos grupos tarefa e padrões:

1. **802.11a**: Padrão de comunicação de 54 Mbps que utiliza a faixa ISM[1] de 5 GHz;
2. **802.11b**: Padrão de comunicação de 11 Mbps, utilizando a faixa ISM de 2.4 GHz. É o padrão mais utilizado atualmente;
3. **802.11d**: Regulamentação de banda em novos países (2.4 GHz);
4. **802.11e**: QoS;
5. **802.11f**: Inter-Access Point Protocol (IAPP);
6. **802.11g**: Padrão de comunicação de 54 Mbps, utilizando a faixa ISM de 2.4 GHz. Padrão em ascensão;

[1]ISM, (Industrial, Scientific and Medical), se refere a faixas de frequência que podem ser utilizadas sem que seja necessária a aprovação prévia ou concessão do governo, desde que sejam obedecidas algumas regulamentações como potência máxima de transmissão (ITU, 2005)

7. **802.11h**: Compatibilidade de espectro de 5 GHz com países da Europa;
8. **802.11i**: Melhorias de segurança como por exemplo o WPA que será visto adiante;
9. **802.11j**: Compatibilidade de faixas de frequência com o Japão;
10. **802.11n**: Padrão de comunicação em desenvolvimento com previsão de taxas de comunicação de 100 Mbps, utilizando a faixa ISM de 5 GHz;
11. **802.11p**: WAVE - *Wireless Ability in Vehicular Environments*. Wi-Fi aplicados a ambulâncias e carros de passageiros;
12. **802.11r**: Fast roaming (Roaming em menos de 50 ms);
13. **802.11s**: Interligação em malha de redes sem fio;
14. **802.11T**: Métodos de teste e métricas;
15. **802.11u**: Interligação com outros padrões de redes sem fio;
16. **802.11v**: Gerenciamento de redes sem fio.

O IEEE definiu como protocolo de segurança para o padrão 802.11 o WEP (*Wired Equivalent Privacy*) (802.11, 1999). Este protocolo não é mandatório e possui uma série de falhas reportadas, com ferramentas automatizadas de forma a executar ataques. Estes ataques e ferramentas serão detalhados mais adiante.

Uma rede sem fio infra-estruturada baseada no padrão 802.11 utiliza uma célula básica de comunicação denominada *Basic Service Set* (BSS). Esta célula básica conta com um ponto de acesso e com as estações de redes sem fio.

Este tipo de rede é denominada rede infra-estruturada, pois depende de uma infra-estrutura prévia para seu funcionamento. Neste caso, o ponto de acesso e possivelmente as definições de domínio de rede (ESSID) e chaves de acesso no caso de WEP estar habilitado, por exemplo.

Todo o processo de comunicação em uma rede infra-estruturada é feita através do ponto de acesso que a princípio conta com todas as definições de uma estação

compatível com o 802.11 além de características particulares para lidar como por exemplo a associação e autenticação dos estações da BSS.

Os nós, ou seja, as estações somente se comunicam através do ponto de acesso, de forma semelhante como se comunicariam com um *switch*. Cada célula possui um identificador denominado de SSID (*Service Set Identifier*) que corresponde ao endereço MAC do ponto de acesso, desta forma cada célula pode ser unicamente identificada, pois o endereço MAC é único no mundo[2].

Cada célula utiliza uma banda do espectro de freqüências disponível, ou seja, um subconjunto das freqüências disponíveis na banda ISM de 2.4 GHz. No 802.11b, foco principal deste trabalho, existem 11 canais, ou seja, 11 bandas de freqüências.

O canal 1 está centralizado na freqüência de 2412 MHz, mas ocupando uma banda de 20 MHz que vai de 2402 MHz a 2422 MHz, em contrapartida o canal 11 está centralizado na freqüência de 2462 MHz, ocupando a faixa de 2452 MHz até 2472 MHz, como pode ser visto na figura 2.1.

Cada canal é separado por uma faixa de 5 MHz, portanto o canal 1 interfere diretamente com o canal 2 e 3. Duas células podem, ou seja, duas BSSs podem ocupar o mesmo espaço físico e utilizar o mesmo canal, pois o SSID é diferente, permitindo que as estações reconheçam a sua BSS através do SSID, no entanto a taxa de transmissão cai expressivamente dada a interferência entre canais.

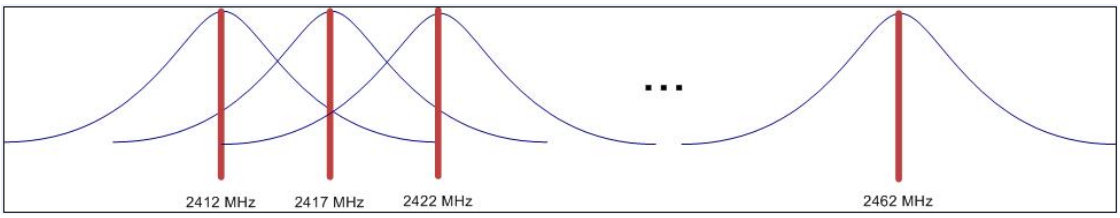

Figura 2.1: Canais utilizados no padrão 802.11b

[2]O endereço MAC é único dado que o IEEE distribui unicamente estes endereços através dos desenvolvedores de placas de rede ou dispositivos contendo interfaces de rede compatíveis com o padrão IEEE 802.3 como por exemplo *switches* e roteadores. No entanto, existem ataques que permitem que o endereço MAC de uma determinada interface de rede seja modificado, portanto passando a existir uma duplicação de um dado endereço. Este ataque é denominado de MAC *Spoofing*.

A conexão de BSSs a uma infra-estrutura comum de acesso é denominada ESS (*Extended Service Set*) e são definidas por um domínio (ESSID *Extended Service Set Identifier*).

Através do ESSID é possível aumentar a área de cobertura de uma rede sem fio de uma dada organização através da adoção de um maior número de pontos de acesso distribuídos geograficamente e que compartilhem o mesmo ESSID.

Uma estação que se distancie de um ponto de acesso e se aproxime de um outro com o mesmo ESSID poderá efetuar o *hand off*[3] na camada 2 do modelo OSI, ou seja, os mecanismos de associação/autenticação na camada 2 poderão ser efetuados de forma transparente[4].

No entanto, não há garantias de manutenção de comunicação das camadas superiores do modelo OSI, para tanto outros mecanismos devem ser propostos como visto em (MobileIP, 2004), por exemplo.

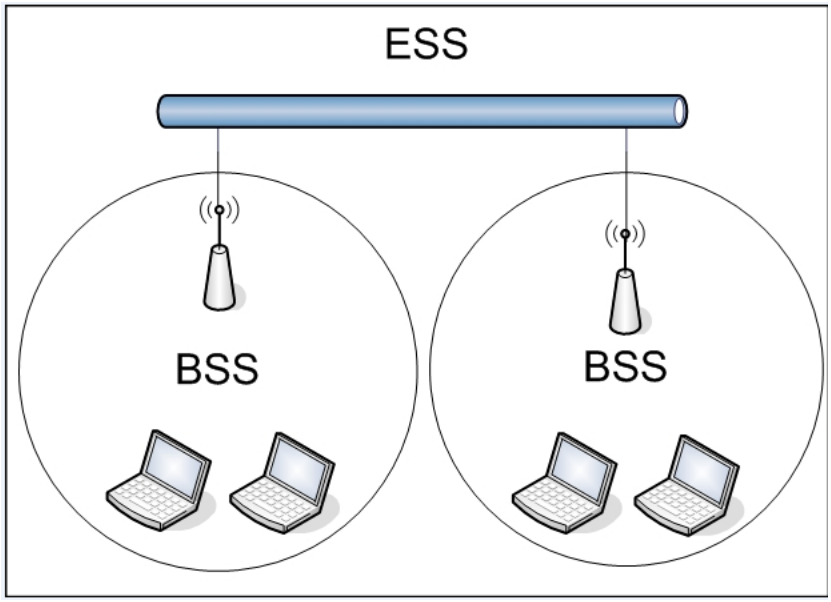

Figura 2.2: Diagrama de uma rede 802.11 representando duas BSSs e uma ESS

[3]Hand Off ou Hand Over são termos que se referem a transição de uma célula (BSS) para outra, desta forma a associação com um ponto de acesso é desfeita ao passo que outra é estabelecida com um ponto de acesso diferente. Este procedimento se aplica a redes celulares onde há transição de uma ERB (Estação Rádio Base) para outra

[4]O *soft hand over* se caracteriza pela continuidade do serviço de forma imperceptível pelo usuário (transparente), enquanto que no *hard hand over* há interrupção perceptível do serviço.

2.2 Falhas de segurança no 802.11

Como apresentado anteriormente o WEP é o protocolo padrão de segurança das redes 802.11 que tem como objetivo garantir confidencialidade, autenticação e integridade nas comunicações. Como visto na figura 2.3, o protocolo é baseado na utilização de uma chave pré-compartilhada entre as estações e o ponto de acesso. Ressalta-se que a mesma chave é de conhecimento de todas as estações e pontos de acesso pertencentes ao mesmo ESS.

A chave pré-compartilhada é concatenada a um vetor de inicialização (IV *Initialization Vector*) e este conjunto alimenta um gerador de números pseudo-aleatórios (PRNG *Pseudo Random Number Generator*) baseado no algoritmo RC4 (SCHNEIER, 1996). Este PRNG será responsável por gerar um fluxo de cifra do mesmo tamanho que o texto a ser cifrado.

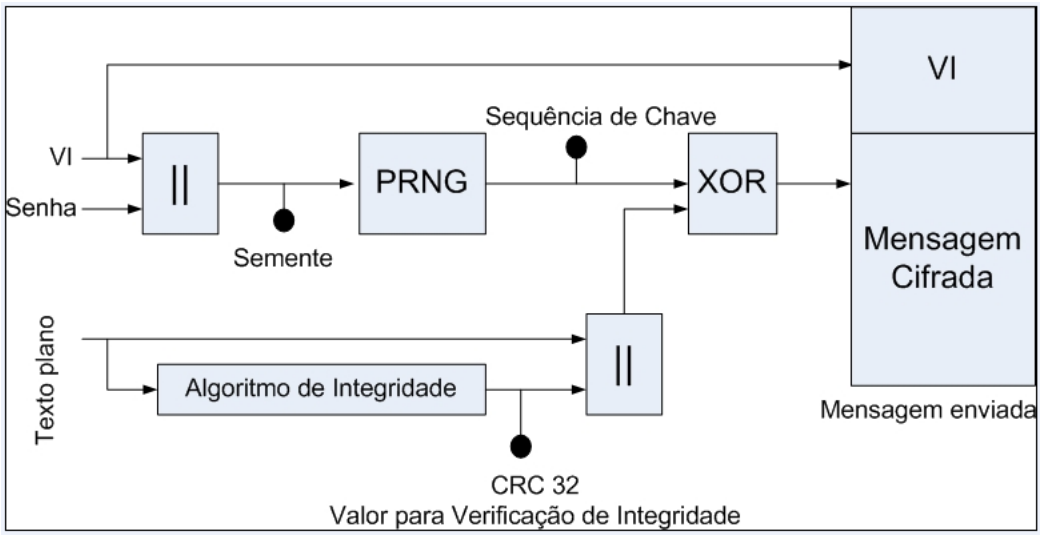

Figura 2.3: Diagrama em blocos do WEP

Com isto o texto cifrado sofrerá uma operação de "ou exclusivo"com o fluxo de cifra, sendo o resultado concatenado ao IV e enviado para o outro nó. Observa-se que o texto a ser cifrado é a concatenação dos dados a serem transmitidos com um código corretor de erros baseado no CRC 32.

O outro nó recebe a cifra e efetua a operação reversa a fim de descobrir os dados

em texto claro[5] e verifica a integridade dos dados através do algoritmo CRC 32, como apontados a seguir.

A chave pré-compartilhada possui 40 bits (WEP 1) ou 104 bits (WEP 2) e o vetor de inicialização possui 24 bits. Chaves de 40 bits estão sujeitas a ataques de força bruta (SCHNEIER, 1996) e o reduzido espaço de 24 bits para o vetor de inicialização possibilitará diversos ataques.

Uma série de falhas foi reportada que implicam desde falhas de projeto do protocolo, falhas de implementação e falhas operacionais. Um sumário destas falhas pode ser visto em (ARBAUGH, 2001), (CASOLE, 2002) e (WELCH, 2003) e dentre elas destacam-se:

1. **Inexistência de um mecanismo de distribuição de chaves**: Não existe um mecanismo de distribuição de chaves definido. Com isto todas as estações compartilham a mesma chave com o ponto de acesso, comprometendo a confidencialidade da mesma.

2. **Falhas na utilização do RC4 (FLUHRER, 2001)**: O RC4 foi utilizado inadequadamente no WEP, permitindo que a chave utilizada seja recuperada após a captura de um determinado número de pacotes. Em 2001 este ataque teórico foi implementado por (STUBBLEFIELD, 2001). Foi disponibilizado em 2001 uma ferramenta de código aberto e gratuita que executa o ataque proposto por (FLUHRER, 2001), denominada de AirSnort. Outra ferramentas já estão disponíveis que diminuem o tempo de ataque de horas para apenas alguns minutos ou segundos como em (WEPCrack, 2005).

3. **Decodificação de cifra sem a chave**: Em (BORISOV, 2001) e (WALKER, 2000) são descritos ataques que permitem a recuperação de um texto claro através da captura de cifras que utilizam o mesmo IV e chave pré-compartilhada. Em (PETRONI, 2003) este ataque é elaborado e permite a recuperação *online* dos fluxos de cifra.

[5]O termo texto claro se refere a uma mensagem que não foi criptografada, ou seja, mensagem original conforme descrita pela origem.

4. **Injeção de mensagens e quebra de integridade**: Em (BORISOV, 2001) e (WALKER, 2000) são exemplificados ataques que permitem a injeção de mensagens ou modificação de mensagens válidas através da manipulação de determinados campos da mensagem cifrada. Esta quebra de integridade é possível dado que o código verificador de erros é linear (CRC 32), possibilitando modificações simples e compensatórias em determinados bits da mensagem cifrada, mantendo-se a integridade da mesma.

5. **Forjar autenticação**: (BORISOV, 2001) apresenta um ataque que possibilita uma estação forjar o processo de autenticação através da captura de mensagens de autenticação válidas, a recuperação da cifra de fluxo e a reutilização da mesma em um novo processo de autenticação (Ataque *Replay*).

6. **Redirecionamento IP**: (BORISOV, 2001) demonstra uma variação do ataque de modificação de mensagens que permite a um atacante redirecionar mensagens para estações sob seu controle.

Como pode ser visto o WEP, cujo objetivo inicial era de prover um nível de segurança compatível com o confinamento provido por uma meio cabeado, não foi eficaz em assegurar a confidencialidade, integridade e disponibilidade das redes sem fio baseadas no protocolo 802.11.

Pode-se afirmar que a invasão física de um ambiente e o posicionamento de uma escuta, seja ela na camada física ou na camada de enlace, são tarefas mais complexas e difíceis do que acessar um ponto de acesso a cerca de 1 km e utilizar alguma ferramenta difundida e gratuita para quebra da criptografia.

Apesar das falhas do WEP muitas corporações utilizam os pontos de acesso sem nenhum tipo de proteção (MOHNEY, 2003), permitindo que qualquer usuário mal intencionado e sem habilidades técnicas possa usufruir dos recursos de redes de terceiros, podendo haver roubo de informação, indisponibilidade de acesso, quebra de integridade entre outros ataques. O WEP serve o propósito de pelo menos afastar usuários com parcas habilidades técnicas.

Devido às diversas falhas apresentadas de forma resumida nesta seção pesquisa-

dores e empresas iniciaram um esforço no sentido de suprir a necessidade de implantação de redes sem fio seguras.

2.3 Propostas existentes de segurança para redes 802.11

Muitas propostas surgiram desde que as falhas do protocolo WEP foram rapidamente reportadas e exploradas. Algumas destas propostas abordam aspectos de topologia e interconectividade, enquanto outras elaboram protocolos que garantam um ou todos os pilares da segurança da informação: integridade, disponibilidade e confidencialidade.

O Apêndice B fornece uma introdução sobre os conceitos básicos de segurança da informação para protocolos de comunicação de redes.

2.3.1 Melhores práticas

Há um conjunto de controles que devem ser aplicados à rede sem fio de forma que a segurança desta rede e da rede cabeada a que ela normalmente se conecta sejam elevadas. Uma lista não exaustiva conta com:

1. **Mudança do ESSID padrão dos pontos de acesso**: cada fabricante configura um valor padrão para o ESSID. Com isto pode-se conectar a um ponto de acesso a partir de algumas tentativas, através de ataques de força bruta[6];

2. **Topologia**: Posicionamento da infra-estrutura da WLAN sob a proteção de um *firewall* e em uma DMZ específica para interconexão com a infra-estrutura de rede cabeada;

[6]Alguns pontos de acesso têm a capacidade de trabalhar de duas formas distintas, sendo a mais comum conhecida como Open System, onde o ESSID é enviado em intervalos regulares em mensagens de broadcast e o Closed System onde esta identificação não é enviada, obrigando cada estação que tenha este valor pré-configurado. Isto mitiga o risco de associação de estações maliciosas, mas qualquer atacante que capture pacotes de associação válidos será capaz de aprender o ESSID.

3. **Mudança do canal padrão**: Os pontos de acesso vêm configurados em canais padrão, possibilitando a identificação do modelo/fabricante por atacantes, além de aumentar a probabilidade de ter vários pontos de acesso próximos interferindo na mesma faixa de freqüência;

4. **Mudança da senha de gerenciamento**: muitos pontos de acesso possuem interfaces de gerenciamento via *web* ou porta serial e o acesso é protegido por uma senha padrão;

5. **Habilitação do WEP**: Apesar das falhas, a habilitação do WEP oferece um nível de proteção contra ataques efetuados por usuários com parcas habilidades técnicas[7];

6. **Utilizar chaves de difícil dedução**: as chaves WEP devem ser escolhidas de forma que não façam parte de dicionários ou sejam de fácil dedução. Chaves WEP de 40 bits permitem que 2^{40} chaves sejam escolhidas, no entanto senhas de fácil dedução utilizam um pequeno subconjunto destas possibilidades. Por exemplo, uma senha de 5 octetos (5 octetos x 8 bits = 40 bits) que seja formada somente por letras minúsculas (27 letras) corresponde a apenas 0.0013% do total das senhas possíveis.

7. **Posicionamento das antenas**: As antenas devem ser posicionados de forma que a figura de radiação privilegie o espaço físico de uma dada corporação, mitigando a possibilidade de acesso de usuários vizinhos não autorizados.

Este *checklist* apresenta um conjunto de melhores práticas para a segurança de redes sem fio, mas não mitiga todos os riscos, como os apresentados pelas falhas do WEP, por exemplo.

Algumas propostas atem-se a utilização de *firewall* para segregação de tráfego da rede sem fio (MURTHY, 1998), outras soluções (GOGBER, 2002) e (TURKULAINEN, 2004), utilizam VPN segura baseada no protocolo IPSec (IPSec, 2004) de forma que o processo de autenticação dos nós seja feita através da própria formação

[7]Quatro chaves poderão ser configuradas, sendo que uma delas é selecionada para cada sessão para cada associação estação e ponto de acesso

do túnel VPN e que todo o tráfego seja encriptado através deste túnel, garantindo-se desta forma, a integridade e autenticidade.

Outras soluções como baseadas em chaves públicas foram apresentadas por (AZIZ, 1994), que não se atém diretamente 802.11, mas a um ambiente sem fio genérico ou outras com aplicações diretas em redes 802.11 como em (PRASAD, 2000), que propõe mecanismos baseados no protocolo Kerberos (MILLER, 1987) ou RADIUS (RIGNEY, 2000) para armazenamento de chaves públicas.

Algumas destas soluções encontraram maior apoio acadêmico ou de mercado como a utilização do protocolo 802.1X (802.1X, 2001) de controle de acesso e autenticação, mas que também teve diversas falhas de segurança reportadas (ARBAUGH, 2002).

2.3.2 Portais de captura

Algumas propostas como (NoCatAuth, 2004), (NetLogon, 2004) e (OASIS, 2004) propõem a criação de um controle de acesso baseado em autenticação via interface HTTP/HTTPS, configuração dinâmica de regras de *firewall* e aplicação de um controle detecção de desligamento de estação.

Qualquer usuário da rede sem fio não terá acesso aos recursos de rede até que ele se autentique em uma página HTTP/HTTPS contida no ponto de acesso. As credenciais serão validadas em um banco de dados e o ponto de acesso apresentará suas credenciais para o usuário através de um certificado digital (este processo é chamado de autenticação mútua).

As regras de firewall contidas no ponto de acesso serão modificadas de forma a prover acesso ao usuário que se autenticou com sucesso. Em seguida um sistema de detecção de desligamento de estação monitorará a estação até o momento em que ela se desconecte da rede (desligamento da máquina[8], usuário está fora do alcance do rádio do ponto de acesso etc.) e assim modificará as regras de firewall para bloquear

[8]O desligamento pode se passivo ou ativo, o que diferencia um do outro é o pedido explícito de desconexão por parte do usuário

o acesso do dado usuário.

Estas propostas estão baseadas no protocolo SSL/TLS e necessitam de um ponto de acesso com um servidor WEB com suporte a SSL/TLS, um banco de dados para verificação das credenciais e armazenamento de *logs* e uma infra-estrutura de chaves públicas (PKI) a fim de que os pontos de acesso tenham certificados assinados por uma autoridade certificadora em que os usuários da rede confiem.

Será feita uma análise mais detalhada dos mecanismos de detecção de desligamento de estação no capítulo 6.

2.3.3 WEP (*Wired Equivalent Privacy*)

Como apresentado anteriormente o IEEE definiu como protocolo de segurança para o padrão 802.11 o WEP (*Wired Equivalent Privacy*) (802.11, 1999). O padrão não define o WEP como mandatório.

O procedimento de autenticação é apresentado abaixo:

A -> B : Requisição de Autenticação
B -> A : Desafio
A -> B : Resposta
B -> A : ACK/NACK

A envia para B uma requisição de autenticação, em seguida B envia para A um desafio, que consta de um número aleatório de 128 bits. A recebe este desafio e utiliza uma chave pré-compartilhada com B para encriptar este desafio e enviá-lo de volta para B (resposta).

Ao mesmo tempo em que A calcula a resposta, B calcula a resposta e tão logo receba a resposta calculada por A, as compara. Se forem iguais, B envia uma mensagem ACK para A indicando que o procedimento de autenticação foi finalizado com sucesso e habilita o acesso aos recursos de rede providos pelo ponto de acesso a A. Se forem diferentes, o procedimento de autenticação falha, indicando que A não possui a chave pré-compartilhada utilizada por B nos procedimentos de autenticação.

O padrão 802.11 indica que até quatro chaves podem ser utilizadas por ponto de acesso e estações. Alguns aspectos importantes devem ser ressaltados:

- O ponto de acesso não é autenticado
- As máquinas são autenticadas e não os usuários
- Não há derivação de uma chave de sessão
- Não há mecanismo de distribuição de chaves

Estas características facilitam uma séries de ataques ao protocolo e dificultam o gerenciamento da rede.

Como apresentado na figura 2.3 o WEP utiliza o RC4 e um vetor de inicialização de 24 bits, como chaves de 40 bits e 104 bits que em resumo levam aos ataques descritos anteriormente.

O protocolo WEP é considerado inadequado como mecanismo de segurança para redes 802.11, por conseguinte uma série de novos mecanismos foram e continuam a ser propostos, sendo os principais analisados neste capítulo.

2.3.4 802.1X

Este protocolo foi proposto tendo como foco as redes cabeadas e os riscos associados à possibilidade de uma estação não autorizada ser conectada a um *switch* e a partir deste momento ter acesso aos recursos de rede da mesma forma que uma estação autorizada.

Na verdade, não há uma definição formal presente no padrão 802.3 com relação a uma estação autorizada e não autorizada, dado que neste protocolo não há processos definidos de autenticação e associação. Isto também se aplica para o 802.11 no modo *Open System*, ou seja, quando o protocolo WEP não está habilitado.

Define-se estação não autorizada qualquer estação para a qual não há uma política implícita ou explícita que autorize sua conexão.

Por exemplo, para o caso de empresas que não autorizam a conexão de *notebooks* de terceiros em suas redes, considera-se o *notebook* uma estação não autorizada [9].

O 802.1X (802.1X, 2001) encontra vasta aplicação nas redes sem fio, pois a associação de estações não autorizadas fica potencializada. A figura 2.4 apresenta os principais componentes do 802.1X.

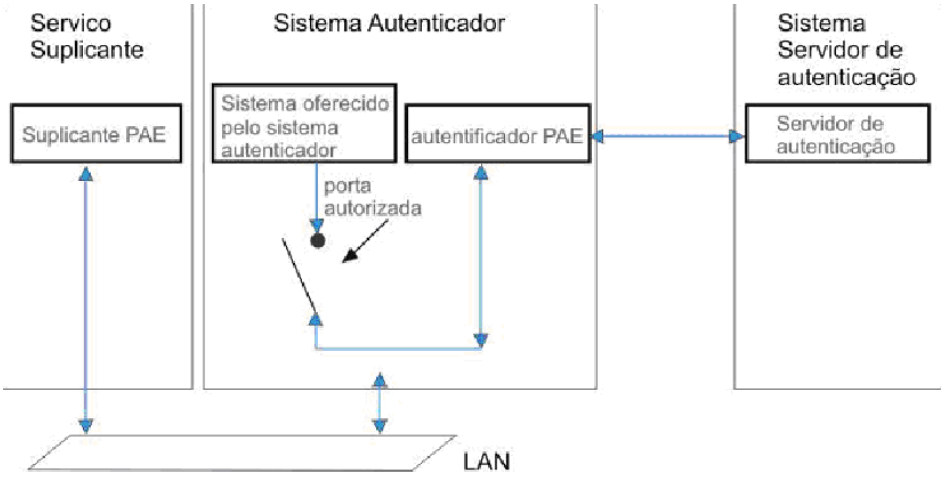

Figura 2.4: Entidades presentes no 802.1X

Na figura 2.4 estão presentes 3 entidades:

1. O suplicante: que representa a entidade que deseja se autenticar na rede e ter o acesso autorizado para uso dos recursos oferecidos através do sistema autenticador;

2. O sistema autenticador: que verifica o status do suplicante e quando adequado encaminha a requisição de autenticação para o servidor de autenticação verificar a validade das credenciais;

3. O sistema de autenticação: que é uma base de dados com as credenciais dos usuários e que contém os logs de utilização dos recursos da rede para fins de cobrança (*billing*), por exemplo.

[9]Interessante explorar este exemplo, pois não há mecanismos automáticos prevenindo a conexão de *notebooks* de terceiros, apenas uma política corporativa. Grande parte das corporações utilizam este modelo de livre associação na camada 2 em suas redes cabeadas, mas mitigam parte do risco com relação ao acesso à internet, impressoras e outros serviços definidos para o domínio. Um exemplo clássico é dado pelos domínios definidos pelo Active Directory da Microsoft

Um exemplo de uso pode ser visto em redes locais, onde um *desktop* está conectado a um *switch* e existe um servidor RADIUS (RIGNEY, 2000) presente na rede. O *desktop* representa o suplicante, o *switch* o sistema autenticador e o RADIUS o sistema de autenticação.

Em um primeiro momento o *switch* somente habilita a porta conectada ao *desktop* para operações de autenticação do protocolo 802.1X e que se destinem ao sistema de autenticação. Com a autenticação efetivada o acesso aos demais recursos de rede é autorizado. Em redes 802.11 o processo é semelhante, mas consideram-se as portas como associações lógicas.

No 802.11 o suplicante corresponde às estações sem fio, o autenticador aos pontos de acesso e servidor de autenticação a um servidor AAA (*Authentication, Authorization and Accounting*) como RADIUS ou TACACS+ (FINSETH, 1993).

O 802.1X é um padrão para controle de acesso e autenticação e pode utilizar diversos protocolos como o EAP-TLS (ABOBA, 1999) (*Extensible Authentication Protocol - Transport Layer Security*) e o EAP-MD5 (PHIFER, 2004) (Extensible Authentication Protocol - Message Digest 5), sendo o primeiro mais utilizado atualmente.

O EAP-TLS é baseado no encapsulamento do EAP em conexões ponto-a-ponto do protocolo TLS (DIERKS, 1999). O TLS é baseado em protocolos de chaves públicas, onde ambas[10] as entidades possuem certificados digitais assinados por uma terceira entidade em que confiam.

O MD5 se baseia em um processo de autenticação mais simples, onde o servidor de autenticação armazena o *hash* MD5 (RIVEST, 1992) da senha de cada usuário, ou seja, a senha do usuário passa por um processo criptográfico definido pelo algoritmo MD5, cujo resultado é armazenado e denominado *hash* da senha.

O processo de autenticação segue um modelo de desafio-resposta e utiliza o *hash* da senha como passo fundamental no processo de autenticação. Observa-se no en-

[10]O TLS obriga o servidor a se autenticar via certificado digital e deixa como opcional a autenticação do cliente, mas está sendo considerado a autenticação mútua que provê um maior nível de segurança.

tanto que a autenticação do servidor é feita de forma indireta, ou seja, o conhecimento do *hash* da senha do usuário serve como indício de que o servidor é confiável. Vale ressaltar que isto não garante de forma absoluta a autenticidade do servidor.

Autenticação baseada em criptografia simétrica e assimétrica possuem vantagens e desvantagens que serão exploradas mais adiante, a partir do momento em que sejam relatadas quais as principais falhas encontradas em sistemas de autenticação e troca de chaves, ou seja, a partir do momento em que serão definidos os parâmetros para comparação dos sistemas de autenticação e troca de chaves existentes e os propostos nesta dissertação.

2.3.5 WPA (*Wi-Fi Protected Access*)

O WPA (*Wi-Fi Protected Access*) (802.11i, 2004) é fruto de um esforço coordenado pelo IEEE através do grupo-tarefa 802.11i em resposta às falhas do WEP. O objetivo do WPA é prover mecanismos de autenticação, confidencialidade e integridade utilizando algoritmos mais robustos e uma arquitetura mais adequada.

O WPA utiliza o TKIP (*Temporal Key Integrity Protocol*) que gera chaves por pacote, um algoritmo de verificação de integridade denominado Michael não-linear, um vetor de inicialização de 48 bits, muito maior do que o utilizado pelo WEP (24 bits) e funcionalidades para troca da chave de sessão após um determinado tempo ou quantidade de bytes encriptados.

O WPA pode ser usado em conjunto com o 802.1X ou com chaves pré-compartilhadas. Através do 802.1X utiliza-se de uma base de dados RADIUS (RIGNEY, 2000) para captura de credenciais e efetuar um controle de acesso por usuário. A utilização de chaves pré-compartilhadas é mais simples, no entanto uma falha de implementação já foi reportada (Technewsworld, 2004). Uma falha no TKIP também foi reportada em (TrueSecure, 2004).

O WPA é um subconjunto da Força Tarefa do 802.11i e inovações importantes serão disponibilizadas ao longo de 2005 e 2006. No entanto, apesar de ser uma promessa quanto à segurança das redes sem fio, há um consenso entre os especialistas

de segurança com relação ao nível de segurança de um protocolo/tecnologia ser diretamente proporcional à maturidade de uma tecnologia (uma condição que tem se mostrado necessária, mas nem sempre suficiente).

2.3.6 Mensagens DHCP Autenticadas

O DHCP com mensagens autenticadas (DROMS, 2001) foi apresentado como uma alternativa segura para utilização do DHCP em redes cabeadas, encontrando uma grande aplicação nas redes sem fio.

Primeiramente é fundamental verificar o funcionamento do DHCP (DROMS, 1997) e isto pode ser brevemente apresentado através da figura 2.5:

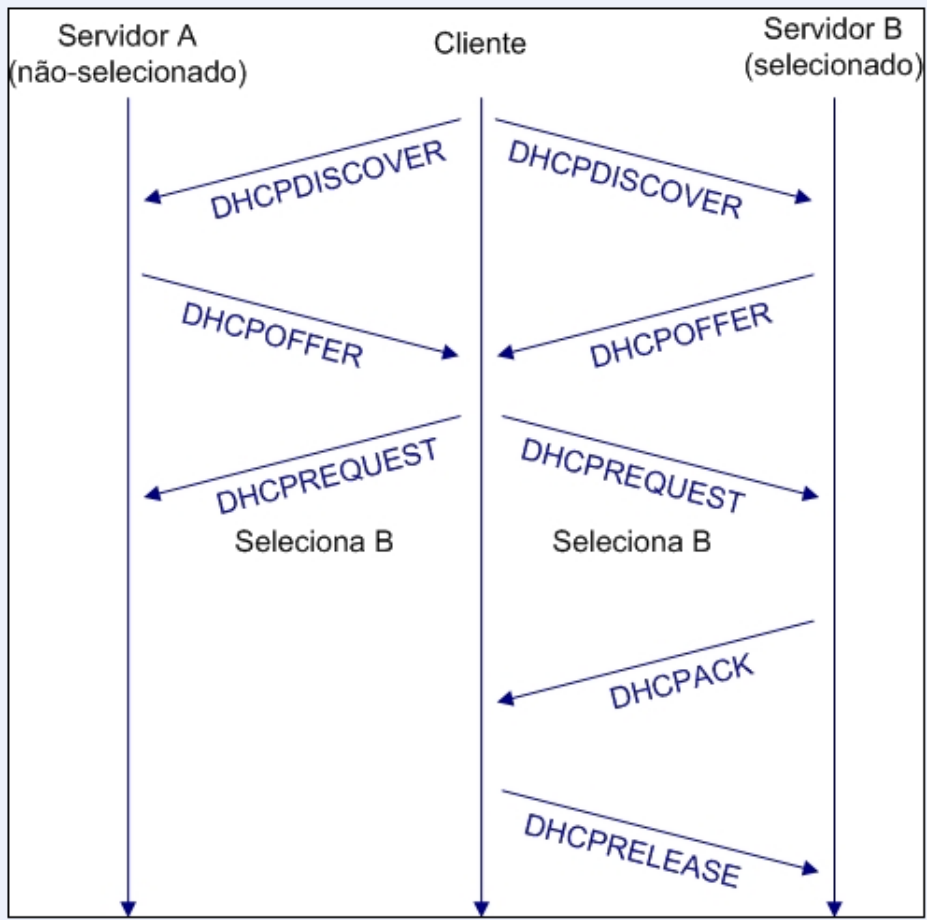

Figura 2.5: Troca típica de mensagens do protocolo DHCP

Segue abaixo uma breve narrativa sobre o funcionamento do protocolo.

O cliente envia uma mensagem DHCPDISCOVER para o endereço de *broadcast* local requisitando um endereço IP e outras informações de configuração de rede[11] a qualquer servidor DHCP presente na rede local. Todos os servidores presentes responderão a esta mensagem com uma mensagem DHCPOFFER que contém a sua oferta de endereço IP e outras configurações de rede para o cliente.

O cliente recebe todas as mensagens e responde com uma mensagem DHCPREQUEST somente a uma delas, diretamente ao endereço IP do servidor escolhido e implicitamente declinando as demais ofertas. Em seguida o servidor responderá a esta requisição com uma mensagem DHCPACK.

Cada servidor possui um conjunto limitado de endereços IP disponíveis e estes são atribuídos dinamicamente como brevemente relatado no mecanismo acima, portanto cada estação poderá utilizar estes endereços por um período pré-determinado chamado de *lease time* (tempo de empréstimo).

Algumas estações podem agir ativamente e avisar ao servidor DHCP que não necessitam mais do endereço IP atribuído (por exemplo, quando a estação está sendo desligada), mas nem sempre isto acontece (por exemplo, quando uma estação trava ou uma estação se afasta de seu ponto de acesso saindo de sua área de cobertura).

O servidor DHCP poderá reutilizar os IPs emprestados após o tempo de empréstimo ter se passado, quando a estação ativamente indica que não utilizará mais o IP ou quando uma estação faz uma requisição a um IP específico[12]. Uma verificação a mais é efetuada com o objetivo de garantir a não colisão de endereços IPs, ou seja, dois endereços MAC diferentes utilizando o mesmo endereço IP, através de mensagens ARP e ICMP *echo request* (ping).

A RFC prevê também situações em que o servidor DHCP é reiniciado ou o cliente é reiniciado e formas de tornar o processo de reafirmação do uso de um endereço IP mais eficiente. Este modelo é inseguro, pois não considera riscos de ações maliciosas e inserção de mensagens falsas.

[11] Domínio, *gateway*, máscara, endereço de *broadcast*, endereço do servidor de DNS, endereço do servidor WINS entre outras.

[12] Isto pode ocorrer quando há reinicialização dos sistemas tanto do cliente quanto do servidor. De acordo com a RFC 2131, o cliente deve tentar reutilizar o último endereço a ela atribuído.

Este processo de reafirmação baseia-se na consideração de que as configurações utilizadas antes da reinicialização continuam válidas e que qualquer validação destas premissas é efetuada via protocolos inseguros como ARP e ICMP vistos adiante.

Sem dar uma salto prematuro à proposta de segurança desta dissertação baseada no DHCP, apresenta-se a máquina de estados do servidor DHCP na figura 2.6.

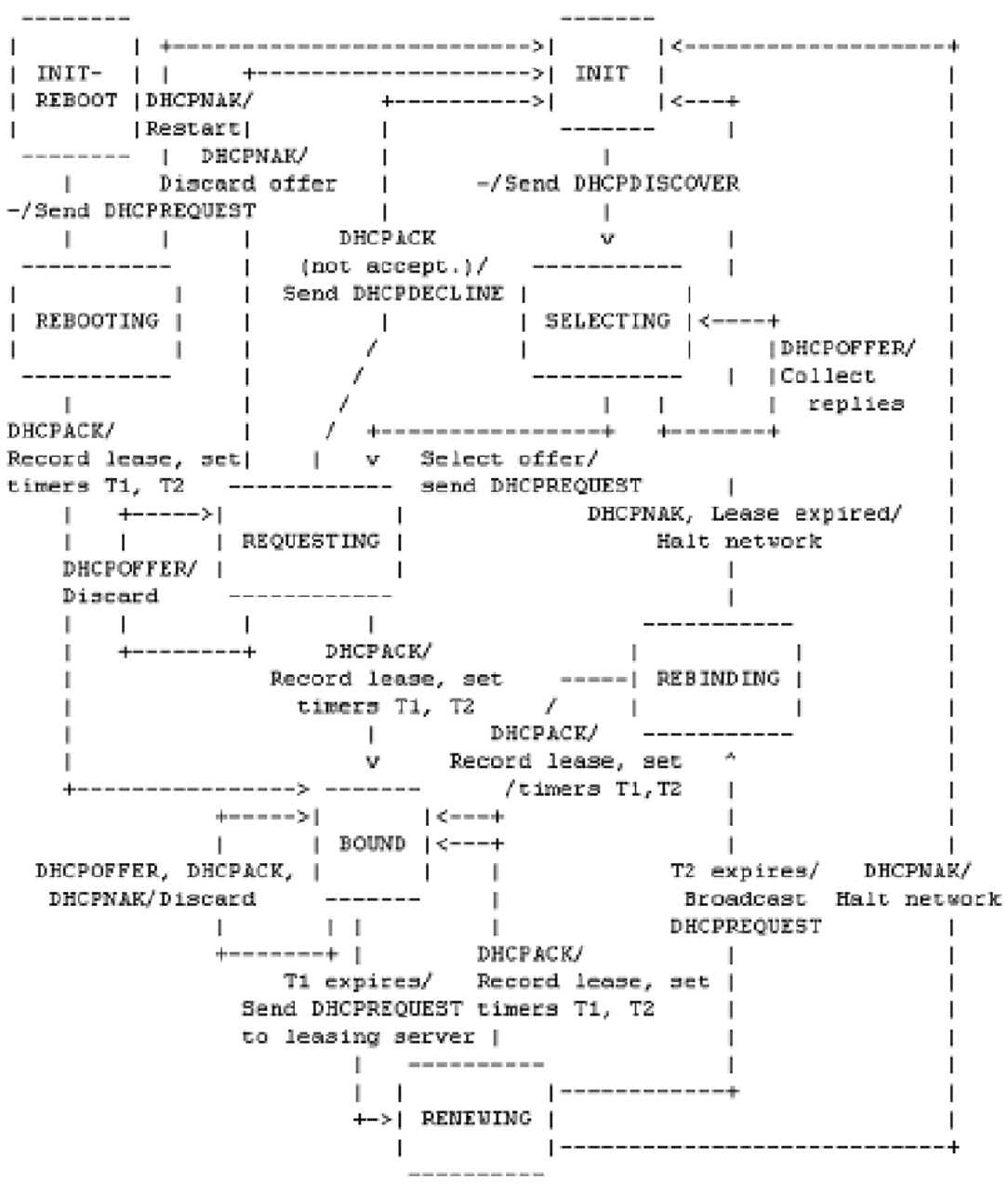

Figura 2.6: Máquina de estados do servidor DHCP (fonte (DROMS, 1997)

Dos 8 (oito) estados definidos pela RFC 2131, alguns destes (REBOOTING,

INIT-REBOOT, RENEWING, REBINDING) são utilizados quando da reinicialização do servidor DHCP e quando um cliente deseja estender o tempo de empréstimo de um endereço.

Estas mensagens estão intrinsecamente relacionadas a dois fatos fundamentais à arquitetura do protocolo:

1. Tentativa de manter a configuração das estações entre reinicializações do servidor DHCP.
2. Tempo de empréstimo ser dado por um tempo decorrido pré-determinado (*timeout*).

Fica evidente a ineficácia deste sistema através do fato (2), já que uma estação pode ser desligada muito antes do tempo de empréstimo ter terminado e mesmo que seja feita uma verificação ativa do uso deste endereço através de mensagens ICMP ou ARP, fica evidente uma falha de segurança já que as respostas a estas mensagens podem ser facilmente forjadas.

Mais deve ser dito quanto ao DHCP apresentado na RFC 2131: não há mecanismos de autenticação ou integridade presentes no protocolo, portanto vários ataques podem ser perpetrados, como:

1. **Utilização de um servidor DHCP falso**: Desta forma um atacante forja as mensagens DHCP passando parâmetros falsos de configuração da rede e efetuando ataques como *man-in-the-middle* ou redirecionamento a páginas WEB falsas (DNS *Spoofing*).
2. **Exaustão do conjunto de endereços IP disponíveis**: por parte de atacantes através de várias requisições de endereços IP, bastando ao atacante forjar endereços MAC para cada IP requisitado.

A RFC 3118 (DROMS, 2001) tenta mitigar os efeitos da falta de integridade e autenticação das mensagens no protocolo DHCP através da utilização de HASH

autenticado (HMAC) do conteúdo da mensagem DHCP, assinalando uma chave simétrica por estação ou por domínio, neste último caso uma PSK (*Pre-shared Key*).

Diz-se por estação, pois as mensagens trocadas entre cliente e servidor utilizam como identificador de usuários o endereço MAC da placa de rede de cada máquina como forma de identificar a senha utilizada ao longo do processo de autenticação.

Não existem implementações desta RFC como pode ser verificado no site do grupo de trabalho DHC (*Dynamic Host Configuration*) em (DHC, 2005).

De acordo com a pesquisa efetuada nesta dissertação, destacam-se as seguintes críticas quanto à RFC 3118:

- **Mensagens ARP e ICMP podem ser forjadas: DoS e baixo desempenho**

Mensagens ARP e ICMP podem ser facilmente forjadas por um atacante possibilitando um ataque de negação de serviço (DoS), por exemplo. Quando um cliente recebe um endereço IP de um servidor DHCP ele utiliza mensagens ARP/ICMP para verificar se nenhuma outra estação na rede está utilizando este endereço.

Um atacante pode forjar as respostas indicando para o cliente que o IP já fora previamente assinalado e desta forma forçando o cliente a reiniciar o processo de aquisição de um novo endereço.

O mesmo ataque pode ser aplicado no caso do servidor que efetua esta verificação de consistência utilizando mensagens ARP/ICMP e o ataque é bastante semelhante.

Outra característica fundamental é o baixo desempenho devido à utilização das mensagens ARP/ICMP ao longo do procedimento de autenticação e configuração para verificação de consistência do *pool* de endereços.

Estes ataques decorrem diretamente da utilização de um temporizador para o empréstimo de um endereço.

- **DHCPDISCOVER não é autenticado: DoS**

Possibilidade de negação de serviço na medida em que um usuário malicioso é capaz de consumir todos os endereços IP disponíveis no *pool* de endereços do servidor DHCP e desta forma impossibilitar um usuário autêntico de adquirir informações de configuração de rede, mesmo quando os mecanismos de segurança propostos pela RFC 3118 são aplicados.

A primeira mensagem DHCP não requer autenticação (mecanismo denominado *delayed authentication*) e portanto um usuário mal intencionado pode gerar uma série de requisições (DHCPDISCOVER) com endereços MAC forjados e o servidor reservaria um endereço do *pool* de endereços disponíveis para serem enviados nas mensagens DHCPOFFER. O impacto do ataque depende do *timeout* após o envio da mensagem DHCPOFFER de forma que o IP seja retornado ao *pool* de endereços disponíveis.

- **Necessidade e geração do XID: Baixo desempenho, DoS e Seqüestro de Sessão**

Todos os clientes utilizam a porta 68/UDP para receber mensagens e o servidor utiliza a porta 67/UDP para receber as mensagens. Além disto o cliente ainda não possui um endereço IP disponível e algumas mensagens necessariamente serão enviadas para o endereço de *broadcast* local. Para que as mensagens sejam multiplexadas de forma correta necessita-se de uma informação adicional a fim de tornar única uma sessão entre um cliente e um servidor. O mecanismo definido pela RFC 2131 é a utilização de um campo no *header* do protocolo DHCP denominado XID.

Sendo assim, o cliente gera uma mensagem com um determinado XID, que a princípio deve ser aleatório e não deve colidir com nenhum XID utilizado no mesmo momento na rede. O servidor recebe todos os pacotes destinados ao endereço 255.255.255.255:67, lê o campo XID de cada um deles e baseado neste XID verifica a qual sessão esta mensagem pertence. Através da descoberta da sessão verifica-se o *status* da sessão que corresponde a um dos estados da máquina de estados apresentada acima.

Dois pontos importantes devem ser analisados neste momento. Um de caráter de

segurança, pois (DROMS, 1997) propõe um mecanismo inadequado de geração do XID, tornando previsível o uso deste valor por parte de um atacante, possibilitando a inserção de mensagens falsas com o XID correto, bastando capturar o primeiro pacote válido enviado pelo cliente. O segundo problema se relaciona com a perda de desempenho por parte do cliente e do servidor.

O cliente necessariamente utiliza mecanismo de captura de quadros *ethernet* e através de um filtro verifica as mensagens destinas a ele. Este filtro pode ser descrito da seguinte forma: todos os datagramas IP, com endereço de destino 255.255.255.255 e com porta de destino 68/UDP e com porta de origem 67/UDP.

Muitas das mensagens conterão informações relacionadas a outros clientes e logo a outras sessões. O cliente verificará isto através da comparação do valor do XID recebido com o XID inicialmente gerado por ele. O servidor efetuará o mesmo procedimento, utilizando um filtro um pouco diferente, e em seguida consultará um tabela de estados para aquela sessão determinada pelo XID informado.

- **Tempo de *lease* dado por *timeout*: Seqüestro de sessão e uso ineficiente do *pool***

Como observado anteriormente, o tempo de empréstimo do endereço IP é dados por um *timeout*, gerando uma ineficiência na utilização do *pool* de endereços e uma janela de vulnerabilidade dada pela relação abaixo:

$$\text{Janela Vulnerabilidade (JV)} = tempo_{lease} \text{ - } tempo_{usoefetivo}$$

1. **Para JV $>$ 0 :** Possibilidade de roubo de sessão.
2. **Para JV $=$ 0 :** Utilização acordada dos recursos de rede por parte do usuário.
3. **Para JV $<$ 0 :** Usuário utiliza recursos da rede, mesmo após o tempo de *lease* ter se esgotado.

Em um grande número de casos haveria um tempo de uso inferior ao tempo de empréstimo e o controle de acesso apresentaria uma falha grande, permitindo o uso

indevido de recursos da rede através de IP *Spoofing* de um usuário válido e até um seqüestro ativo de uma sessão (Caso 1 acima).

Como forma de mitigar o uso ineficiente do *pool* de endereços e verificar se um destes ainda está em uso utilizam-se mensagens ARP/ICMP, cujas respostas podem ser forjadas.

O Caso 2 apresentado acima representa uma utilização esperada dos recursos da rede por parte de um usuário. Este recebe um IP do servidor DHCP, utiliza os recursos de rede e se desconecta da rede a partir do momento em que o tempo de empréstimo deste endereço IP termina.

Esta desconexão, no entanto, pode ser evitada por um usuário malicioso, recaindo portanto no caso 3 ($JV < 0$). Isto implica que um usuário continua utilizando um endereço IP que o servidor DHCP já recuperou para seu *pool* de endereços.

Este caso é desafiador, pois ressalta duas categorias de ataques: sequestro de sessão e Negação de Serviço (DoS).

O sequestro de sessão pode ser evitado através uso integrado do DHCP, Firewall e Sistemas DDE Seguros, como proposto em (CARRION, 2003a), (CARRION, 2003b) e (CARRION, 2003c) e (NUNES, 2004).

Os ataques de negação de serviço não são totalmente combatidos por esta solução, dado que qualquer usuário malicioso pode configurar o seu dispositivo com qualquer endereço IP, causando colisões na rede e eventualmente congestionamentos.

Indo mais além, um usuário malicioso pode lançar um ataque **ARP Poisoning**, que o torna capaz de causar uma Negação de Serviço no *switch* ou até mesmo lançar ataques **Man-in-the-Middle**, onde toda a comunicação entre dois pontos passa primeiro pelo dispositivo de rede do atacante para depois ser encaminhado ao seu destino final.

Esta dissertação não foca na solução deste tipo de ataque, mas uma possível solução para esta ameaça é a autenticação na camada 2 utilizando 802.1X, controle de acesso nos *switches*/*firewalls* e integração com autenticação de rede do usuário. Um exemplo parcial desta arquitetura pode ser vista em (Cisco, 2005).

- **Configuração default insegura: Autenticação de Atacantes e DoS**

A RFC 3118 possui um mecanismo de autenticação baseado em texto claro que permite a um atacante capturar os usuários e senhas válidos e futuramente utilizá-los para:

- Autenticar-se na rede
- Consumir todos os endereços IPs disponíveis, caso possua diversas credenciais
- Ou em um caso mais grave consumir todos os IPs disponíveis utilizando somente um usuário com endereços MAC forjados, já que a RFC não proíbe que um mesmo usuário se autentique diversas vezes concomitantemente.

Capítulo 3

Categorização dos protocolos

3.1 Introdução

OS capítulos anteriores discorreram sobre as principais características de redes sem fio, as principais falhas dos protocolos utilizados e também as características de segurança desejáveis de protocolos de autenticação e troca de chaves.

A fim de selecionar quais os protocolos que serão analisados e implementados é fundamental definir um subconjunto significativo deles. Ou seja, os protocolos de autenticação e troca de chaves devem ser categorizados de acordo com algumas características básicas.

Como pode ser visto em (STALLINGS, 1998), um processo de autenticação pode ser baseado em uma ou mais combinações das características abaixo, definidas como natureza da chave:

- Algo que se sabe - como uma senha lembrada por uma pessoa;
- Algo que se tem - como a posse de um *token* ou um *smartcard*;
- Algo que se é - como a impressão digital (biometria) .

Esta dissertação se baseia nos dois primeiros grupos, ou em senhas lembradas por um usuário (algo que se sabe) ou em certificados digitais (algo que se tem).

Estes grupos são aplicados a determinados protocolos que autenticação. Isto implica que **saber**, **ter** e **ser** servem de entrada para protocolos de autenticação (por vezes protocolos de troca de chaves também) a fim de prover uma determinada saída: autenticado ou não-autenticado, sucesso ou falha.

Dos protocolos de autenticação destacam-se três tipos de acordo com o tipo de compartilhamento da chave:

- Protocolos que utilizam chaves simétricas;
- Protocolos que utilizam chaves assimétricas;
- Protocolos que utilizam chave/verificador.

Destes protocolos existem um grande número de implementações e propostas, como por exemplo protocolos que utilizam criptografia simétrica baseados em algoritmos de cifra de fluxo, quanto em algoritmos de cifra de bloco, por exemplo.

Identificam-se 4 (quatro) grupos básicos de protocolos de autenticação, que combinam natureza das chaves e tipo de compartilhamento das chaves:

1. Protocolos baseados em texto claro;
2. Protocolos baseados em chave simétrica (desafio-resposta);
3. Protocolos baseados em chaves assimétricas (certificados digitais)
4. Protocolos baseados em chave/verificador.

Estes quatro grupos serão a célula básica de definição de análise e implementação. No entanto, para cada grupo existe uma vasta gama de propostas e as mais pertinentes devem ser selecionadas. Categorizar dentro destes grupos envolve as seguintes definições estabelecidas neste trabalho:

- Implementação: Facilidade de implementação do protocolo, tanto em escala industrial, quanto aos recursos possíveis para o desenvolvimento desta dissertação;

- Inovação: Protocolos recentemente desenvolvidos ou protocolos mais antigos, porém, recentemente empregados em redes sem fio;
- Troca de chaves: Se o protocolo efetua troca de chaves, além do processo de autenticação;
- Utilização: Protocolos que são implementados em diversos produtos de mercado atualmente;
- Pesquisa: Protocolos que são pesquisados para aplicação em redes sem fio atualmente.

Segue um sumário das quatro categorias fundamentais e de suas características básicas.

3.2 Categorias

3.2.1 Baseados em Texto claro

Existem alguns protocolos com características rudimentares de segurança, como por exemplo, aqueles que se utilizam de autenticação baseada em texto claro:

- HDLC (*High-level Data Link Control*) e PPP (PAP), protocolos ponto-a-ponto geralmente utilizado em enlaces seriais entre roteadores
- RFC 3118, que provê autenticação para as mensagens DHCP e possui a opção de se enviar a senha em texto claro

Obviamente estas propostas possuem um nível de segurança muito baixo, pois a interceptação das senhas é suficiente para que a segurança do sistema seja afetada. Vale ressaltar que protocolos ponto-a-ponto utilizados em meios confinados, em última instância, que trafegam em meio cabeado, creditam grande parte da segurança do protocolo de autenticação à dificuldade de se instalar escutas no meio da comunicação.

A segurança física destes meios torna-se fundamental e a única barreira para a manutenção da segurança lógica e portanto o devido controle de acesso. A utilização de um segundo nível de defesa é altamente desejável e de certa forma mais eficaz e escalável do que proteger todo o meio físico através do qual os dados fluirão.

3.2.2 Baseados em algoritmos simétricos utilizando desafio-resposta

Protocolos desta natureza possuem um risco associado à descoberta da senha pré-compartilhada através da análise das mensagens de autenticação.

Por vezes, a senha pré-compartilhada é escolhida por um ser humano através de processos mnemônicos a fim de facilitar a sua memorização. No entanto, estes processos mnemônicos implicam no uso de um pequeno subconjunto das senhas possíveis possibilitando a dedução das mesmas por atacantes ou sua descoberta através de ataques de dicionário ou de força bruta.

Outro risco está associado à segurança do banco de dados de credenciais que armazena as senhas pré-compartilhadas entre o servidor e os clientes. Neste caso, a exposição das senhas através de um ataque bem sucedido ao banco de dados de credenciais implica na ruptura da confiança entre as partes (cliente-servidor)

3.2.3 Baseados em algoritmos assimétricos utilizando certificados digitais

A autenticação através de certificados digitais pode ser observada na equação abaixo:

A -> B : $Cert_A$
B -> A : N_b
A -> B : S_a (N_b)

A envia seu certificado que pode ser auto-assinado ou assinado por uma autoridade certificadora para B, este envia um desafio para A (N_b) e A assina N_b enviando o resultado de volta para B, que se certificará da validade do mesmo verificando a assinatura com o certificado de A.

Primeiramente deve-se observar que existe um conjunto de requisitos de implementações que deve ser obedecido de forma que os algoritmos de chave pública não sejam suscetíveis a ataques como abordado em (SCHNEIER, 1996).Estes requisitos dificultam em muito a escolha eficiente de um mecanismo de autenticação utilizando certificados digitais.

Outro aspecto importante se refere à utilização de um certificado digital que a princípio deve ser assinado por alguma entidade. Um certificado auto-assinado possui um significado bastante limitado no procedimento de autenticação e carece de escalabilidade.

A solução evidente neste caso se refere a utilização de uma infra-estrutura de chaves públicas (ICP ou *PKI - Public Key Infrastructure*) onde um entidade em que ambos confiam (Autoridade Certificadora - AC) assina, gera e revoga os certificados.

Neste caso B pode verificar se o certificado enviado por A é válido desde que ele tenha sido assinado pela chave privada da CA em que ele confia. Neste caso esta validação é feita através da chave pública da CA que é amplamente conhecida.

No entanto, a apresentação do certificado não é garantia de autenticidade pois qualquer atacante pode apresentar um certificado previamente utilizado em uma autenticação prévia, até porque os certificados são de natureza pública mesmo.

O que garante a autenticação é o fato de A responder ao desafio de B através da assinatura do *Nonce* com sua chave privada par do certificado apresentado, esta sim somente de posse do sujeito que apresentou o certificado no primeiro passo.

A validação do certificado garante que a entidade que se apresenta é A e que qualquer operação feita com a chave privada de A será desfeita com a chave pública de A contida no certificado.

Isto é o que ocorre após o passo três do algoritmo acima: após receber a resposta, B verifica a assinatura decriptando seu conteúdo com a chave pública e verificando se o valor decriptado corresponde ao desafio enviado no segundo passo.

3.2.4 Baseados em chave/verificador

Estes algoritmos utilizam uma chave de conhecimento do usuário e de um verificador desta chave armazenado em um banco de dados de autenticação, sendo assim toda chave possui somente um verificador próprio e vice-versa.

Um arranjo semelhante se observa em protocolos baseados em chaves públicas que são compostos por chaves privadas e chaves públicas. De certa forma um corresponde ao verificador do outro.

No entanto, tanto a chave privada como a chave pública devem ter mais de 1024 bits para garantia de segurança, ou seja, para que não seja possível descobrir uma a partir da outra.

Isto não ocorre nos protocolos descritos nesta seção, pois estes foram desenvolvidos com o objetivo de se utilizar chaves capazes de serem facilmente lembradas por seres humanos (por exemplo chaves entre 8 e 16 caracteres).

Para os algoritmos baseados em certificados digitais não há nenhum comprometimento da segurança do sistema quando há divulgação da chave pública, até porque sua finalidade implica em que ela seja pública.

Esta característica permite inclusive que uma determinada informação seja assinada por uma entidade e sua autenticidade comprovada por outras, desde que exista uma relação de confiança entre elas, que normalmente se dá através de uma Autoridade Certificadora e portanto de uma Infra-estrutura de Chaves Públicas [1].

[1]Esta funcionalidade não tem sido explorada nas redes sem fio, no sentido de comprovar uma determinada informação para um terceiro, ou seja, alguma entidade além do cliente (usuário) e do servidor (ponto de acesso) quando de sua aplicação como protocolo de autenticação em redes sem fio. Os protocolos apresentados nesta seção não possuem esta característica inerente, pois focam na autenticação cliente-servidor

Por outro lado, protocolos baseados em chave/verificador devem manter seguros os seus verificadores, porque o conhecimento destes permite que um usuário malicioso se passe por um servidor autêntico e efetue um ataque *Man-in-the-Middle*, por exemplo.

A dificuldade de se descobrir a chave a partir do verificador é condição necessária para que este tipo de protocolo tenha sucesso, ou seja, a solução ou cálculo necessários para descoberta da chave a partir do verificador é computacionalmente inviável em um tempo menor do que idade do universo, por exemplo. Como exemplo tem-se o Logaritmo Discreto, a Inversão de Função de Hash e a Fatoração de Números Primos (SCHNEIER, 1996).

Observa-se portanto, que a descoberta do verificador por um atacante não implica a descoberta da senha do usuário correspondente. Isto porque a descoberta desta senha implica em um problema computacionalmente inviável de ser resolvido como descrito no parágrafo anterior.

A facilidade de se definir um verificador a partir de uma chave é outra condição fundamental para o sucesso deste tipo de protocolo. Isto permite que a escolha de uma chave para um usuário implique na definição simples e direta do seu verificador par.

Para o protocolo SRP a definição de um verificador é dada por:

$$v = g^x \bmod N$$

onde,

- v : verificador da chave x
- x : chave ou senha do usuário
- g : gerador do campo[2]
- N : Módulo

A chave do usuário nunca é transmitida em texto claro e não há equivalência ao texto claro, pois o servidor armazena o verificador. Descobrir a chave a partir do verificador é uma tarefa computacionalmente inviável, pois exigiria o cálculo do logaritmo discreto:

$$x = log_g\ (g^x \bmod N)$$

A Figura 3.1 apresenta a troca de mensagens do protocolo SRP. Carol representa um usuário (cliente) e Steve um ponto de acesso (servidor). Steve possui o verificador da chave de Carol, além de **s** que corresponde ao Sal (*salt*) [3].

	Carol		Steve
1.		$\xrightarrow{C}$	(lookup s, v)
2.	$x = H(s, P)$	$\xleftarrow{s}$	
3.	$A = g^a$	$\xrightarrow{A}$	
4.		$\xleftarrow{B,u}$	$B = v + g^b$
5.	$S = (B - g^x)^{a+ux}$		$S = (Av^u)^b$
6.	$K = H(S)$		$K = H(S)$
7.	$M_1 = H(A, B, K)$	$\xrightarrow{M_1}$	(verify M_1)
8.	(verify M_2)	$\xleftarrow{M_2}$	$M_2 = H(A, M_1, K)$

Figura 3.1: Troca de mensagens do protocolo SRP (fonte (WU, 1998))

A chave efetivamente utilizada corresponde ao *hash* da concatenação da senha do usuário (**P**) com o salt **s**. Tanto Carol e Steve geram chaves públicas temporárias **A** e **B** respectivamente[4].

Carol e Steve calculam no passo 5 o valor da chave trocada **S** e em seguida a chave **K** que será utilizada para encriptar a sessão e confirmar que o processo de autenticação foi completado com sucesso tanto para o cliente quanto para o servidor.

[3]Salt é um valor concatenado à chave do usuário para dificultar a construção de base de dados com senhas pré-calculadas (*Rainbow-Tables*). No Unix o *salt* corresponde a dois caracteres adicionais à senha do usuário.

[4]Steve também gera um valor aleatório **u** devido a falhas que poderiam ser exploradas por Carol (WU, 1998)

PFS

Afirmação: A descoberta da senha **P** de Carol e portanto a chave utilizada para autenticação no protocolo SRP não implica na descoberta das chaves de sessão **K** ou **S** já trocadas em execuções anteriores do protocolo entre Carol e Steve.

Narrativa: Para calcular **S** um atacante precisa ter acesso à chave privada de A ou de B. Para tanto o atacante deve executar o cálculo do logaritmo discreto ou um ataque de força bruta. Isto é computacionalmente inviável.

Replay

Afirmação: Um atacante não pode efetuar ataques *replay*, reinserindo mensagens de autenticações válidas anteriores.

Narrativa: Em cada sessão tanto o cliente quanto o servidor geram números aleatórios correspondentes às chaves públicas temporárias **A** e **B**. Estes valores mudam para cada sessão e impedem que um atacante venha a calcular corretamente a chave **S** e obtenha sucesso na verificação de autenticação efetuada nos passos 7 e 8.

ETC e descoberta da senha P

Afirmação: O servidor não armazena o Equivalente de Texto Claro da senha do usuário e as informações trocadas pelo protocolo não possibilitam a recuperação da senha **P** em um tempo razoável[5].

Narrativa: Como dito anteriormente o servidor armazena o verificador da senhas do cliente e a descoberta desta senha partindo do verificador é computacionalmente inviável. Para descobrir uma chave **P** via ataque de dicionário um atacante deveria efetuar os seguintes cálculos, dado que ele tenha armazenado uma troca de mensagens válidas de autenticação:

1. Eleger uma senha candidata a partir do dicionário de senhas

[5]Define-se neste texto como tempo razoável a um tempo inferior à idade do próprio Universo

2. Calcular o *hash* desta senha concatenada com o *salt*

3. Eleger uma chave privada candidata que tenha gerado A ou B

4. Calcular S candidato

5. Calcular o *hash* de S para obter K

6. Calcular o *hash* da concatenação de A, B e K

7. Se o cálculo no passo anterior for igual à M1 então a senha candidata do passo 1 corresponde à senha do usuário

O passo 3 exige um ataque de força bruta ou a inversão do logaritmo discreto o que é computacionalmente inviável, além disto os cálculos nos passos 3 e 4 consomem poder de processamento muito maior se comparados à cálculos de *hash*, aumentando o tempo médio necessário para descoberta das senhas.

Efetivamente o passo 3 previne a descoberta da senha P partindo-se da observação das mensagens trocadas entre Carol e Steve.

No entanto, um atacante poderia efetuar um ataque interativo e desta forma não precisaria calcular o logaritmo discreto de A ou de B. Neste caso o atacante se passaria por um usuário válido da rede e efetuaria a troca de mensagens normalmente.

1. Eleger uma senha candidata a partir do dicionário de senhas

2. Calcular o *hash* desta senha concatenada com o *salt*

3. Calcular S candidato

4. Calcular o *hash* de S para obter K

5. Calcular o *hash* da concatenação de A, B e K (denominado M1)

6. Enviar M1 e caso tenha sucesso receberá M2 do servidor

O dicionário de senhas utilizado pelo **John the Ripper** (JOHN, 2005), programa para quebra de senhas do UNIX possui cerca de 20.000 senhas. Assumindo

que uma senha pode ser quebrada em média nas 1000 primeiras tentativas haveria 999 entradas no *log* do servidor informando as tentativas inválidas.

É usual que o servidor limite as tentativas inválidas de *login* entre 3 a 10 tentativas, com bloqueio temporário de 30 minutos, por exemplo. Desta forma um atacante poderia tentar descobrir 3 senhas a cada 3 minuto, um tempo aproximado de 9960 minutos, ou seja, 166 horas ou 7 dias.

Espera-se que este tempo seja suficiente para alertar o administrador que seu sistema de autenticação está sob ataque e tomar desta forma providências adicionais.

3.3 Categorização dos protocolos

A tabela abaixo apresenta os tipos de autenticação considerados nesta dissertação (Texto claro, Desafio-Resposta, Certificados Digitais e Chave/Verificador) com as características utilizadas para categorização dos protocolos.

Os protocolos baseados em texto claro não foram implementados e analisados em maiores detalhes haja vista a fragilidade de suas premissas de segurança dado que a senha trafega em texto claro na rede. Isto elimina o HLDC e a RFC 3118 (utilizando texto claro).

Três categorias são consideradas fundamentais para aplicação em redes sem fio, sendo os protocolos baseados em desafio-resposta, certificados digitais e exponenciação. A seguir serão detalhados os motivos que delinearam quais os protocolos que foram selecionados para serem analisados com mais detalhes e/ou implementados. Os protocolos que não foram selecionados não serão mais mencionados na discussão a seguir.

Desafio-Resposta

Dos protocolos cuja implementação é considerada mais simples tem-se: EAP-MD5 e RFC-3118. Implementar o EAP-MD5 em sua totalidade não faz sentido pois alguns protocolos proprietários já executam esta tarefa (PHIFER, 2004) e a imple-

Protocolos	Autenticação	Implementação	Inovação	Troca de Chaves	Pesquisa	Utilização	Pontuação
HDLC	Texto Claro						0
RFC 3118	Text Claro	X	X		X		3
IPSec	Desafio-Resposta			X	X	X	3
WEP	Desafio-Resposta				X	X	2
EAP-MD5	Desafio-Resposta	X		X		X	3
Kerberos	Desafio-Resposta	X					1
RFC 3118	Desafio-Resposta	X	X		X		3
Portais	Certificados Digitais	X					X
EAP-TLS	Certificados Digitais	X		X		X	3
SSL/TLS	Certificados Digitais	X		X		X	3
IPSec	Certificados Digitais			X	X	X	3
AKE-SRP	Chave-Verificador	X	X	X	X		4
EKE	Chave-Verificador	X		X	X		3
SPEKE	Chave-Verificador	X		X	X		3

Tabela 3.1: Comparativo de Protocolos

mentação da RFC-3118 contribuiria com um modelo funcional desta RFC definida em 2001, mas que de acordo com o grupo HDC do IEEE até o momento não foi implementada.

No entanto, foram demonstradas nesta dissertação inúmeras deficiências presentes neste protocolo invalidando a implementação de um modelo com falhas sérias de arquitetura. Para fins de representatividade desta categoria propõe-se a implementação de um protocolo baseado em desafio-resposta utilizando SHA-1 (EASTLAKE, 2001). A definição deste protocolo e o porquê da escolha do SHA-1 como função de *hash* serão analisados em capítulo posterior.

Certificados-Digitais

O protocolo EAP-TLS será analisado em maiores detalhes por ser representativo em sua categoria e por ser referência constante na indústria e no meio acadêmico.

O protocolo EAP-TLS atua na camada de enlace, portanto considera-se fundamental implementar um protocolo de autenticação na camada de sessão utilizando o protocolo SSL/TLS. Esta implementação permitirá uma avaliação de segurança, desempenho e usabilidade mesmo quando utilizando o protocolo SSL/TLS, mas em camadas diferentes.

Chave/Verificador

Os protocolos baseados em chave/verificador são considerados o estado da arte da autenticação e troca de chaves (WU, 1998), dentre os processos baseados em senhas gravadas na memória do usuário (algo que o usuário sabe) e até mesmo para todos os demais protocolos (algo que você tem ou algo que você é)[6].

Dos protocolos apresentados o AKE-SRP é o que apresenta melhores características de segurança e desempenho como pode ser visto em (WU, 1998).

Além do desempenho e segurança providos pelo SRP, há um alto grau de inovação na utilização deste protocolo aplicado às WLANs.

3.4 Escolha dos protocolos para implementação e avaliação

De acordo com a categorização e relevância dos protocolos propõe-se a implementação dos seguintes protocolos:

1. Um protocolo baseado em desafio-resposta;

[6]Vale lembrar que os processos de autenticação podem ser baseados em: algo que você sabe (como uma senha memorizada pelo usuário), algo que você tem (como um certificado digital armazenado em um *smartcard*) e/ou algo que você é (biometria).

2. Um protocolo baseado em certificados digitais (TLS);

3. Um protocolo baseado em chave-verificador utilizando exponenciação (SRP);

4. Um protocolo baseado em desafio-resposta e no DHCP.

O EAP-TLS foi extensivamente avaliado por (MISHRA 2003a), que fornece subsídios suficientes para a comparação deste importante protocolo face aos demais que serão implementados nesta dissertação.

Capítulo 4

Sistemas implementados

4.1 Introdução

DE acordo com os sistemas de autenticação estudados e apresentados na sessão anterior, podem-se vislumbrar algumas características que diferenciavam cada uma destas propostas e a mais marcante é a utilização de certificados digitais versus senhas mnemônicas.

4.2 Características gerais de implementação

4.2.1 Arquitetura dos protocolos

A implementação dos protocolos de autenticação e troca de chaves seguiu alguns procedimentos básicos para garantir a sua correção e segurança:

- Programação segura por padrão
- Modularização
- Utilização da biblioteca de criptografia OpenSSL (OpenSSL, 2004) e de redes unpv12e (STEVENS, 1999) consagradas e extensivamente testadas

- Integridade das mensagens e uso de chaves iguais ou maiores que 128 bits

Não há funcionalidade nestes protocolos que tenha sido inserida sem que a segurança da mesma tenha sido analisada e definida.

Não há opções que façam com que os programas que implementam os protocolos propostos e analisados nesta dissertação sejam executados em modo inseguro (*downgrade*).

Diversos parâmetros em tempo de compilação foram configurados como forma de evitar *buffer overflows* e incompatibilidade de tipos, obrigando a conversões *(casts)* automáticos, tanto de tipo quanto de uso de sinal (*signed/unsigned*) que potencializam ataques de *buffer overflows*. A principal referência utilizada foi o livro de David Wheeler (WHEELER, 1999) onde os seguintes parâmetros de compilação são sugeridos:

gcc -Wall -Wpointer-arith -Wstrict-prototypes -O2 -pedantic

Parâmetro	Funcionalidade
Wall	Advertências geradas pelo compilador
Wpointer-arith	Advertências para "‘size of"’
Wstrict-prototypes	Adverte para observância dos protótipos
O2	Otimização de desempenho do código
Pedantic	Utilização padrão ANSI

Tabela 4.1: Parâmetros de Compilação

Várias informações relacionadas à utilização adequada de funções criptográficas assim como a definição de protocolos de autenticação foram retiradas de (SCHNEIER, 2003). Dentre estas se destacam o uso de *hashs*, *nonces* e a limpeza de regiões de memória. Neste último caso, a limpeza da região de memória foi levada ao extremo, não só apagando regiões que continham dados notoriamente sensíveis como credenciais, como de todas as variáveis que não estavam sendo mais utilizadas.

O princípio de modularização do código esteve em voga durante todo o desenvolvimento como forma de facilitar a aplicação de melhorias e a análise de segurança em

pequenas porções de código por vez. Com isto tem-se obrigatoriamente que se voltar os olhos para o acoplamento de funções e em especial a passagem de parâmetros.

Por último ressalta-se a utilização da biblioteca OpenSSL de código aberto e reconhecida por implementar diversos algoritmos de criptografia. Esta tarefa por si só requer um grande conhecimento dos protocolos e um grande esforço de programação, tanto no desenvolvimento quanto nos testes dos módulos desenvolvidos. O grupo do OpenSSL realiza este trabalho com sucesso há muitos anos e este não era o objetivo final desta dissertação. Outra biblioteca utilizada foi a presente no livro de Richard Stevens (STEVENS, 1998), que contém código para tratamento de erros além de permitir a homogeneização do código a ser desenvolvido.

Todas as mensagens são acompanhadas por *hashes* como forma de garantir a integridade das mesmas. O algoritmo utilizado é o SHA-1 (160 bits) (EASTLAKE, 2001) considerado mais seguro do que o MD5 (128 bits) (RIVEST, 1992). Outra característica fundamental é a utilização de chaves de sessão e senhas com pelo menos 128 bits.

4.2.2 Blocos fundamentais

- Biblioteca de redes (STEVENS, 1998)
- Biblioteca de criptografia (OpenSSL, 2004)
- Módulo de conexão GUI/Modo texto
- Módulo de troca de chaves
- Funções de transmissão/recepção (para cada etapa do protocolo)
- Cabeçalhos (.h) e funções de suporte

4.3 Implementação do AirStrikeCR

O sistema AirStrike baseado em desafio-resposta requer que o usuário entre com o par login/senha que será passado para o ponto de acesso, que por sua vez verificará

a validade destas credenciais em uma base de dados desenvolvida em MySQL.

4.3.1 Protocolo

O protocolo possui quatro mensagens e todas possuem o mesmo cabeçalho.

Todo o pacote possui o número de versão (*major* e *minor*), o contexto, o tamanho do *payload* e o próprio *payload* que corresponderá a uma determinada estrutura de acordo com o contexto.

O cabeçalho do pacote tem tamanho fixo, neste caso dado que *unsigned char* tenha um byte, o cabeçalho ocupará 4 bytes.

O contexto representa a fase em que se encontra o protocolo de autenticação que corresponde às quatro mensagens existentes no protocolo e o fim de uma rodada de autenticação.

A estrutura das mensagens segue o modelo do protocolo desafio-resposta apresentado na sessão 4.1.2.

Cada mensagem é tratada por um conjunto de funções denominadas *build* e *process*, ou seja, há uma função utilizada pelo cliente denominada buildHelloClientPayload e outra utilizada no servidor denominada processHelloClientPayload. Uma constrói a mensagem e a outra verifica os dados recebidos. O mesmo se aplica para as demais mensagens presentes neste protocolo.

4.4 Implementação do AirStrikeTLS

O AirStrikeTLS baseia-se inteiramente no protocolo TLS e utiliza-se largamente da API (*Application Programming Interface*) do TLS desenvolvida pelo OpenSSL. O usuário fornece o caminho para o certificado, o qual é passado como parâmetro para a API do OpenSSL, além de uma senha que será utilizada para decriptografar a sua chave privada para que posteriormente esta seja utilizada para encriptar um

desafio enviado pelo servidor[1].

O servidor aguarda as conexões SSL e verifica a validade dos certificados e se os mesmos são assinados pela autoridade certificadora na qual este confia [2]. Importante ressaltar que na implementação desta dissertação o cliente obrigatoriamente deve enviar o certificado para o servidor, fato este opcional de acordo com a RFC.

4.4.1 Protocolo

Inteiramente baseado no protocolo TLS (DIERKS, 1999) e na API disponibilizada em (OpenSSL, 2004).

4.5 Implementação do AirStrikeSRP

O AirStrikeSRP baseia-se no protocolo SRP e utiliza a API disponibilizada em srp.stanford.edu. A API não provê um completo encapsulamento das funções, mas foi fundamental para a diminuição no tempo de codificação do protocolo.

4.5.1 Protocolo

O protocolo possui quatro mensagens e todas possuem o mesmo cabeçalho.

Todo o pacote possui o número de versão (*major* e *minor*), o contexto, o tamanho do *payload* e o próprio *payload* que corresponderá a uma determinada estrutura de acordo com o contexto.

O cabeçalho do pacote tem tamanho fixo, neste caso dado que unsigned char tenha um byte, o cabeçalho ocupará 4 bytes.

[1]Este fato garante que o cliente é quem diz ser a partir do momento em que o servidor recebe a resposta e verifica a validade da mesma utilizando o certificado do cliente assinado pela autoridade certificadora de sua confiança.

[2]Por vezes pode ser necessário identificar toda uma cadeia de certificados digitais denominada de cadeia de confiança.

O contexto representa o estado em que se encontra a sessão que executa o protocolo de autenticação, ou seja, corresponde a uma das quatro mensagens existentes no protocolo e o fim de uma rodada de autenticação.

Interessante notar que o **processamento das informações do protocolo estão totalmente encapsuladas pela API da biblioteca libsrp** e que as mensagens desenvolvidas são o arcabouço para o transporte deste processamento.

As credenciais dos usuários são mantidas no próprio ponto de acesso utilizando uma variante do `passwd` denominada de `tpasswd`. A própria biblioteca libsrp possui ferramentas para converter o `passwd` para o formato `tpasswd` e para a criação de novas senhas de acordo com este novo formato.

É perfeitamente possível criar um banco de dados contendo estas credenciais no formato utilizado pelo SRP, inclusive a própria API prevê isto, mas requer um esforço de programação adicional que não contribuiria para provar o conceito, ou seja, a garantia de segurança e desempenho oferecidos por este protocolo, os quais são os principais objetos desta dissertação.

4.5.2 Interface

A interface do cliente AirStrikeSRP é semelhante ao do AirStrikeCR e AirStrikeTLS, como pode ser visto na figura 4.1:

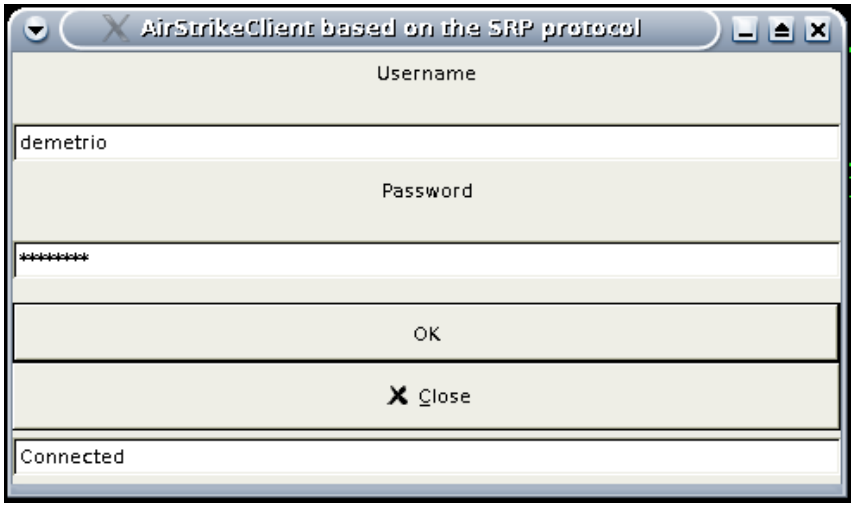

Figura 4.1: Tela do cliente AirStrikeSRP

4.6 Implementação do AirStrikeDHCP

A autenticação das estações ou dos usuários através do protocolo DHCP não é uma novidade em si. A RFC 3118 de 2001 propõe a utilização de novas opções no protocolo DHCP (DROMS, 1997) com o intuito de autenticar usuários na rede no momento em que uma oferta de IP é feita a uma determinada estação.

Este mecanismo garante também a integridade das informações que trafegam entre as estações e o servidor DHCP. Este protocolo já foi explorado em maiores detalhes na seção **Mensagens DHCP Autenticadas** desta dissertação, mas vale relembrar os aspectos fundamentais já que a proposta a seguir se baseia na autenticação através do DHCP, no entanto utilizando uma série de técnicas que melhoram a segurança do sistema e seu desempenho com relação à RFC 2131 e à RFC 3118.

O protocolo proposto e desenvolvido nesta dissertação se baseia em três decisões fundamentais de projeto, sendo elas:

1. A utilização de procedimentos de autenticação e troca de chaves é obrigatória e intrínseca ao processo de aquisição de informações e configuração da estação na rede.

2. Todo o processo de reaquisição de endereços IPs deve ser reiniciado entre reinicializações do sistema operacional do servidor e dos clientes DHCP.

3. Não há tempo de empréstimo pré-determinado, mas um sistema de Detecção de Desligamento de Estação (DDE) que verifica quais os IPs que estão disponíveis.

4.6.1 Protocolo

O protocolo AirStrikeDHCP possui uma estrutura fundamental das mensagens que se repete para todos os contextos, sendo eles:

1. DHCPDISCOVER

2. DHCPCHALLENGE

3. DHCPRESPONSE

4. DHCPOFFER

O processo de autenticação é semelhante ao de um protocolo de desafio-resposta sendo que a principal diferença neste caso consta na passagem de parâmetros de configuração de rede juntamente com as mensagens e a verificação de integridade de todos os parâmetros enviados.

Um grande desafio envolvido para o desenvolvimento do AirStrikeDHCP é o fato de a estação requerendo autenticação e configurações de rede não possuir informações suficientes para se utilizar de funcionalidades dos protocolos de camada de transporte (UDP e TCP). Em última análise a estação não possui endereço IP, máscara de rede, entre outras coisas. Faz-se necessária a utilização de APIs para acesso aos quadros de camada enlace e construção de pacotes UDP/TCP.

O cliente cria um pacote UDP e envia para uma determinada porta UDP para o endereço de *broadcast* limitado [3], além de acrescentar um número aleatório de identificação da mensagem (XID). Um servidor dedicado deve verificar todas as mensagens enviadas para esta porta e endereço e iniciar um processo filho que atenda a esta requisição em separado.

O servidor não possui o endereço do cliente de forma que a resposta seja encaminhada a um endereço *unicast*, portanto ele constrói um pacote e envia para uma outra porta UDP com endereço de destino o endereço de *broadcast* limitado.

Em contrapartida o cliente captura pacotes da rede até o momento em que filtra uma mensagem destinada a porta UDP cliente, com endereço de *broadcast* limitado e com o identificador XID inicialmente gerado por ele.

Este identificador de mensagem possibilita que vários clientes DHCP possam ser utilizados em paralelo e que um servidor mantenha um processo filho em separado para cada um deles com maior segurança. Isto porque cada cliente será identificado pela porta aleatória gerada para ele (o cliente não possui uma porta fixa) e pelo XID.

[3]Broadcast limitado é a definição do endereço IP 255.255.255.255

A figura 4.2 apresenta a máquina de estados do servidor, onde quatro estados correspondentes às quatro mensagens principais do protocolo descrevem o funcionamento deste sistema aqui proposto.

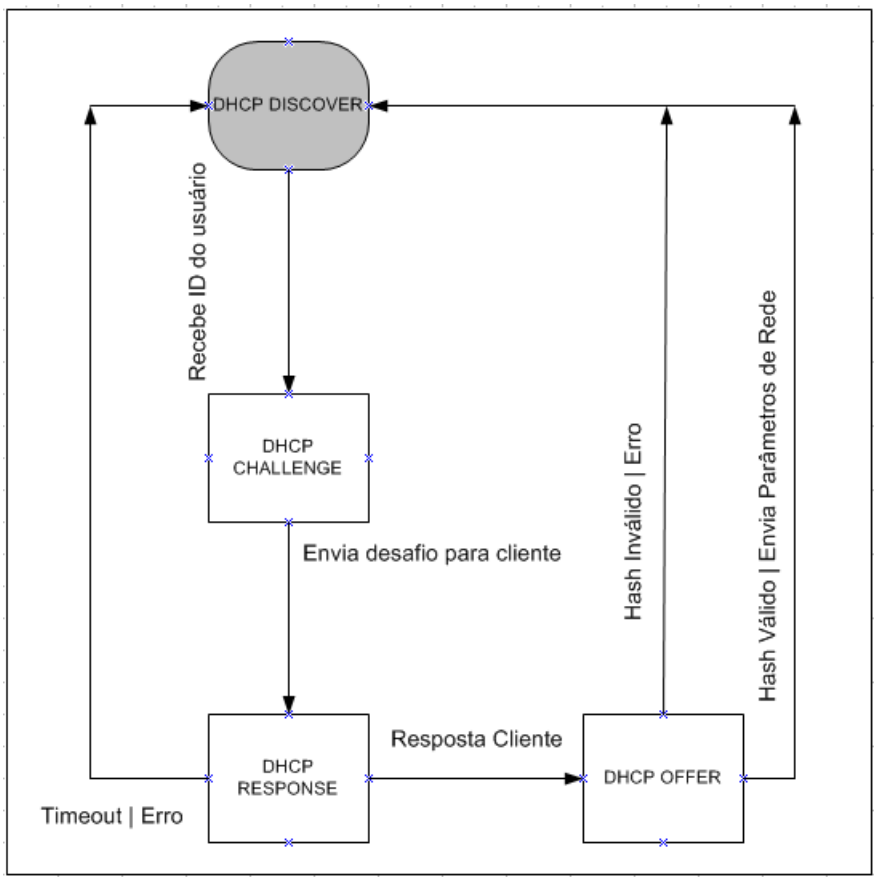

Figura 4.2: Máquinas de estados do servidor AirStrikeDHCP

As mensagens possuem a seguinte estrutura:

```
struct protocolVersion {
        unsigned char major;
        unsigned char minor;
        unsigned char patchLevel;
};

struct dhcpPacket {
        struct protocolVersion version;
        unsigned char operation;
```

```
        unsigned char xid[12];
        unsigned char username[USERNAME_SIZE];
        unsigned char nonce[NONCE_SIZE];
        unsigned char yourIP[IP_ADDR_SIZE];
         unsigned char yourNetmask[IP_ADDR_SIZE];
        unsigned char yourBroadcast[IP_ADDR_SIZE];
        unsigned char yourGW[IP_ADDR_SIZE];
        unsigned char yourDNS[IP_ADDR_SIZE];
        unsigned char yourDomain[DOMAIN_SIZE];
        unsigned char packetHash[SHA_SIZE];
};
```

Segue abaixo uma breve descrição de cada uma das variáveis definidas para utilização no protocolo AirStrikeDHCP:

- **Operation**: contexto em que se encontra o protocolo;
- **Username**: contém o nome do usuário requisitando autenticação. YourIP: endereço IP oferecido ao cliente pelo servidor a partir de um pool de endereços disponíveis;
- **YourNetmask**: máscara de rede oferecido ao cliente pelo servidor;
- **YourBroadcast**: endereço de *broadcast* oferecido ao cliente pelo servidor;
- **YourGW**: endereço do *gateway* oferecido pelo servidor;
- **YourDNS**: endereço do servidor de DNS oferecido pelo servidor;
- **YourDomain**: nome do domínio de rede oferecido pelo servidor;
- **PacketHash**: *Hash* das informações contidas no pacote concatenados às credenciais (nome do usuário e senha).

Os campos não utilizados em determinadas mensagens são preenchidas com nulos. Isto obviamente é um desperdício de banda, mas este não foi objeto de otimização pois a maior contribuição em um primeiro momento era a prova do conceito.

4.6.2 Interface

A interface do AirStrikeDHCP é toda por linha de comando.

Capítulo 5

Avaliação de segurança e de desempenho

5.1 Introdução

ESS te capítulo define o conjunto de métricas utilizado para avaliação dos protocolos apresentados e implementados nas sessões anteriores.

Será descrito o ambiente de teste definido para a execução dos testes e um gráfico contendo um comparativo de latências dos protocolos AirStrikeCR, AirStrikeTLS, AirStrikeSRP e AirStrikeDHCP.

Por último são apresentadas considerações relativas a estes testes em conjunto com uma tabela que resume os principais resultados de acordo com as métricas definidas inicialmente.

5.2 Métricas

As métricas propostas são compostas por:

- Um conjunto quantitativo que requer um ambiente de testes para a verificação de seus valores e impacto no sistema como um todo

- Um conjunto qualitativo que se baseia em padrões de segurança, gerenciamento e facilidade de uso.

São propostas as métricas abaixo como características e parâmetros para a verificação dos protocolos de segurança.

- **Desempenho**

  Número de mensagens trocadas: Número total de mensagens trocadas entre a STA e o AP em um procedimento de autenticação válido. Os pacotes SYN, SYN-ACK e ACK correspondentes ao *3-way handshake* do TCP não foram computados.

  Número de bytes transmitidos: Número total de bytes transmitidos entre a STA e o AP em um procedimento de autenticação válido. Os pacotes SYN, SYN-ACK e ACK correspondentes ao *3-way handshake* do TCP não foram computados.

  Latência de autenticação: Diferença do tempo de chegada da última mensagem de autenticação e o primeiro pacote do *3-way handshake*. Esta latência é a observada pelo cliente, indicando o início do processo de autenticação até o momento em que o cliente recebe a mensagem sinalizando que o processo de autenticação foi validado pelo servidor (pacotes FIN não foram considerados).

  Processamento (CPU): Baseado no total de funções criptográficas calculadas para uma rodada de autenticação válida. Duas funções principais são utilizadas nos protocolos avaliados: funções de *hash* e exponenciação modular. As exponenciações modulares requerem maior tempo de processamento do que as funções de *hash* e ainda são subdivididas conforme a Tabela 5.1 .

- **Segurança**

  PFS (*Perfect Forward Secrecy*): a descoberta de uma senha pré-compartilhada ou da chave privada (criptografia assimétrica) não compromete as chaves de sessão estabelecidas anteriormente.

Resistência ao ataque Denning-Sacco (DENNING, 1981): Resistência a ataques de dicionário.

Uso de Equivalência ao Texto Claro (ETC): Ambos os lados do procedimento de autenticação (cliente e servidor) precisam ter acesso à mesma senha ou segredo de autenticação. Não é ETC quando um lado possui uma senha e o outro um verificador desta senha.

Tamanho da chave: Resistência a ataque de força bruta onde todas as senhas podem ser verificadas em um tempo hábil com poder de processamento limitado.

Atualizações: Atualizações ou correções são baseados em software, *firmware* ou *hardware*. Baseados em *software* são mais rápidos e simples de se desenvolver e de aplicar, *firmware* e *hardware* são mais difíceis, a princípio.

DTPT (*Disclosure Time to Patch Time*): Tempo decorrido entre o anúncio de uma falha de segurança até o momento em que uma correção é disponibilizada.

Equivalência: O protocolo baseia sua segurança na dificuldade de se resolver um problema matemático como: Inversão de Função de *Hash* (IFH), Fatoração de Número Primo (FNP) e Cálculo do Logaritmo Discreto (CLD) (SCHNEIER, 1996), (MENEZES, 1996).

- **Gerenciamento e Uso**

Usabilidade: Conjunto de credenciais que um usuário deve saber ou ter de forma a efetuar o processo de autenticação. Memorizar uma senha é considerado mais simples do que utilizar um *token*, *smartcard* etc.

Facilidade de gerenciamento: O esforço necessário para gerar, armazenar, recuperar e revogar um conjunto de credenciais.

5.3 Ferramentas de suporte

5.3.1 Tcpdump

O tcpdump (Tcpdump, 2004) é um dos capturadores de tráfego, *sniffers*, mais utilizados em ambientes UNIX/LINUX.

Com o tcpdump foi possível capturar o tráfego de camada 3 entre o ponto de acesso e as estações, juntamente com a hora, minuto e segundo em que este pacote é capturado. Com estas informações pode-se verificar a latência de autenticação, assim como verificar a funcionalidade dos protocolos estudados e principalmente os implementados.

As mensagens abaixo apresentam um processo inteiro com sucesso de uma sessão de captura do protocolo de autenticação AirStrikeSRP, a numeração foi adicionada à lista para facilitar a visualização.

As mensagens *1, 2 e 3* representam o *3-way handshake* com os pacotes SYN, SYN-ACK e ACK do TCP, enquanto que as mensagens *11 e 12* representam a finalização da sessão (*FIN e FIN-ACK*).

As mensagens específicas do protocolo AirStrikeSRP estão compreendidas entre as mensagens *4 e 10*.

1. 16:52:26.322990 192.168.0.2.32838 > 192.168.0.1.7779: S 958215563: 958215563 (0) win 5840 <mss 1460,sackOK,timestamp 186734 0,nop,wscale 0> (DF)

2. 16:52:26.325746 192.168.0.1.7779 > 192.168.0.2.32838: S 2262111011: 2262111011 (0) ack 958215564 win 17376 <mss 1460,nop,nop,sackOK,nop,wscale 0,nop, nop, timestamp 1390607151 186734> (DF)

3. 16:52:26.325782 192.168.0.2.32838 > 192.168.0.1.7779: . ack 1 win 5840 <nop, nop, timestamp 186734 1390607151> (DF)

4. 16:52:26.326404 192.168.0.2.32838 > 192.168.0.1.7779: P 1:22(21) ack 1 win 5840 <nop,nop,timestamp 186734 1390607151> (DF)

5. 16:52:26.351068 192.168.0.1.7779 > 192.168.0.2.32838: P 1:156(155) ack 22 win 17376 <nop,nop,timestamp 1390607151 186734> (DF)

6. 16:52:26.351124 192.168.0.2.32838 > 192.168.0.1.7779: . ack 156 win 6432 <nop,nop,timestamp 186737 1390607151> (DF)

7. 16:52:26.354452 192.168.0.2.32838 > 192.168.0.1.7779: P 22:154(132) ack 156 win 6432 <nop,nop,timestamp 186737 1390607151> (DF)

8. 16:52:26.409362 192.168.0.1.7779 > 192.168.0.2.32838: P 156:288(132) ack 154 win 17376 <nop,nop,timestamp 1390607152 186737>(DF)

9. 16:52:26.412898 192.168.0.2.32838 > 192.168.0.1.7779: P 154:178(24) ack 288 win 7504 <nop,nop,timestamp 186743 1390607152> (DF)

10. 16:52:26.511667 192.168.0.1.7779 > 192.168.0.2.32838: P 288:312(24) ack 178 win 17376 <nop,nop,timestamp 1390607152 186743> (DF)

11. 16:52:26.529900 192.168.0.1.7779 > 192.168.0.2.32838: F 312:312(0) ack 178 win 17376 <nop,nop,timestamp 1390607152 186743> (DF)

12. 16:52:26.550193 192.168.0.2.32838 > 192.168.0.1.7779: . ack 313 win 7504 <nop,nop,timestamp 186757 1390607152> (DF)

As mensagens de 4 até 10 são as mensagens do protocolo de autenticação AirStrikeSRP, que neste caso correspondem à 7 mensagens [1].

5.3.2 Flawfinder

O Flawfinder (Flawfinder, 2004) é uma ferramenta desenvolvida pelo consultor de segurança D. Wheeler (WHEELER, 2004) que tem como objetivo procurar por falhas potenciais comuns em programas desenvolvidos nas linguagens C/C++.

[1] O protocolo AirStrikeSRP é composto por seis mensagens de autenticação. Deve-se notar que cada mensagem enviada é sucedida por um ACK. As mensagens ACK são transmitidas sempre que possível juntamente com dados, mecanismo conhecido como Piggyback. A mensagem 6, no entanto, corresponde a um ACK somente, não houve o Piggyback. Por isto esta rodada do protocolo trocou 7 mensagens.

O programa contém um banco de dados de funções reconhecidamente problemáticas com relação à segurança dos sistemas. O Flawfinder faz uma varredura do código fonte em C/ C++ a procura destas funções e as categoriza entre os níveis 0 (baixa ameaça) e 5 (alta ameaça).

Um exemplo de função suscetível a ataques e falhas é a strcpy cujo protótipo é:

```
char *strcpy (char *dest, const char *orig)
```

Esta função copia o conteúdo apontado pelo ponteiro `char orig` para o endereço de memória apontado pelo ponteiro `char dest`. Um ataque do tipo *buffer overflow* (ONE, 2004) pode ocorrer caso a quantidade de memória requerida para armazenar **orig** seja maior do que quantidade de memória alocada para **dest**. Com isto `orig` preencherá a memória alocada para `dest` e continuará sobrescrevendo outras variáveis adjacentes podendo causar uma falha de proteção de memória ou sobrescrever o ponteiro da pilha, *buffer overflow*.

O Flawfinder procura, entre outras coisas, pelo uso de funções `strcpy` e recomenda o uso de uma função mais robusta com relação, `strncpy`, cujo protótipo é:

```
char *strncpy(char *dest, const char *orig, size_t n)
```

A função strncpy tem funcionalidade semelhante à função strcpy, mas somente copiará n *bytes* de **orig** para **dest**, onde usualmente n corresponde ao tamanho máximo de memória alocado para **dest** (`sizeof(dest)`).

Percebe-se portanto que `orig` não poderá escrever mais dados do que os pré-alocados a `dest`. O Flawfinder e os especialistas de segurança classificam a utilização do strncpy como mais segura, mas recomendam uma função mais robusta ainda, `strlcpy` presente no sistema operacional OpenBSD, que garante que a string de destino termine em `NULL`.

A documentação do Flawfinder indica que nem todas as falhas reportadas são falhas realmente, mas sim que são trechos de código onde normalmente surgem as

falhas de segurança exploradas pelos atacantes e portanto merecem uma atenção redobrada.

Todos os protocolos sugeridos e implementados nesta dissertação foram submetidos ao Flawfinder, mas nenhuma falha de segurança foi encontrada. Foram reportadas algumas oportunidades de melhoria relacionadas ao cuidado com strings com tamanho variado, que são mais suscetíveis a ataques de *buffer overflow*. O apêndice D mostra a varredura feita para todos os sistemas implementados.

5.3.3 BFB

BFB Tester (BFB, 2004) é uma ferramenta de verificação de segurança de programas binários. Ele testa *overflows* através de passagem de múltiplos argumentos em linha de comando e *overflows* devido à configuração de variáveis de ambiente.

Todos os sistemas de autenticação desenvolvidos nesta dissertação foram submetidos ao BFB e nenhuma falha foi encontrada. O apêndice E apresenta a saída resultante dos testes do BFB para cada um dos sistemas implementados.

5.4 Ambiente de testes

Foi montado um ambiente de testes conforme a Figura 5.1 de forma a avaliar a latência de autenticação, quantidade de mensagens trocadas e a quantidade de bytes transmitidos para cada autenticação válida dos protocolos:

- **AirStrikeCR**: sistema de autenticação baseado no modelo desafio-resposta
- **AirStrikeTLS**: sistema de autenticação baseado no protocolo TLS
- **AirStrikeSRP**: sistema de autenticação baseado no protocolo SRP

O ponto de acesso utilizado consta de um PC, com uma placa de rede Wi-Fi, executando o sistema operacional OpenBSD (CARRION, 2003a). O Ponto de Acesso possui um conjunto mínimo de processos em execução de forma a não impactar nos

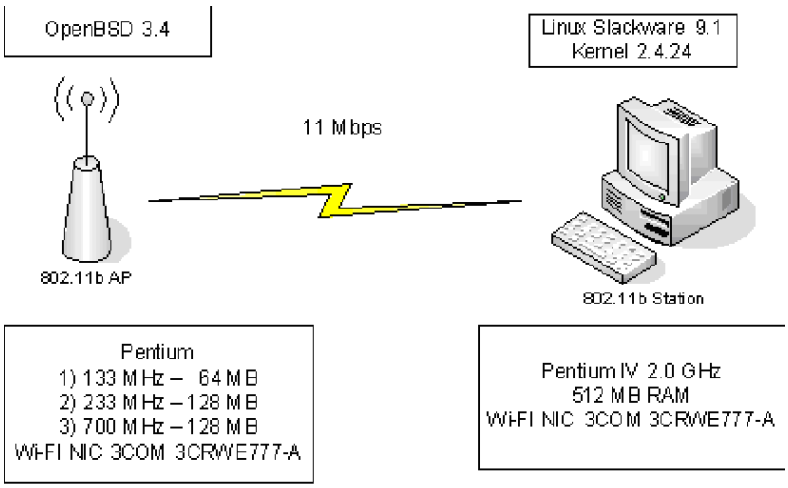

Figura 5.1: Ambiente de testes

testes. O cliente Wi-FI consta de um PC com placa Wi-Fi executando o sistema operacional Linux.

No momento do teste não existiam outros clientes ou pontos de acesso utilizando o canal Wi-Fi, nem mesmo os canais adjacentes passíveis de interferência. Isto foi verificado através do programa NetStumbler (NetStumbler, 2005) que efetua varredura de canais Wi-Fi.

Para cada protocolo foram executados 20 processos válidos de autenticação e o tráfego foi capturado com a ferramenta tcpdump. Alguns *shell scripts* foram desenvolvidos com o objetivo de facilitar a captura da latência, quantidade de mensagens e *bytes* transmitidos por rodada de autenticação.

Dados sobre o protocolo EAP-TLS foram coletados de (MISHRA, 2004), (ABOBA, 1999) e incorporados às conclusões subsequentes.

5.5 Resultados obtidos e Considerações

Foram utilizados conjuntos ortogonais [2] de métricas para se avaliar os protocolos implementados e o EAP-TLS. Algumas destas métricas foram avaliadas com o auxílio do ambiente de testes montado e apresentado na seção anterior e outras foram avaliadas de acordo com a natureza e os recursos apresentados pelos protocolos.

Primeiramente serão avaliadas as métricas quantitativas como latências, número de mensagens e quantidade de *bytes* transmitidos e em seguida serão apresentadas as métricas qualitativas como facilidade de uso e gerenciamento.

5.5.1 Expectativas

Houve uma expectativa inicial de que o protocolo baseado em desafio-resposta apresentaria a menor latência de autenticação já que ele depende da execução de funções de *hash* que computacionalmente são mais rápidas de serem processadas do que funções de exponenciação modular utilizadas pelo TLS e SRP e a quantidade de *bytes* transmitidos é menor do que dos dois protocolos. Esta expectativa foi comprovada conforme será visto adiante.

Concluir qual dos dois protocolos, SRP ou TLS, teria uma latência de autenticação menor requisitou alguns cálculos matemáticos e algumas observações.

As operações de exponenciação modular são as mais custosas em termos computacionais e consequentemente são determinantes na definição da ordem de grandeza da latência dos protocolos quando comparadas à criptografia simétrica, *hashs* e geração de números pseudo-aleatórios.

Obviamente que outras características como quantidade de mensagens trocadas e quantidade de *bytes* transmitidos são fundamentais no quadro geral de avaliação do tempo total de autenticação dos protocolos, mas pelo momento somente será

[2] Define-se ortogonalidade no sentido de que o conhecimento de um conjunto de métricas não fornece indícios concretos com relação a outros conjuntos, aplicando-se neste caso ao conjunto Segurança, Desempenho e Usabilidade/Gerenciamento

analisado o poder de processamento requerido pelos protocolos.

Desta forma, de acordo com (WU, 1998), as exponenciações modulares foram categorizadas da seguinte forma:

- Operações com base pequena (menos de 32 bits), com tempo de execução t_g
- Operações com expoente pequeno (menos de 32 bits), com tempo de execução t_e
- Operações onde base e expoentes são grandes (mais de 128 bits), com tempo de execução t_b

Isto provê o seguinte relacionamento em termos de tempo de execução: $t_b > t_g > t_e$

Foram analisados os protocolos SRP e TLS e a tabela 5.1 apresenta o tempo total de execução das exponenciações modulares efetuadas pelo cliente e servidor de ambos:

Protocolo	Cliente	Servidor
SRP	$2t_g + 2t_b$	$t_g + t_b + t_e$
TLS	$3t_e + t_b$	$2t_b + 2t_e$

Tabela 5.1: Cálculos exponenciais modulares

Desta forma verifica-se que o protocolo SRP possui requisitos de processamento maiores do que do protocolo TLS e por conseguinte o tempo de execução do mesmo é maior. Isto pode ser comprovado no gráfico apresentado na Figura 5.2 onde a primeira coluna apresenta a execução dos protocolos AirStrikeCR, AirStrikeTLS e AirStrikeSRP na mesma máquina, ou seja, tanto o cliente quanto o servidor foram executados na mesma máquina, minimizando desta forma questões como a quantidade de mensagens e *bytes* transmitidos para cada protocolo.

5.5.2 Latência e processamento

A Tabela 5.2 contem os resultados obtidos através da execução dos protocolos de autenticação AirStrikeCR, AirStrikeTLS, AirStrikeSRP e AirStrikeDHCP. Além disto apresenta-se uma tabela referente ao processo de autenticação através do AirStrkeCR sem consulta ao banco de dados[3]

Verifica-se no gráfico apresentado na Figura 5.2 a média da latência para cada um dos quatro protocolos em três plataformas diferentes de Ponto de Acesso (Pentium 133, 233 e 700 MHz), sendo que em um dos testes o cliente e o servidor estavam na mesma máquina (*loopback*):

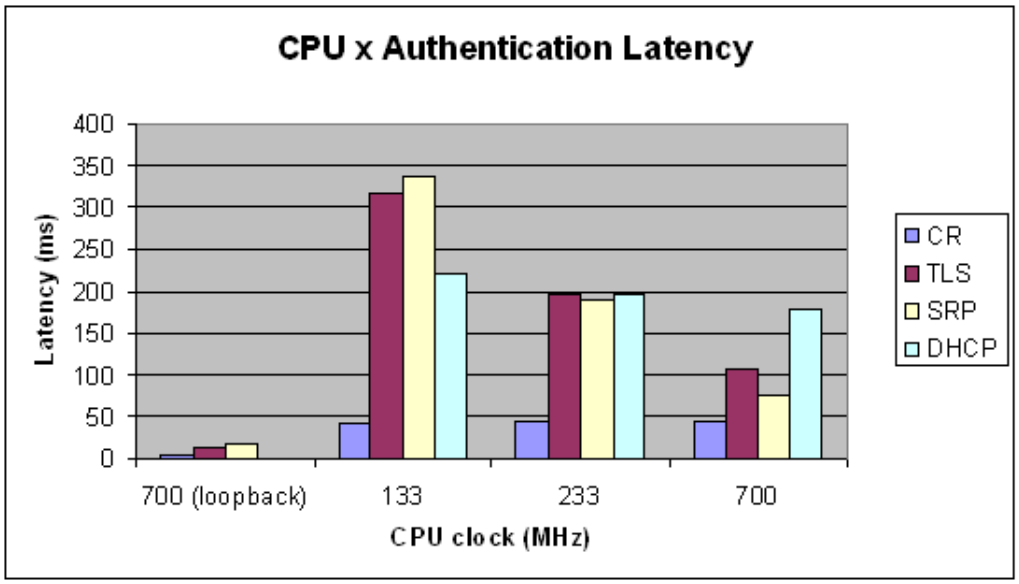

Figura 5.2: Latência de autenticação para os protocolos implementados

O teste executado em *loopback* permite verificar onde está o gargalo da aplicação: ou no processamento de funções matemáticas ou na transmissão de dados pela rede.

Percebe-se que conforme o poder de processamento aumenta menor a latência de autenticação do protocolo SRP comparada com o TLS. Isto indica que a quantidade de bytes transmitidos pelo sistema baseado no protocolo TLS transforma-se no caminho crítico do procedimento de autenticação. Para os testes executados no Ponto

[3]Este cenário corresponde a um procedimento de reautenticação onde há o cachê da chave de sessão e portanto não há consulta a um banco de dados centralizado.

de Acesso Pentium 700 com transmissão via canal Wi-FI (4º conjunto de colunas do gráfico) o protocolo SRP possui latência menor do que o TLS.

O EAP-TLS, conforme apresentado por (MISHRA 2004a) possui latência de autenticação de 800 ms quando implementado em um ponto de acesso baseado em um PC de 133 MHz e pode-se extrapolar estes resultados para se afirmar que o tempo de autenticação do EAP-TLS é muito superior quando comparado com os demais protocolos.

A avaliação das métricas qualitativas se deram através da avaliação da própria natureza dos protocolos e este resultado pode ser observado na Tabela 5.3.

MÉTRICAS	CR	TLS	SRP	EAP-TLS	DHCP
# de Mensagens	++	++	+	-	++
Wireless LAN	4	4	6	14	4
LAN	0	0	0	8	0
Total	4	4	6	22	4
Tamanho (bytes)	++	-	+	-	-
Nominal (ms)	129	3747	488	4000	1000
Normalizado	0,032	0,937	0,122	1	-
Latência	++	+	+	-	+
Nominal (ms)	44,8	107	75,5	800	179,02
Normalizado	0,06	0,13	0,09	1	0,224
CPU	++	+	+	+	++
Hash	4	24	7	28	4
t_g	0	0	3	0	0
t_e	0	5	1	5	0
t_b	0	3	2	3	0
Segurança	-	+	++	+	-
PFS	N	N	S	N	N
Denning-Sacco	S	N	N	N	Y
ETC	S	N	N	N	Y
Equivalência	IFH	FNP	CLD	FNP	HFI
Chave (bits)	64	1024	1024	1024	64
Atualizações	S	S	S	S/F/H	S
DTPT	Baixo	Baixo	Baixo	Médio	Baixo
Usabilidade	++	-	++	-	++
Sabe	1	1	1	1	1
Tem	0	2	0	2	0
Gerenciamento	++	-	++	-	++
Sabe	1	1	1	1	1
Tem	0	2	0	2	0

Tabela 5.3: Resultados dos testes de acordo com as métricas definidas

5.5.3 AirStrikeDHCP: Serviços adicionais

Comparar o AirStrikeDHCP pode fornecer uma visão deturpada em termos de segurança e funcionalidade dos quatro sistemas implementados, no entanto há uma forte contribuição quando da integração de características mais robustas de seguranças apresentadas no AirStrikeCR, AirStrikeTLS e AirStrikeSRP na arquitetura e implementação do mesmo.

O AirStrikeDHCP é um sistema de autenticação e de configuração dinâmica de *hosts* em uma rede, cuja maior contribuição é na segurança da comunicação, algo vislumbrado em (DROMS, 2001), mas que nunca fora implementado.

Na Figura 5.3 pode-se verificar as características gerais de segurança do AirStrikeDHCP, mas a todo o momento deve-se ter em mente que este protocolo fornece funções diferenciadas quando comparado aos demais, haja vista, as características de configuração dinâmica de estações que aumenta demasiadamente o tamanho das mensagens que são transmitidas na medida em que um maior número de configurações são passadas pelo servidor para seus clientes (endereço IP, máscara de rede, domínio etc).

O AirStrikeDHCP possui em geral características de segurança muito mais limitadas em comparação ao AirStrikeSRP, como pôde ser visto nesta dissertação. No entanto, o AirStrikeDHCP oferece um nível de segurança muito mais avançado do que o DHCP como apresentado em (DROMS, 1997) e mesmo quando comparado ao DHCP autenticado (DROMS, 2001).

No entanto, pode-se verificar que a latência do AirStrikeDHCP é de 179,02 ms, correspondendo a apenas 22% do EAP-TLS, sendo este último o protocolo mais utilizado para autenticação em redes sem fio, após o WEP.

A utilização do AirStrikeDHCP é factível, pois o número de mensagens e a latência de rede estão próximos dos protocolos considerados mais eficientes, mas que não efetuam configuração dinâmica de estações e por oferecer um nível de segurança superior ao proposto nas RFCs 2131 e 3118.

5.5.4 Segurança

Por último, vale ressaltar os ganhos oferecidos pelo protocolo AirStrikeSRP que é mais robusto em termos de segurança dadas as propriedades apresentadas na Figura 5.3 e também por ter latência de rede igual ou muito inferior comparando-se com protocolos mais consagrados e utilizados atualmente, dentre eles o TLS e o EAP-TLS.

5.5.5 Gerenciamento e Usabilidade

A utilização do SRP simplifica também a utilização dos recursos de rede por parte dos usuários, pois não necessita de um *token* ou certificado e também facilita o gerenciamento da rede, pois não há as dificuldades de manutenção de uma PKI, condição necessária para os protocolos baseados em TLS.

	Loopback			Pentium 133 MHz				Pentium 233 MHz				Pentium 700 MHz			
#	CR	TLS	SRP	CR	TLS	SRP	DHCP	CR	TLS	SRP	DHCP	CR	TLS	SRP	DHCP
1	9,526	12,64	17,36	42,01	319,32	349,37	221,31	43,02	195,84	188,68	175,38	40,02	109,49	77,54	170,37
2	9,012	12,50	17,51	33,17	317,96	336,26	181,44	43,36	198,11	191,08	218,67	46,60	107,90	79,60	196,90
3	9,084	12,59	17,58	38,87	313,80	334,92	241,17	41,09	198,27	191,13	250,12	41,16	100,89	77,29	240,24
4	2,481	12,11	18,16	50,62	309,90	334,87	224,53	37,37	195,35	187,73	162,04	43,20	113,87	77,20	121,46
5	2,332	12,27	17,93	42,59	314,70	334,89	201,82	42,57	192,87	189,51	218,13	34,11	104,90	72,15	210,41
6	1,645	12,10	17,62	46,23	319,46	336,53	173,39	39,14	193,86	188,64	234,54	39,33	107,32	69,26	145,90
7	2,428	12,33	18,38	50,56	313,37	334,96	156,71	48,65	195,68	189,51	196,65	52,09	103,85	69,77	171,47
8	1,692	12,37	19,02	47,50	316,00	334,78	297,63	44,39	194,72	188,50	213,74	45,14	107,05	78,98	219,92
9	1,689	12,24	17,56	47,64	314,07	339,45	228,47	49,38	191,87	189,19	196,15	48,58	107,66	69,11	194,28
10	1,774	12,26	21,29	52,16	317,85	335,53	216,98	46,09	196,26	190,84	160,92	46,61	110,25	81,36	202,49
11	2,004	12,19	19,39	45,92	313,18	333,14	266,93	49,70	187,99	189,11	170,33	49,93	111,27	80,91	142,80
12	1,666	12,33	17,59	40,03	320,63	336,07	259,64	49,13	197,38	192,43	240,62	48,66	104,50	77,27	200,53
13	1,772	12,59	17,65	43,87	319,39	334,11	253,41	49,24	190,41	190,30	163,83	47,21	110,94	71,71	165,55
14	1,67	12,78	17,11	39,62	318,99	333,65	174,80	40,22	196,62	187,64	213,68	45,25	108,75	79,51	212,84
15	5,189	12,36	17,52	48,15	320,99	338,27	264,76	38,40	197,47	188,60	157,69	51,49	109,89	76,65	136,24
16	7,936	12,24	17,92	46,08	318,61	337,61	165,35	55,50	200,30	189,92	228,51	46,51	108,51	77,41	120,92
17	5,588	12,19	17,14	40,70	314,93	337,16	196,96	42,77	193,05	190,95	222,14	52,07	101,64	73,04	161,11
18	5,216	12,52	17,46	43,49	312,87	332,69	218,27	52,51	205,30	192,35	188,36	36,52	101,52	77,13	206,44
19	5,246	12,26	17,72	40,97	315,85	332,31	254,48	45,13	202,11	189,75	194,15	44,73	109,76	72,26	140,91
20	1,951	12,15	17,62	41,30	315,48	336,85	206,12	51,10	196,13	190,66	133,31	38,32	107,37	76,67	219,60
Média	4,00	12,35	17,98	44,07	316,37	336,17	220,21	45,44	195,98	189,83	196,95	44,88	107,37	75,74	179,02
Variância	2,87	0,19	0,96	4,71	3,03	3,62	38,67	5,04	3,94	1,38	32,22	5,21	3,51	3,87	35,93

Tabela 5.2: Latência de autenticação em milisegundos

Capítulo 6

Sistema de Detecção de Desligamento de Estação (DDE)

6.1 Introdução

SIstemas de Detecção de Desligamento de Estação(DDE) são utilizados em diversos sistemas que de alguma forma necessitem monitorar a continuidade ou disponibilidade de um determinado serviço constantemente.

Isto quer dizer que a disponibilidade do serviço não é testada somente quando uma requisição de transação eletrônica é feita, mas sim ao longo de toda a conexão.

O foco desta análise é de Sistemas DDE aplicados a ambientes de rede, como por exemplo um cliente que constantemente sonda um servidor a fim de verificar se este está disponível ou não.

De forma simplificada, um sistema DDE atua através de um conjunto de mensagens como descrito abaixo:

1. Cliente envia mensagem (sondagem) para o servidor: HELLO_CLIENT
2. O servidor responde ao cliente: HELLO_SERVER

Um sistema mais simples denominado *Heartbeat* conta somente com mensagens

do servidor:

1. O servidor envia ao cliente: HELLO_SERVER

Há vantagens quanto ao sistema *Heartbeat* na medida em que há menor consumo de banda e menor atraso, pois somente uma mensagem é enviada pelo servidor.

Os sistemas DDE possuem pelo menos duas mensagens, mas este fluxo pode ser controlado pelo cliente, já que o servidor responde às requisições do cliente. Além disto há um aspecto fundamental: o sistema DDE detecta desligamentos de ambos os lados.

Uma janela de tempo pode ser estipulada pelo servidor a fim de que este considere o cliente como desligado, ou para que gere um evento em que o servidor sonde o cliente. Com isto pode-se perceber que os sistemas DDE são mais flexíveis e robustos.

Um detalhe quanto a nomenclatura diz respeito aos papéis representados pelo cliente e servidor no DDE e na própria rede. Um cliente DDE **pode** atuar na rede como um servidor e um servidor DDE atuar como um cliente na rede.

Este cenário é explorado no projeto AirStrike [Carrión 2003a], onde a estação (cliente) da rede sem fio é sondada pelo ponto de acesso (Servidor). Pela perspectiva do DDE a estação é o servidor e o ponto de acesso é o cliente.

Um dos resultados de pesquisa desta dissertação culmina com a recomendação de integração de mecanismos DDE na arquitetura do 802.1X. Uma das principais falhas deste protocolo refere-se ao roubo de sessão, no momento em que um atacante é capaz de:

- Enviar para o cliente uma mensagem 802.11 de desassociação se passando pelo servidor;
- Forjar o endereço deste cliente para se comunicar com o AP.

Isto ocorre porque há um desacoplamento das máquinas de estados do 802.11 e do 802.1X. Este risco seria mitigado na medida em que um mecanismo DDE estivesse

presente, pois o usuário malicioso não teria condições de responder corretamente às mensagens de sondagem enviadas pelo AP.

Isto não implica na solução de negação de serviço do cliente desconectado, no entanto evitaria ataques mais destrutivos como o roubo de sessão e serviria com uma camada adicional de segurança no sistema (segurança em profundidade).

6.2 Propostas atuais e suas limitações

6.2.1 Método baseado em tráfego de rede para detecção de desligamento de pares IKE

A RFC 3706 (PEERS, 2004) foi proposta ao mesmo tempo em que o projeto final de curso de graduação do DEL/UFRJ (CARRION, 2003a) estava sendo desenvolvido.

Esta RFC tem como objetivo diminuir o consumo de recursos de rede através da verificação de túneis VPN inativos e a possibilidade de desabilitá-los. Ela parte do princípio que caso haja tráfego no túnel VPN não há necessidade de envio de mensagens de sondagem. Após um período de inatividade uma mensagem é enviada a espera de uma resposta do outro lado do túnel VPN.

O túnel VPN utilizando o IPSec possui características intrínsecas de segurança como: identificação dos pontos comunicantes, confidencialidade do tráfego, prevenção contra ataques *replay* e integridade dos dados.

Esta proposta, portanto, esta diretamente ligada à disponibilidade de um canal com estas características, pois de outra forma seria semelhante a uma mensagem ECHO REQUEST aguardando um ECHO REPLAY, ou seja, uma mensagem de sondagem sem garantias de disponibilidade, integridade e confidencialidade.

6.2.2 Ping

Um mecanismo bastante simples de verificação de desligamento de estação é a utilização de mensagens ICMP *ECHO REQUEST* e *ECHO REPLY*, que em muito sistemas são representadas pelo comando ping.

Uma entidade envia mensagens *ECHO REQUEST* periodicamente e aguarda as respostas, caso um número pré-determinado de mensagens não forem respondidas em série conclui-se que estação sendo sondada foi desligada e portanto o *enlace* com esta estação pode ser terminado.

Apesar da simplicidade há uma falha de segurança relativa à capacidade de se forjar os endereços IP das estações e as mensagens *ECHO REQUEST* e *ECHO REPLY* são direcionadas aos endereços IP, não se utilizando de qualquer mecanismo adicional de autenticação.

Neste cenário um atacante encontra-se apto a roubar a sessão de um usuário válido, pois no momento em que o usuário válido se distancie de um ponto de acesso ou até mesmo desligue o dispositivo de comunicação, o atacante forjaria o IP daquele, continuando a responder às mensagens de sondagem (ICMP) enviadas pelo ponto de acesso.

6.2.3 Verificação de tabelas de estado do firewall

Este tipo de verificação é passiva, pois verifica que enquanto uma estação utiliza os recursos da rede haverá entradas na tabela de estados do firewall indicando esta utilização. Isto implica claramente que o ponto de acesso deve utilizar um *firewall* baseado em estados.

Este método não é seguro pelo mesmo motivo exposto na sessão anterior. Um atacante pode forjar o endereço IP de um usuário válido e sequestrar a sessão do mesmo.

6.2.4 Sistema isAlive-1

O isAlive surgiu como um requisito de segurança do projeto AirStrike desenvolvido em (CARRION, 2003a), sendo implementado em colaboração com outro aluno do laboratório Ravel [1], culminando com outro projeto final (NUNES, 2004).

O princípio do isAlive é bastante simples. O ponto de acesso age como cliente e se conecta a uma porta da estação, que age como servidor. De tempos em tempos o ponto de acesso envia uma mensagem de sondagem criptografada para a estação. A estação decriptografa esta mensagem e envia de volta um ACK.

Uma característica interessante neste protocolo é que o ponto de acesso envia a próxima chave que será usada para encriptar a próxima mensagem de sondagem encriptada com a última chave de conhecimento da estação e dele. Isto previne ataques *replay* desde que as chaves de encriptação não se repitam. Para tanto as chaves de sessão têm 128 bits e utilizam geradores de números pseudo-aleatórios criptograficamente fortes.

O processo se repete até o momento em que a estação é desligada ou se afasta da zona de cobertura do ponto de acesso e por conseguinte não há respostas às mensagens de sondagem enviadas pelo Ponto de Acesso. Após três sondagens enviadas e não respondidas tem-se a retirada do IP da estação das regras de *firewall*, negando portanto qualquer acesso aos recursos da rede para este IP.

A figura abaixo apresenta uma troca de mensagens válidas utilizando o isAlive:

A arquitetura e implementação do isAlive, denominado agora de isAlive-1, sofre de falhas de segurança e baixo desempenho como descobertas ao longo das pesquisas para esta dissertação.

Um dos aspectos mais frágeis deste protocolo é a troca da primeira mensagem. Neste momento, o ponto de acesso e a estação fazem uso de uma chave pré-compartilhada que pode ser alvo de ataques devido ao formato fixo das mensagens

[1] Hoje o projeto AirStrike conta com a colaboração de diversos alunos de graduação e mestrado do laboratório Ravel e sempre contou com a liderança de pesquisa do Prof. Luis Felipe M. de Moraes RAVEL-PESC/COPPE/UFRJ

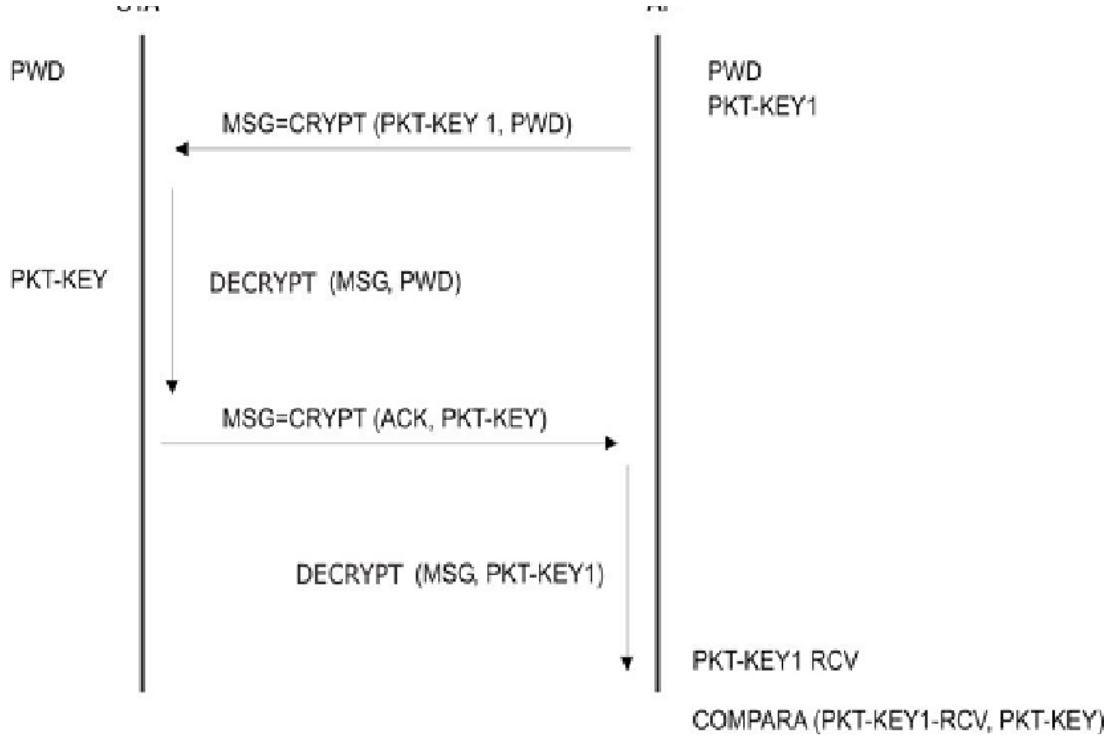

Figura 6.1: Troca de mensagens do isAlive-1

de sondagem. Isto não que dizer que o conteúdo das mensagens é fixo, mas que alguns campos sim, como por exemplo mensagens de ACK.

A troca de chaves para cada mensagem é um processo um pouco heterodoxo, pois na literatura de criptografia, vê-se largamente o uso de protocolos de troca de chaves e a utilização de *nonces* como forma de se combater este problema.

Há uma série de problemas na implementação deste protocolo, destacando-se:

1. DoS : Um inimigo pode injetar mensagens inválidas tanto no cliente quanto no servidor de forma a caracterizar não recebimento de respostas às mensagens de sondagem e consequëntemente um ataque de negação de serviço (DoS).

2. Não utilizar MAC (*Message Authentication Codes*) como MD5 e SHA-1 como forma a garantir a integridade das mensagens.

3. As estações são sondadas através de um disciplina de *polling* utilizando uma política *Round Robin*. Há problemas quanto a escalabilidade e janela de vulnerabilidade. O AP sonda uma estação e aguarda a resposta até passar para a

próxima estação. Uma estação só será considerada desligada depois da terceira passagem. Caso o timeout de espera do ACK da estação seja de 3 segundos e haja 10 estações no banco de dados, tem-se 90 segundos para que a estação seja considerada desligada.

4. Efetua-se a cada passagem (round) uma conexão com o banco de dados de forma a capturar as novas estações que se autenticaram enquanto a última sondagem estava sendo efetuada.

5. Serialização da sondagem e problemas de *race conditions* podem ocorrer através da leitura-escrita da tabela de estações autenticadas.

Para sanar estas falhas foi proposto o protocolo isAlive-2, visto na próxima seção.

6.3 Projeto e implementação do sistema isAlive-2

O protocolo isAlive-2 é uma evolução de arquitetura e programação seguras quando comparado ao isAlive-1. O isAlive-2 não contém funções inseguras, ou seja, funções como strcp e todas as variáveis de memória são apagadas quando o programa é finalizado.

Concomitante a esta evolução quanto a programação segura encontrá-se arquitetura segura do protocolo, que utiliza um algoritmo mais ortodoxo para a criptografar as mensagens e para a garantia da integridade dos pacotes.

No isAlive-1 cada pacote carregava a chave que criptografaria a próxima mensagem. No isAlive-2 utiliza-se uma chave para a conexão e um *nonce* é gerado para cada pacote, ou seja, utiliza-se um procedimento de desafio-resposta, ou seja, uma solução mais ortodoxa e testada pelo meio acadêmico e pela indústria.

6.4 Análise de segurança

As melhorias do isAlive-2 podem ser mais facilmente visualizadas quando comparadas às falhas de segurança presentes para o isAlive-1.

1. DoS : Ataques de negação são possíveis na medida em que um cliente com maior poder de processamento e com disponibilidade de maior largura de banda inunde o alvo com pacotes inválidos Isto se aplica ao isAlive-1 e diversos protocolos de comunicação. Isto ocorre para todos os servidores, mas o servidor e cliente não são alvo de mensagens de desligamento inválidas, que simplificavam significativamente ataques de DoS.

2. O isAlive-2 garante a integridade de suas mensagens através da utilização do protocolo SHA-1. Isto implica em que ambos os pontos, cliente e servidor, têm garantias quanto a validade do conteúdo dos pacotes e garantias quanto a autenticidade das mensagens.

3. As estações são sondadas em paralelo e por *threads* independentes. Isto torna o procedimento escalável e mais seguro, dado que as *threads* atuam de forma independente e a sondagem é feita em paralelo.

4. Somente uma conexão com o banco de dados é feita pelo servidor a fim de validar as credencias dos usuários no início do processo de autenticação, após o início do isAlive não há consultas ao banco de dados, pois as informações são mantidas na memória.

5. O isAlive-2 foi submetido às ferramentas BSB e Flawfinder como pode ser visto nos Apêndices C e D. Nestes apêndices são apresentadas as análises de segurança para todo o sistema desenvolvidos, o que engloba o isAlive-2.

Capítulo 7

Considerações finais

7.1 Conclusões

ESta dissertação contribui na avaliação de diversos protocolos de autenticação e troca de chaves, na implementação de dois protocolos inovadores e na definição de métricas ortogonais para avaliação de cada um dos protocolos.

O protocolo baseado no SRP constitui uma inovação dada a sua aplicação no ambiente de redes sem fio e em especial em redes baseadas no IEEE 802.11. Este protocolo é o estado da arte na autenticação e troca de chaves utilizando senhas mnemônicas.

Isto contribui aspectos como:

- Estado da arte na autenticação e troca de chaves baseado em chave/verificador
- Nível de segurança maior ou igual quando comparado à utilização de certificados digitais
- Facilidade de gerenciamento (administrador)
- Facilidade de utilização (usuário)
- Mitiga ataques ao banco de dados de credenciais

- Latência inferior quando comparado a sistemas que utilizam certificados digitais

Um aspecto muito interessante refere-se à latência do processo de autenticação no caso do SRP. Deve-se notar que o maior impacto desta latência se deve ao processamento necessário para se calcular as exponenciações modulares, o que pode ser diminuído conforme o poder computacional vai aumentando ou a incorporação de hardwares criptográficos.

No caso dos sistemas baseados em certificados digitais continua existindo exponenciações modulares, mas grande parte da latência é causada pelo grande tamanho e a grande quantidade das mensagens utilizadas, o que pode ser mitigado face ao crescimento da largura de banda disponível.

Com isto explora-se o fato de que há maiores possibilidades da latência dos protocolos SRP diminuírem se comparada a latência do sistemas que utilizam certificados digitais.

Outra contribuição desta dissertação de mestrado refere-se à implementação de um protocolo DHCP seguindo o *framework* apresentado em (Droms, 2001). O grupo de trabalho DHC (Dynamic Host Configuration) do IETF (DHC, 2004)informa em sua página que não há implementações deste *framework* até o momento.

Esta implementação baseia-se em diversos conceitos apresentados por (Droms, 2001), mas também tece críticas e apresenta soluções relativas à segurança dos protocolos envolvidos.

Analisou-se a proposta de segurança do DHCP em (DROMS, 2001) e verificaram-se diversas falhas e complexidades herdadas do próprio protocolo DHCP (DROMS, 1997). A partir disto pode-se contribuir de forma significativa com melhorias na própria máquinas de estados do DHCP, tornando-a mais simples e eficiente, e nas características de segurança.

Afirma-se portanto que o protocolo DHCP, como apresentado nesta dissertação representa um grande avanço na disponibilidade dos serviços hoje provisionados pelo DHCP como visto na RFC 2131, contendo agora nesta nova proposta aqui

apresentada maior simplicidade e segurança.

No entanto, vale ressaltar que o protocolo aqui desenvolvido é incompatível com as versões anteriores, o que pode causar dificuldades na adoção do mesmo em grande escala e no curto prazo.

Ainda deve-se comentar que o método de autenticação utilizado é baseado em um sistema de desafio-resposta, que requer pouco poder computacional e não aumenta a latência do processo de autenticação quando comparado aos demais protocolos apresentados na tabela 5.3.

Por último, o sistemas de detecção de desligamento de estação vem sendo pesquisado desde 2002 no laboratório Ravel como pode ser visto em (CARRION, 2003a) e (NUNES, 2004), que colaboraram no desenvolvimento do sistema isAlive.

Concomitante a esta proposta do Laboratório Ravel PESC/COPPE/UFRJ foi apresentado por um outro grupo de pesquisadores da comunidade internacional ao IETF (Internet Engineering Task Force) um *draft* que em seguida se tornou a RFC 3706 (HUANG, 2004) relativa à detecção de desligamento de pares IKE em túneis IPSec.

A proposta da RFC 3706 é específica para túneis IPSec e toda a sua segurança reside na segurança do túnel e dos protocolos utilizados para criação do mesmo. O foco dos autores era de extinguir túneis zumbis e otimizar a utilização de recursos computacionais.

A proposta em (CARRION, 2003a) e (NUNES, 2004) é mais abrangente, pois tem como objetivo desalocar recursos não utilizados em tempo real, ou com mínimo de latência possível.

Em (CARRION, 2003) o recurso desalocado era a permissão de acesso de um determinado IP através de um regra no *firewall*, ou seja, uma determinada regra era extinguida dado que sistema isAlive assim informasse tal necessidade.

Com isto pode-se utilizar o isAlive de forma a desalocar qualquer recurso, bastando um esforço adicional de programação. Soma-se a isto que todo o processo de segurança do sistema isAlive é intrínseco a ele, ou seja, autenticação, troca de

chaves, integridade e confidencialidade de suas mensagens.

Foram estes aspectos melhorados, otimizados e ampliados nesta dissertação com o completo redesenho do protocolo e de uma implementação que priorizou a modularidade do mesmo.

Em consonância com o desenvolvimento do isAlive-2 sugeres-se também a utilização de mecanismos DDE embutidos em paralelo com protocolo 802.1X. Com isto as falhas de segurança devido ao desacoplamento da máquina de estados do 802.1X e do 802.11 seriam mitigadas.

Em resumo, esta dissertação colaborou com:

- Compilação e categorização dos principais protocolos de autenticação e troca de chaves aplicados às redes sem fio;
- Implementação de um protocolo inovador e seguro para utilização em redes sem fio (AirStrikeSRP);
- Primeira implementação segura do protocolo DHCP seguindo o *framework* da RFC 2131;
- Definição de um conjunto de métricas ortogonais para avaliação de protocolos de autenticação e troca de chaves;
- Melhorias consideráveis de segurança e usabilidade do protocolo isAlive, agora denominado isAlive-2;
- Sugestão de utilização de sistemas DDE como forma de mitigar/prevenir falhas de segurança como o roubo de sessão no 802.1X em redes 802.11.

Capítulo 8

Trabalhos futuros

O Desenvolvimento desta dissertação permitiu ganhos expressivos com relação ao desenvolvimento de dois sistemas inovadores, um baseado no SRP (Secure Remote Protocol) e outro baseado no DHCP seguro, que apresenta inovações e melhorias consideráveis quando comparado a (DROMS, 2001).

Novas possibilidades podem ser vislumbradas como a integração do DHCP com autenticação baseada no SRP. Isto melhoraria ainda mais a segurança do DHCP, no entanto há uma possível perda relacionada ao desempenho do sistema.

A autenticação do DHCP proposta nesta dissertação utiliza um mecanismo de desafio-resposta que como comprovado nos testes apresentados no capítulo 6 é muito mais eficiente em termos de tempo de processamento do que uma autenticação baseada no SRP, vide figura 5.2.

Não obstante mostra-se interessante efetuar esta implementação de forma que possam ser verificados os ganhos de segurança versus o possível aumento da latência, fato este que precisa ser comprovado.

Julga-se também, que de acordo com o modelo adotado nesta dissertação, pode-se efetuar o desenvolvimento de um DHCP baseado em certificados digitais. Esta proposta é comentada em (DROMS, 2001) e seria mais um fator para comparação de Desafio-Resposta, Certificados Digitais e SRP, agora no domínio do DHCP.

Estas duas propostas DHCP+SRP e DHCP+TLS precisam ser avaliadas criteriosamente a fim de se verificar a possibilidade de implementação, dado o tamanho das mensagens que podem ser restritivas e a complexidade associada.

O tempo de implementação destas duas propostas não é trivial, necessitando um estudo e um investimento consideráveis em tempo de implementação.

Esta dissertação abordou os aspectos de autenticação, troca de chaves e DDE, mas não comentou de forma aprofundada mecanismos de confidencialidade.

Vale ressaltar que a troca de chaves e a autenticação propostas pelo AirStrikeTLS e AirStrikeSRP podem vir a alimentar um protocolo de confidencialidade baseado no IPSec.

Isto quer dizer que o IPSec como proposto em (KENT, 1998) poderia ser modificado para que a autenticação e troca de chaves seja feita por um dos mecanismos propostos nesta dissertação. Além disto há o ganho na utilização do isAlive, que resolveria o problema de túneis IPSec zumbis, fato este percebido e resolvido por (HUANG, 2004), mas que agora contaria com uma solução geral, não dependende dos mecanismos de confidencialidade integridade do IPSec.

Esta generalidade do isAlive possibilita a utilização de qualquer sistema ou protocolo de confidencialidade o que garante a utilização de protocolos com características específicas para cada possível aplicação.

A integração com o IPSec requer também um estudo aprofundado deste complexo protocolo, onde simplificações e novas propostas poderiam ser feitas como nos moldes apresentados em (FERGUSON, 2001) e com isto levar à integração dos resultados deste trabalho.

A implementação de protocolos como IPSec requer profundo conhecimento da pilha TCP/IP e grande habilidade de programação e Engenharia de Software.

Por último ressalta-se a integração do trabalho desta dissertação às propostas contidas em (ALBUQUERQUE, 2005) que possibilitam o *handoff* seguro de estações utilizando a solução do AirStrikeSRP. A continuidade deste trabalho contando com ambientes mais complexos, ou seja, maior número de pontos de acesso e estações,

mostra-se como uma área promissora de resultados.

Capítulo 9

Referências Bibliográficas

802.11, 1999, *IEEE Standard 802.11*, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", Edição de 1999.

802.11i, *2004 IEEE Standard 802.11i*, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 6: Medium Access Control (MAC) Security Enhancements", Edição de 2004.

802.1X, 2001, *IEEE Std. 802.1X*, "Standards for Local and Metropolitan Area Networks: Standard for Port Based Network Access Control", Oct 2001.

Aboba, B e Simon, D., 1999, "PPP EAP TLS Authentication Protocol", *RFC 2716*, 1999.

Albuquerque, L. R., 2005, "Um mecanismo de suporte à conectividade durante transições de estações móveis em redes 802.11 visando garantir recursos pré-alocados", Tese de mestrado defendida em Março de 2005 no Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ.

Arbaugh, W.A., Shankar, N. and Wan, 2001 "Your 802.11 Wireless Network has No Clothes"em Proceedings of the First IEEE International Conference on Wireless LANs and Home Networks, páginas 131-144,Singapura, Dezembro 2001.

AlohaNet, 2004, Artigo referente ao projeto AlohaNet desenvolvido em 1970 na Universidade do Havaí por Norman Abramson http://en.wikipedia.org/wiki/ALOHAnet

Berners-Lee, R. e Fielding, H. Frystyk, 1996, *RFC 1945* "Hypertext Transfer Protocol – HTTP/1.0"maio de 1996.

BFB, 2004, Brute Force Binary Tester (BFB) é um software que efetua varredura de segurança em binários. Site visitado pela última vez em Agosto de 2004. http://bfbtester.sourceforge.net/

Blaster, 2003, Informativo e segurança da Symantec quanto ao worm Blaster. http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html

Borisov, N., Goldberg, I. e Wagner D., 2001, "Intercepting Mobile Communications: The Insecurity of 802.11", em Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, páginas 180-189, Julho 2001

Carrión, Demetrio S. D., 2003a "Implementação de um ponto de acesso seguro para redes 802.11b baseado no sistema operacional OpenBSD", *Projeto Final de Engenharia Eletrônica e de Computação*, Departamento de Eletrônica da UFRJ, 2003.

Carrión, Demetrio S. D. e Moraes, Luís Felipe M., 2003b, "Implementação de um ponto de acesso para redes 802.11b baseado no OpenBSD", Apresentado no Workshop de Segurança do SBRC 2003 em Natal/RN.

Carrión, Demetrio S. D., 2003c, "Implementação de um ponto de acesso seguro para redes 802.11b baseado no sistema operacional OpenBSD", Apresentado na Primeira Semana de Eletrônica da UFRJ, Setembro de 2003.

Casole, M., 2002, "WLAN Security - Status, Problems and Perspective", European Wireless 2002.

CodeRed, 2001, Informativo e segurança da Symantec quanto ao *worm* Code Red. http://securityresponse.symantec.com/avcenter/venc/data/codered.worm.html

Cisco Inc., 2005, "Overview of 802.1x and Cisco IBNS Technology", http://www.cisco.com/application/p
ccmigration_09186a0080258e2f.pdf

Denning, D. e Sacco, G., 1981, "Timestamps in the key distribution systems", Communications of the ACM, Volume 24, Número 8, páginas 533-536, Agosto de 1981.

DHC, 2005, Página do grupo tarefa DHC (Dynamic Host Configuration) do IEEE. Página visitada pela última vez em 2005. http://www.ietf.org/html.charters/dhc-charter.html

Dierks, T. e Allen, C., 1999, *RFC 2246* - "The TLS Protocol", Janeiro de 1999.

Droms, R., 1997, "Dynamic Host Configuration Protocol", *RFC 2131*, Março 1997.

Droms, R. et al, 2001, "Authentication for DHCP Messages", *RFC 3118*, Junho de 2001.

Eastlake, D e Jones, P., 2001, *RFC 3174* - "US Secure Hash Algorithm 1 (SHA1)", Setembro de 2001.

ETSI, 2004, Portal relativo ao padrão de comunicação de redes sem fio Hiperlan, http://portal.etsi.org/bran/kta/Hiperlan/hiperlan2.asp

Ferguson, N. e Schneier, B., 2001, "A Cryptographic Evaluation of IPsec", Relatório técnico retirado de http://downloads.securityfocus.com/library/ipsec.pdf.

Finseth, C., 1993, "An access Control Protocol, Sometimes called TACACS"*RFC 1492*, Julho 1993.

Flawfinder, 2004, O Flawfinder é um software que executa varredura de segurança no código fonte em C. Site visitado pela última vez em Agosto de 2004. http://www.dwheeler.com/flawfinder/

Fluhrer, S., Mantin, I. e Shamir, A., 2001 "Weaknesses in the key scheduling algorithm of RC4", 8th Annual International Workshop on Selected Areas in Cryptography, páginas 1-24, Agosto de 2001

Godber, A. e Dasgupta, P., 2002, "Secure wireless gateway", ACM Wireless Security Workshop (WiSe 02), 2002.

Goldberg, Ian e Wagner, David, 1995, "Netscape SSL implementation cracked", Setembro de 1995: http://seclists.org/lists/bugtraq/1995/Sep/0064.html

Haartsen, Japp C., 2000, "The Bluetooth Radio System", Publicado na revista IEEE Personal Communications em Fevereiro de 2000.

Huang, G., Beaulieu, S. e Rochefort, D., 2004, "RFC 3706 A Traffic-Based Method of Detecting Dead Internet Key Exchange(IKE) Peers". February 2004.

IPSec, 2004, IPSec, Página web do Grupo de Trabalho IPSec do IEEE, Página visitada em 2004, http://www.ietf.org/html.charters/ipsec-charter.html

Klein, D. V., 1990, "Foiling the Cracker; A Survey of, and Improvements to Unix Password Security", Proceedings of the United Kingdom Unix User's Group, London ENGLAND, páginas 5-14, Julho 1990.

ITU, 2005, Página da International Telecommunication Union sobre faixas de frequência ISM e condições regulatórias. Página visitada pela última vez em 2005, http://www.itu.int/ITU-R/terrestrial/faq/index.html#g013

Kent,S. e Atkinson, R., 1998, "Security Architecture for the Internet Protocol", *RFC 2401* de Novembro de 1998.

Libpcap, 2005, Página na Internet com informações sobre a biblioteca de captura de pacotes em nível de usuário, Libpcap.
Página visitada em 2005 http://sourceforge.net/projects/libpcap/

Linux-NIS, 2004, Página na Internet com informações sobre NIS (Network Information System). Página visitada em 2005. http://www.linux-nis.org/

Mazzeo, Luzia M. et al, 2000, "Evolução da Internet no Brasil e no Mundo", Ministério da Ciência e Tecnologia, Secretaria de Política de Informática e Automação, Assessoria SEPIN, Abril/2000.

Menezes, A. van Oorshit, V. e Vanstone, S., 1996, "Handbook of Applied Cryptography", 1st edition, Ed. CRC Press, 1996.

Miller, S. P. et al, 1987, "Kerberos Authentication and Authorization System", M.I.T. Project Athena, Cambridge, 1987.

Mishra, Arunesh e Arbaugh, William, 2003, "An Initial Security Analysis of the IEEE 802.1X Standard", Technical Report, University of Maryland, Department of Computer Science CS-TR-4328, UMIACS-TR-2002-10, Fevereiro de 2001.

Mishra, A. et al, 2004a, "Proactive Key Distribution Using Neighbor Graphs"IEEE Wireless Communications., Feb 2004.

Mishra, A. et al, 2004b, "Context Caching using Neighbor Graphs for Fast Handoffs in a Wireless Network", INFOCOM 2004.

MobileIP, 2004, Página na Internet do grupo tarefa Mobile IP (IP Móvel) do IETF. http://www.ietf.org/html.charters/mobileip-charter.html

Mohney, Doug, 2003, "WiFi Wardriving", Artigo escrito em Setembro de 2003 para o site http://mrtmag.com/mag/radio_wifi_wardriving/

Moskovitz, B., 2004, "Passphrase Flaw Exposed in WPA Wireless Security", Available: http://www.technewsworld.com/perl/story/32070.html

Myers, M. et al, 1999, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", *RFC 2560*, Junho de 1999.

Myers, Eamon, 2004, "HomeRF Overview and Market Positioning", http://www.palowireless.com/homerf/homerf.asp

Murthy, U. et al, 1998, "Firewalls for Security in Wireless Networks", Proceedings of the 31st Hawaii International Conference on System Science (HICSS'98), páginas 672-680, 1998.

NetLogon, 2004, Ferramenta de autenticação centralizada, Página visitada em 2004, http://www.unit.liu.se/dokument/natverk/netlogon.html

NoCatAuth, 2004, Ferramenta de autenticação centralizada, Página visitada em 2004, http://www.nocat.org

Nunes, Bruno A. A., 2004, "Um Mecanismo Seguro de Autenticação Mútua com Detecção de Desconexão e Gerenciamento Dinâmico de Regras de Firewall", *Projeto Final do curso de Engenharia Eletrônica*, DEL/UFRJ, 2004.

OASIS, 2004, Ferramenta de autenticação centralizada, Página visitada em 2004, http://software.stockholmopen.net/

One, Aleph, 2004, "Smashing the Stack for Fun and for Profit", Phrack 49, Volume

7, Edição 49. Página visitada em 2004. http://www.insecure.org/stf/smashstack.txt

OpenSSL, 2004, Biblioteca de criptografia. Site visitado pela última vez em Agosto de 2004. http://www.openssl.org

Ossmann, Michael, 2004, "WEP: Dead Again, Part 1", Artigo publicado no site da Security Focus: http://www.securityfocus.com/infocus/1814

Phifer, Lisa, 2004 "Deploying 802.1X for WLANs: EAP Types", Artigo publicado no site Wi-FiPlanet. Página visitada em 2004.
http://www.wi-fiplanet.com/tutorials/article.php/3073201

Rigney, C. et al, 2000, "Remote Authentication Dial In User Service (RADIUS)", June 2000, *RFC 2865*.

Rivest, R., 1992, *RFC 1321* - "The MD5 Message-Digest Algorithm", Abril de 1992.

Sasser, 2004, Informativo e segurança da Symantec quanto ao worm Sasser
http://securityresponse.symantec.com/avcenter/venc/data/w32.sasser.worm.html

Schneier, B., 1996, "Applied Cryptography Second Edition - Protocols, Algorithms and Source Code in C", 2nd edition, Ed. John Wiley & Sons, 1996.

Scheneier, B e Ferguson, N., 2003, "Practical Cryptography", Editora Wiley, 1a edição, Março de 2003.

Shaw, Derek G. e Boscia, Nichole K., 2002, "Wireless Firewall Gateway White Paper - Revision 3", NASA Advanced Supercomputing Division, 2002.

Stallings, W., 1998, "Cryptography and Network Security", 2a Edição, New Jersey, Prantice-Hall, 1998.

Stevens, Richard, 1998, "UNIX Network Programming, Volume 1, Second Edition: Networking APIs: Sockets and XTI", Prentice Hall, 1998, ISBN 0-13-490012-X.

Stevens, R. 2004,Página na Internet com informações completadres ao livro (Stevens 1998) http://www.kohala.com

Stubblefield, Adam et al, 2001, "Using the Fluhrer, Mantin, and Shamir Attack to Break WEP", Relatório Técnico da AT&T Labs TD-4ZCPZZ, Rev. 2, Agosto de

2001.

Tanenbaum, Andrew, 2002, "Computer Networks, Fourth Edition", 4a Edução, Prentice Hall PTR, 2002.

Tcpdump, 2004, Software que efetua captura de tráfego. Página visitada pela última vez em Agosto de 2004. http://www.tcpdump.org

Truesecure, 2004, "A Denial of Service Attack on the TKIP (Temporal Key Integrity Protocol) Algorithm":
http://www.trusecure.com/cgi-bin/download.cgi?ESCD=W0160&file=wp_tkip.pdf

Turkulainen, Jarkko, 2004, "Securing 802.11 with OpenBSD", Página na Internet visitada em 2004. http://klake.org/ jt/tips/80211.html

Veja, 2004, "Revolução Digital", Artigo da revista VEJA Edição 1874 de 6 de outubro de 2004.

UnixInsider, 2001, Artigo do portal especializado em segurança da informação ITWorld: "Deconstructing DoS attacks", UnixInsider, Março de 2001.
http://security.itworld.com/4339/swol-0302-buildingblocks/page_1.html

Walker, R., 2000, "Unsafe at any key size: An Analisys of the WEP encapsulation", IEEE Document 802.11-00/362

WepCrack, 2005, Site de desenvolvimento da ferramenta WEPCrack para quebra de confidencialidade do protocolo WEP, http:
wepcrack.sourceforge.net/, página visitada pela última vez em Março de 2005.

Welch, Donald J. e Lathrop, Scott D., 2003, "A Survey of 802.11a Wireless Security Threats and Security Mechanisms", Relatório Técnico ITOC-TR-2003-101 para o Army G6, Information Technology and Operations Center, Department of Electrical Engineering and Computer Science, United States Military Academy, West Pont em 2003.

Wheeler, David A., 1999, "Secure Programming for Linux and Unix HOWTO", 3 Março de 2003. http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/index.html

Wheeler, David A., 2004, Site do consultor de segurança David Wheeler. Site visitado pela última vez em Agosto de 2004. http://www.dwheeler.com

Wu, T., 1998 "The Secure Remote Password Protocol", In the proceedings of the Internet Society Symposium on Networks and Distributed System Security, pages 97-111, March 1998.

Wu, T., 2002, "SRP-6: Improvements and Refinemetns to the Secure Remote Password Protocol", Submission to the IEEE P1363 Working Group, Oct 2002.

Apêndice A

SSL e TLS

O P rotocolo SSL foi proposto pela Netscape para integrar a sua linha de *browsers* para Internet e em seguida tornou-se um padrão aprovado pelo IETF (DIERKS, 1999) contendo algumas modificações sob o nome de TLS (*Transport Layer Security*). Este protocolo é o padrão nas comunicações seguras via web, reconhecida pelo prefixo de HTTPS.

O TLS é um protocolo de comunicação segura que atua na camada de sessão com propósito de encriptar todo o tráfego de uma determinada comunicação entre o cliente e servidor que estejam utilizando um protocolo confiável como o TCP.

Os objetivos do TLS são:

- Autenticar dois pares de uma comunicação (cliente e servidor)
- Trocar uma chave de sessão
- Encriptar toda a comunicação entre os dois lados utilizando a chave de sessão

O protocolo não obriga a autenticação mútua, mas sim a autenticação do servidor e para que isto seja alcançado são utilizados certificados digitais. Um diagrama com a troca de mensagens do protocolo pode ser visto abaixo:

O cliente envia uma mensagem para o servidor requisitando autenticação e informando um conjunto de cifras e protocolos de autenticação e criptografia desejados

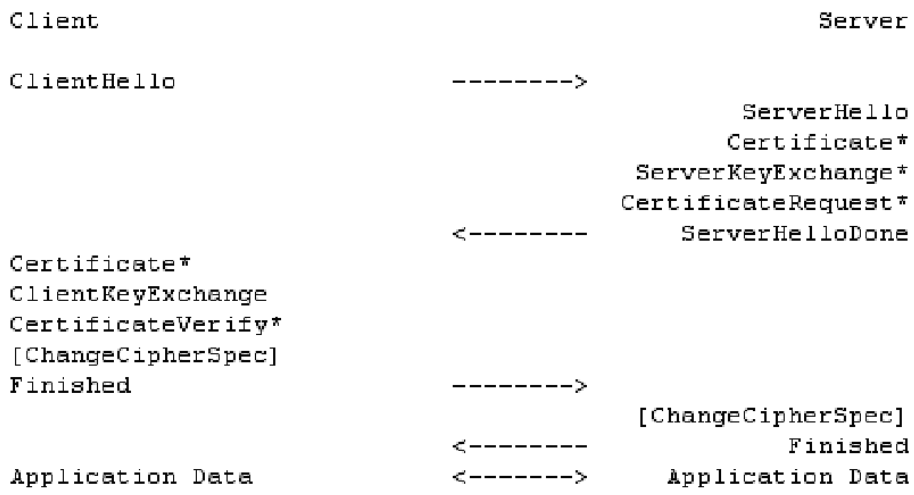

Figura A.1: Troca de mensagens do TLS fonte (DIERKS 1999)

e suportados por ele.

Em contrapartida o servidor envia uma resposta informando quais destes protocolos poderão ser utilizados, o seu certificado digital, a requisição do certificado do cliente e um desafio (*nonce*).

O cliente deve verificar a validade do certificado apresentado pelo servidor através de uma série de mecanismos dentre eles:

- O certificado não ter expirado;
- O certificado ter sido emitido para o *site* visitado, ou seja, o campo denominado *common name* (CN) presente no certificado deve ser o mesmo da URL que foi digitada para acessar o servidor;
- O certificado ter sido assinado por uma autoridade certificadora em que o cliente confia;
- O certificado não ter sido revogado e neste caso o cliente precisa ter acesso a uma lista confiável de certificados revogados (CRL - *Certification Revocation List*) ou ter acesso ao um servidor de consulta on-line (OCSP - *Online Certificate Status Protocolo*) (MYERS, 1999).

Se todos os critérios acima forem válidos tem-se a autenticação do servidor e a

partir deste momento haverá uma troca de chaves e a finalização do processo de autenticação (correspondendo às mensagens *Finished*).

No caso de o servidor requerer a autenticação do cliente, alguns passos adicionais serão conduzidos através das mensagens acima. Neste caso o servidor envia um desafio para o cliente e este o assina com sua chave privada e retornando o resultado para o servidor.

O servidor verificará a assinatura do cliente utilizando o certificado digital recebido do mesmo, efetuando as verificações supracitadas e em seguida comparará o valor decriptado com o *nonce* enviado. Caso sejam iguais tem-se a autenticação do cliente, caso contrário há uma falha de autenticação e uma mensagem de erro será gerada.

Observe que o campo *common name* não serve para verificar a autenticidade do cliente no cenário deste exemplo, pois o mesmo pode estar acessando o servidor de diversas localizações diferentes com endereços IPs diferentes a cada conexão .

Importante ter em mente que a autenticação do servidor se dá através da apresentação do certificado digital e este corresponder à URL sendo acessada e a autenticação do cliente se dá através de um mecanismo de desafio e resposta utilizando chaves públicas.

Para que este mecanismo de chaves públicas tenha validade é imprescindível a existência de uma ICP (Infra-estrutura de Chave Pública), pois os clientes e diversos servidores de uma rede devem possuir um certificado digital assinado digitalmente por uma entidade em todos confiam.

Esta entidade, Autoridade Certificadora, será responsável por emitir os certificados garantindo a veracidade das informações apresentadas pelos reclamantes (*common name*, uso do certificado etc) e a revogação dos mesmos, no caso por exemplo de um usuário ter perdido a sua chave privada correspondente ao seu certificado.

Apêndice B

Segurança de protocolos de comunicação

B.1 Introdução

DE acordo com (Stallings, 1998), os ataques à segurança dos sistemas de computadores e redes de computadores pode ser melhor caracterizados através da visualização da função dos sistemas de computadores como provedores de informação.

Desta forma um fluxo de comunicação entre uma fonte e um destino pode ser representado pelo item (a) da Figura B.1.

Partindo desta representação quatro categorias gerais de ataques são definidas: interrupção, interceptação, Modificação e fabricação. Estes grupos estão representados na Figura B.1, entre os itens (b) e (e).

- Interrupção: Um ativo do sistema se torna indisponível ou não utilizável. Esse ataque é contra a disponibilidade. Por exemplo, DoS e DDoS.
- Interceptação: Um usuário não autorizado obtém acesso ao fluxo de informações, podendo inclusive ser capaz de entender as informações transmitidos. Esse ataque é contra a confidencialidade. Por exemplo, interceptação de uma

comunicação criptografada para o qual o atacante possui mecanismos para decriptar a comunicação.

- Modificação: Um usuário não autorizado captura o fluxo de informações, modifica seu conteúdo para em seguida enviá-lo para o destino. Esse ataque é contra a integridade. Por exemplo, ataques *man-in-the-middle*.

- Fabricação: Um usuário não autorizado estabelece um fluxo de informação entre ele e o destino, fazendo-se passar pela fonte. Este ataque é contra a autenticidade. Por exemplo, roubo de senhas de um sistema seguido de estabelecimento de uma comunicação.

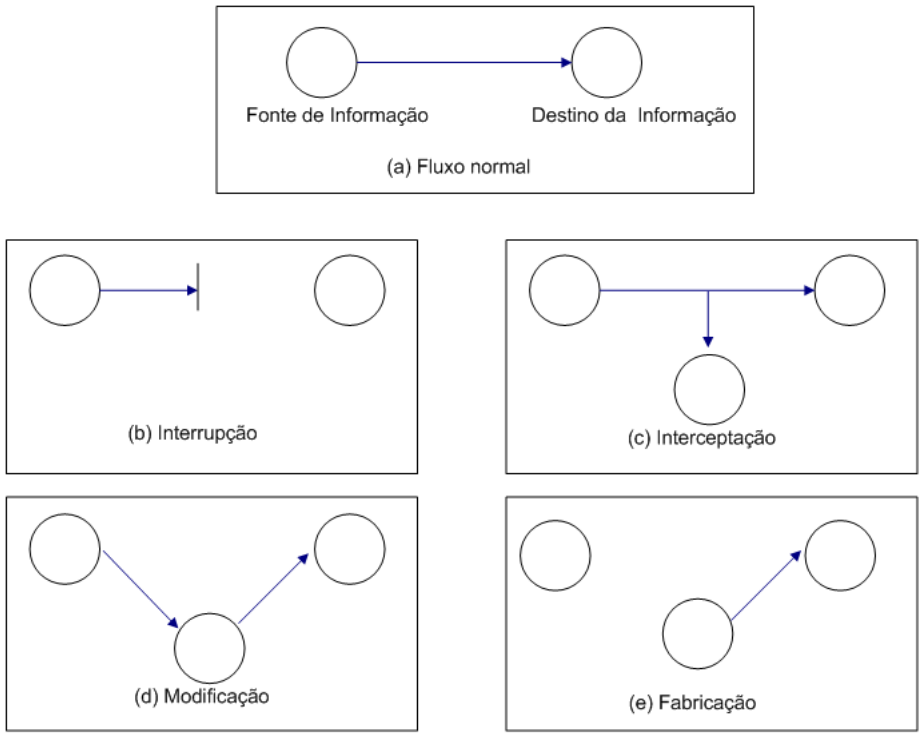

Figura B.1: Ataques à segurança da informação

Um modelo de segurança em redes de computadores também é apresentado em (Stallings, 1998), e representa diversas informações, sistemas e protocolos que por ventura estarão envolvidos na segurança do fluxo de informações, como pode ser visto na Figura B.2

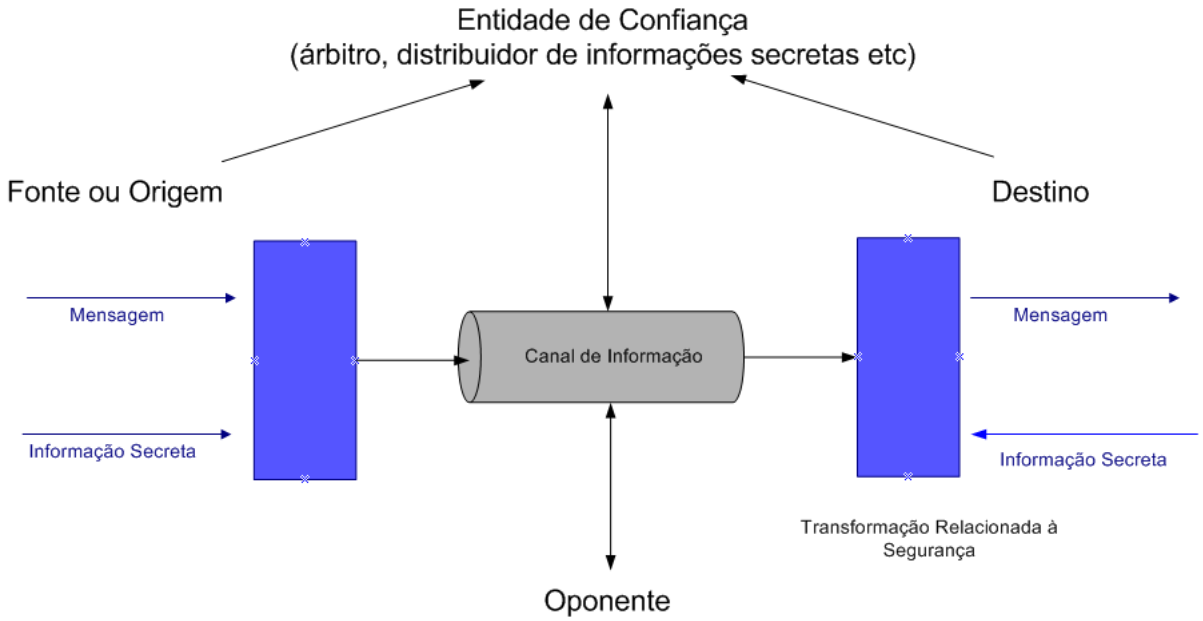

Figura B.2: Modelo de Segurança de Redes (Stallings, 1998)

Esta dissertação foco nos mecanismos/protocolos necessários para o estabelecimento de um canal seguro a partir da autenticação dos pares envolvidos (fonte e destino) (autenticidade[1]), na troca de chaves que serão utilizadas na criptografia do canal de comunicação (confidencialidade) e na detecção de desligamento de estação como mecanismo capaz de identificar automaticamente que um dos pares de comunicação ou ambos não desejam ou não são mais capazes de manter o canal seguro de comunicação (disponibilidade).

Não se quer dizer que estes protocolos de segurança apresentam soluções para todos os desafios da confidencialidade, integridade e disponibilidade, mas que estes elementos são de alguma forma contemplados por cada um dos protocolos analisados e propostos.

[1]Autenticidade pode ser vista como um elemento de disponibilidade da tríade CIA (confidencialidade, integridade e disponibilidade)

B.2 Protocolos de autenticação e troca de chaves

Os protocolos de autenticação e troca de chaves são aplicados em diversos tipos de redes, desde redes sem fio a redes cabeadas, redes *ad hoc* a redes infra-estruturadas.

Estes protocolos podem ser implementados a partir de quatro construtores básicos como visto abaixo (maiores detalhes em (Stallings, 1998))

- Criptografia simétrica
- Criptografia assimétrica
- Código de Autenticação de Mensagens (*MAC - Message Authentication Code*)
- Híbridos

Os protocolos de autenticação normalmente são sucedidos por um protocolo de troca de chaves, por vezes o próprio protocolo de autenticação incute um protocolo de troca de chaves. Quais são os objetivos destes protocolos, no entanto?

Os protocolos de autenticação identificam as entidades que desejam iniciar uma transação eletrônica e os protocolos de troca de chaves estabelecem uma chave de sessão para encriptar o tráfego entre os dois nós.

Protocolos como WEP possuem procedimentos de autenticação, integridade e confidencialidade, mas não possuem procedimentos para troca de chaves. Com isto pode-se tornar necessária a utilização de um conjunto de protocolos que possam cobrir todas as necessidades de uma determinada rede.

Um exemplo de complementaridade é dado pela utilização conjunto do WPA-2 e do 802.1X, sendo que o primeiro garante a confidencialidade e integridade e o segundo garante o controle de acesso e autenticação.

A seguir serão apresentadas características de segurança que são utilizadas para avaliar a segurança de protocolos de autenticação e troca de chaves independentes do meio físico.

B.3 Aspectos de segurança dos sistemas de autenticação e troca de chaves

Alguns ataques aos protocolos de autenticação e troca de chaves serão abordados a seguir. Estes representam os principais ataques em conjunto com os principais requisitos na implementação destes protocolos.

B.3.1 Utilização de Equivalência ao Texto Claro (ETC)

Um protocolo de autenticação mais simples poderia apresentar o fluxo de mensagens apresentado abaixo:

$$\text{A -> B} : ID_a,\ P_a$$

Esta notação, que será utilizada ao longo do texto indica que a entidade A, podendo ela ser um usuário ou um dispositivo, envia uma mensagem para a entidade B e o conteúdo da mensagem é a identificação da entidade (ID_a) e a senha correspondente (P_a). A entidade B receberá a mensagem e verificará a validade destas credenciais em um banco de dados de credencias.

Obviamente que este mecanismo é bastante inseguro, mas no entanto ele é utilizado em diversos protocolos de autenticação como visto em (Droms, 2001).

Qualquer atacante com acesso ao canal de comunicação, aspecto bastante facilitado no meio não confinado de redes sem fio, pode capturar o par de credencias e utilizá-lo futuramente como forma de conseguir acesso aos recurso de rede oferecidos.

A fim de tornar este procedimento mais seguro pode-se optar pelo envio do *hash* da senha, no lugar da senha em si, como visto abaixo:

$$\text{A -> B} : ID_a,\ \text{H}(P_a)$$

Neste caso A envia para B a sua identificação juntamente com o *hash* da senha de A. A senha de A nunca é enviada na rede, mas é fácil observar que um inimigo

não necessita da senha de A para conseguir o acesso à rede, já que a entidade autenticadora somente espera pelo *hash* da senha A que será igual em todas as sessões.

Mais uma vez, qualquer inimigo com acesso ao canal de comunicação, aspecto bastante facilitado no meio não confinado de redes sem fio, pode capturar o par de credenciais e utilizá-lo futuramente como forma de conseguir acesso aos recursos de rede oferecidos.

A falha do protocolo apresentado é que o valor do *hash* é fixo para todas as sessões, desta forma deve-se prover um mecanismo que permita que este valor mude, não possibilitando a utilização de um valor passado para efetuar uma autenticação futura.

Um protocolo que alcança este objetivo é chamado de desafio-resposta (Menezes, 1996) e (Schneier, 1996). Segue abaixo uma troca de mensagens seguindo este princípio:

A -> B : ID_a
B -> A : N_b
A -> B : H(N_b | H(P_a))[2]

A envia uma mensagem para B identificando-se e requisitando autenticação. B envia para A um desafio denominado de *nonce*[3]. A envia a resposta a este desafio que corresponde ao *hash* do *nonce* enviado por B concatenado com o *hash* da senha de A. B verifica a validade da resposta de A perfazendo as mesmas operações utilizando o *hash* da senha de A armazenado no banco de dados de autenticação e concatenando com o *nonce* enviado por ele e por fim comparando o valor calculado com o valor recebido.

[3] *NONCE* é um número aleatório gerado por uma entidade e utilizado somente uma vez. Para garantir que o número seja utilizado somente um vez dois aspectos devem ser observados: a escolha do número deve ser aleatória e o espaço para escolha deste número deve ser grande, algo em torno de 128 bits, onde o termo grande significa que os recursos computacionais e o tempo disponíveis são insuficientes para predizer o valor gerado ou até mesmo gerar todos os valores possíveis.

A utilização do *hash* da senha de A não é estritamente obrigatório para este protocolo, mas é uma conveniência na medida que o banco de dados de senhas não conterá nenhuma senha em texto claro de nenhum usuário.

O diferencial deste protocolo é que a resposta no terceiro passo sempre será diferente para cada de senha de um usuário, desde que N_b seja aleatório e escolhido de um espaço grande (por exemplo 128 bits).

Desta vez um atacante que capture uma troca de mensagens como exposta acima, somente teria em mãos: H(N_b | H(P_a)), N_b e ID_a. Este atacante poderia forjar a primeira mensagem IDa, receberia um valor N_b´ e de nada serviria utilizar a 3ª mensagem capturada já que B espera como resposta H (N_b´ | H(P_a)) o que devido ao efeito avalanche dos algoritmos de *HASH* é bastante diferente do valor H (N_b | H(P_a))[4].

Este método de autenticação era considerado robusto já que o problema de utilização da mesma mensagem de autenticação em diversas sessões fora resolvido.

Observando mais atentamente pode-se averiguar que o primeiro protocolo utiliza a senha como chave do processo de autenticação enquanto que o segundo protocolo utiliza um texto claro equivalente à senha como chave para o processo de autenticação.

Isto é denominado equivalência ao texto claro (EP *Equivalent-Plaintext*) já que a chave para autenticação não é a senha em si, mas algo que se equivale a ela no processo de autenticação. Mais uma vez: o inimigo não precisa da senha, mas apenas do equivalente a ela. O terceiro protocolo também se utiliza de EP. Em resumo tem-se a tabela abaixo:

Enfatiza-se que os dois primeiros protocolos são suscetíveis a ataque *Replay*, ou seja, a reinserções de mensagens válidas já utilizadas em uma sessão.

[4]Os valores da 3ª mensagem não precisam ser completamente diferentes, basta ser diferente em um bit para que o processo de autenticação falhe. O que se enfatiza aqui é impossibilidade computacional de haver uma colisão da 3ª mensagem, ou seja, dados um valor N_b e N_b´ o hash *gerado* ser igual.

Protocolo	Replay	EP
Envio de Senha	Sim	Sim
Envio de Hash da Senha	Sim	Sim
Desafio-Resposta	Não	Não

Tabela B.1: Equivalência ao Texto Claro e ataques *replay*

Infelizmente os protocolos baseados em desafio-resposta são suscetíveis a outros ataques como poderá ser visto adiante.

B.3.2 Ataques de dicionário

O exemplo clássico para ataques de dicionário pode ser visto em (Klein, 1990) e (Denning, 1981) e seu formato fundamental pode ser visto na Figura B.3.

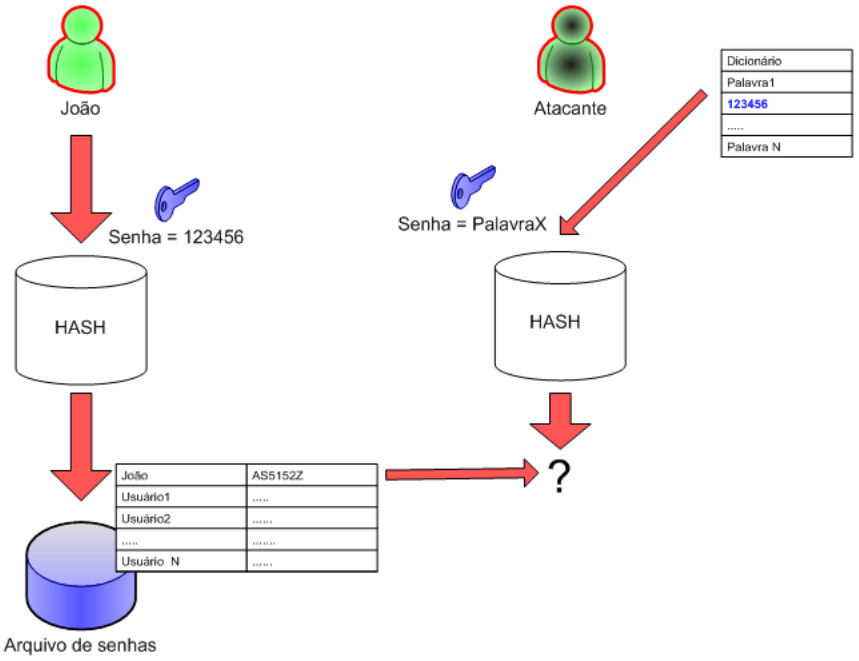

Figura B.3: Ataque de dicionário

Os sistemas UNIX possuem um arquivo de credenciais de usuários como banco de dados de autenticação denominado de `/etc/passwd` (no sistemas modernos `/etc/shadow` no Linux ou `/etc/master.passwd` no OpenBSD).

De forma sucinta estes arquivos possuem a identificação dos usuários e o *hash*[5] das senhas destes usuários.

Um inimigo poderá executar um ataque de dicionário desde que possa ler as credenciais armazenadas nestes arquivos e existem diversas formas de se alcançar este objetivo como ler diretamente o `/etc/passwd` para sistemas que não utilizam o shadow já que `/etc/passwd` tem permissão de leitura para todos os usuários do sistema ou através de outros ataques como a utilização do comando abaixo:

$ ypcat passwd

Este comando funciona para sistemas que estejam executando o NIS (*Network Information Service*) (Linux-NIS, 2004) e permite a um usuários baixar o arquivo de usuários e *hashs* de senhas para a sua máquina.

A principio, de posse do *hash* da senha e do nome do usuário caberia ao atacante tentar todas *strings* possíveis que tenham por exemplo 8 bytes, por simplicidade supõe-se que todas as senhas tenham obrigatoriamente 8 bytes e que todos os 8 bits possam ser gerados para cada byte[6]. Tem-se 264^8 strings possíveis com 8 bytes e um tempo computacional dilatado[7] até que o usuário consiga descobrir qual é a senha do usuário.

Este ataque denominado de força bruta será executado da seguinte forma:

1. Gerar uma *string* de 64 bits
2. Calcular o *hash* desta *string*
3. Comparar o *hash* calculado com o *hash* armazenado no (*/etc/shadow* por exemplo)
4. Se for igual conclui-se que a string corresponde à senha do usuário - FIM
5. Se for diferente, voltar ao passo 1.

[5]Pode-se utilizar o *hash* da senha ou então pode-se utilizar a senha como chave para um algoritmo simétrico que irá encriptar um conjunto de bits pré-definidos (64 zeros por exemplo).

[6]ASCCI utiliza 7 bits e um de paridade.

[7]Para verificar o tempo necessário para quebrar senhas de diversos tamanho veja (Schneier, 1996).

A possibilidade de *strings* diferentes gerarem o mesmo *hash* é remota, portanto pode-se afirmar com alto grau de confiança que a *string* gerada é igual à senha do usuário em questão.

O que é demonstrado em (Klein, 1990) é que os usuários tendem a utilizar um subconjunto pequeno do espaço de senhas possíveis, neste exemplo 256^8. O ser humano tem uma capacidade limitada de guardar senhas na memória e se utiliza de diversos processos mnemônicos como forma de facilitar o armazenamento desta informação.

Por exemplo, utilizar-se de nomes de pessoas, ruas, datas, o próprio nome do usuário, em suma palavras comuns que colocadas desta forma representam um dicionário ou por que não o próprio dicionário de uma determinada língua, inglês, português etc.

Apesar de um dicionário ter muitas palavras para que um ser humano decorar é fácil de se perceber que ele é um subconjunto muito pequeno de todas as 2^{256} *strings* possíveis do exemplo dado. Mais surpreendente ainda é que 25% das senhas recairão em palavras armazenadas em dicionário e algo como 40% serão derivações simples destas.

O ataque de dicionário é chamado de ataque Denning-Sacco devido aos autores do artigo e para mais detalhes deve-se consultar (Denning, 1981) e (Klein, 1990).

B.3.3 Universo das Chaves

O tamanho do universo das chaves é o número de pares de chave de criptografia disponível para um dado sistema criptográfico (Menezes, 1996). Uma condição necessária, porém não suficiente para a segurança dos sistemas criptográficos é que o universo das chaves seja grande (fato 1.40 em (Menezes, 1996)).

Como visto na seção anterio, um usuário tende a utilizar apenas um subconjunto do universo de chaves possíves, desta forma a condição necessária para a segurança do sistema criptográfico é ameaçada.

Em geral um usuário esolhe:

1. Senhas com poucos *bytes*, ou seja, poucos caracteres
2. Senhas de dicionário

Com isto diz-se que as senhas escolhidas pelos usuários tendem a ser um subconjunto limitado do universo de chaves disponíveis. Uma implicação prática disto diz respeito aos estudos atuais comparando-se o nível de segurança requerido pelos sistemas computacionais e o tamanho das senhas utilizadas pelos usuários como denotado em (Schneier, 2003) e (Schneier, 1996).

B.3.4 Ataque ao banco de dados de autenticação

A maior parte dos procedimentos de autenticação simétricos contam com o armazenamento da senha em dois pontos:

1. Na memória do usuário;
2. No banco de dados de senha (pode ser centralizado ou distribuído).

O banco de dados de senha é um ponto de ataque deste sistemas pois através dele pode-se descobrir a senha diretamente, no caso de sistemas que se utilizem de senhas em texto claro, ou *hash* da senhas, e em seguida executar um ataque de dicionário para se obter a senha em si[8], no caso de sistemas mais modernos.

Como será visto mais adiante existem protocolos que não precisam ter as senhas armazenadas no banco de dados, mas somente um verificador destas senhas. Descobrir a senha a partir do verificador é impraticável computacionalmente[9].

[8]Podem existir casos em que o atacante não precise descobrir a senha, mas somente o EP para conseguir efetuar o processo de autenticação.

[9]O termo "impraticável computacionalmente"indica que mesmo utilizando o poder computacional existente no mundo em para descobrir um valor, mesmo assim levaria um tempo impraticavelmente longo para descobri-lo, como por exemplo a idade do próprio universo.

B.3.5 Perfect Forward Secrecy (PFS)

Os protocolos de autenticação são por vezes combinados com procedimento de troca de chaves. Para fins didáticos tem-se o exemplo abaixo:

A -> B : ID_a, N_a, E_k (g^a mod N)
B -> A : N_b, E_k(g^b mod N)
A -> B : E_s (N_b, N_a)
B -> A : E_s (N_a, N_b)

Onde,

k - corresponde a uma chave pré-compartilhada
s - corresponde a g^{ab} mod N

Ao final deste processo A e B compartilham uma chave que pode ser utilizada para encriptar toda a informação transferida no canal de comunicação. Esta chave é denominada de chave de sessão, por ser derivada e utilizada a cada sessão.

No exemplo acima há uma troca de chaves baseada no protocolo Diffie-Hellman combinado com autenticação utilizando uma chave pré-compartilhada. O protocolo Diffie-Hellman efetua a troca de chaves, mas não garante a autenticação mútua e portanto é suscetível a um ataque *man-in-the-middle*. No momento em que se autenticam as trocas de Diffie-Hellman através da encriptação de (g^a mod N) e (g^b mod N) e posteriormente a sua decriptação utilizando uma chave k pré-compartilhada entre A e B, ambos os lados tem a garantia de estar recebendo a mensagem de quem eles realmente esperam.

Obviamente, existem outros fatores que devem ser observador como proteção de ataques *replay* (reinserção de mensagens válidas utilizadas em sessões anteriores), além de problemas matemáticos relacionados a teoria dos números quando da encriptação de (g^a mod N) e (g^b mod N) deixar vazar informações sobre o conteúdo encriptado (SCHNEIER, 2003).

O que se deseja ressaltar nesta sessão é a possibilidade de se proteger todas as comunicações encriptadas por suas chaves de sessão, mesmo que a chave pré-

compartilhada tenha sido comprometida.

Supondo que a chave k tenha sido comprometida e que todas as comunicações entre A e B tenham sido gravadas pelo atacante, pode-se observar que o atacante consegue descobrir os valores (g^a mod N) e (g^b mod N), antes protegidos, mas não consegue derivar a chave de sessão (g^{ab} mod N).

Somente A e B conseguem computar esta chave, pois cada um possui ou a ou b, que podem ser utilizados para calcular o valor de s tendo g^b mod N ou g^a mod N, respectivamente.

Agora, observando o protocolos abaixo:

A -> B : ID_a, N_a, E_k (a)
B -> A : N_b, E_k(b)
A -> B : E_s (N_b, N_a)

Onde,

k - corresponde a uma chave pré-compartilhada
s - corresponde (a xor b)

O protocolo apresentado não apresenta Perfect Forward Secrecy (PFS), pois a descoberta da chave pré-compartilhada (k) revela as chave de sessão que neste caso nada mais é que a xor b.

Os ataques apresentados nas sessões anteriores tornam a falta de PFS um problema de grande preocupação, pois a possibilidade de comprometimento de uma senha pré-compartilhada, possibilitam ataques de dicionário e até mesmo ataques de força bruta[10] e revelando todas as comunicações protegidas por esta chave pré-compartilhada.

[10]Por exemplo, o protocolo WEP utilizando chaves de 40 bits.

B.3.6 Equivalência criptográfica

Todos os protocolos de autenticação e troca de chaves baseiam sua segurança em algoritmos criptográficos, dentre eles destacam-se:

- Inversão de Função de Hash (IFH): O IFH representa a dificuldade de um atacante conseguir obter o texto claro dado um determinado *hash*. A segurança de uma função de *hash* reside em três aspectos (Schneier, 1996):

  Difusão, ou efeito, avalanche onde a mudança de um bit na entrada muda cerca de 50% dos bits do resultado na saída;

  Calcular o *hash* de uma entrada é uma operação simples, mas dada uma saída calcular a entrada é computacionalmente difícil, ou seja, requer um algoritmo polinomial, por exemplo;

  A probabilidade de duas entradas distintas possuírem a mesma saída é muito pequena (colisão).

- Fatoração de Número Primo (FNP): a fatoração de números primos é um problema matemático estudado há 3000 anos, desde Euclides, e o melhor algoritmo existente para fatoração de um número primo é de ordem polinomial.

- Cálculo do Logaritmo Discreto (CLD): o cálculo do logaritmo discreto possui sua melhor solução dada por um algoritmo de ordem polinomial, mas o cálculo exponencial discreto é uma operação computacional simples.

Pode-se exemplificar cada um destes protocolos tomando-se um conjunto dos mais significativos e utilizados atualmente como:

- Baseado no IFH: EAP-MD5;
- Baseado no FNP: TLS-RSA;
- Baseado no CLD: TLS-Diffie-Hellman.

Outros protocolos de autenticação e troca de chaves utilizam outros mecanismos criptográficos não analisados aqui, como protocolos baseados em curvas elípticas,

baseados em *One-Time Passwords*, utilizando algoritmos de fluxo como RC4 no caso de WEP etc.

B.3.7 Negação de serviço (*Denial of Service* - DoS)

Há uma vasta gama de ataques de negação de serviço, dentre eles:

- Consumo de CPU, de memória, de banda etc
- Interferência nas frequências de comunicação em redes sem fio (*jamming*)
- Consumo de endereços IP de um servidor DHCP
- Conexões meio-abertas [11] (*Half-Open Connections*)

Dentre os ataques apresentados acima os mais significativos na implementação dos protocolos de autenticação e troca de chaves no contexto desta dissertação é o consumo de todos os endereços IPS disponíveis para empréstimo de um servidor DHCP e o estabelecimento de conexões meio-abertas.

O primeiro ataque é característico de uma determinada implementação, que neste caso é a utilização do protocolo DHCP como forma de prover informações de configurações de rede para os clientes em uma determinada rede.

Na proposta original do DHCP (Droms, 1997) é muito simples de se efetuar um ataque em que todos os IPs disponíveis no *pool* de endereços para empréstimo sejam utilizados por um atacante, bastando fazer requisições a estes endereços através de técnicas de MAC spoofing.

Este problema perdura, mesmo quando se utiliza autenticação das mensagens DHCP como previsto em (Droms, 2001), pois as mensagens de requisição não são

[11] Este tipo de ataque é muito utilizado contra servidores web, onde atacantes fazem múltiplas requisições de conexões e o servidor aloca recursos aguardando a confirmação destas conexões, confirmações que nunca chegam propositalmente. Com isto o servidor exaure todo o seu buffer de entrada para estabelecimento inicial de conexões

autenticadas e a segunda mensagem que parte do servidor para o cliente já contém uma proposta de IP, este que foi retirado do *pool* de IPs disponíveis.

Basta um atacante fazer várias requisições com endereços MAC diferentes (e possivelmente nomes de usuários diferentes) e não responder às ofertas do servidor (conexões meio-abertas) de forma a esgotar todos os endereços disponíveis no *pool* e desta forma negar aos usuários válidos a possibilidade de adquirir um endereço IP para uso dos recursos da rede.

Como forma de prevenção a estes ataques devem-se utilizar temporizadores que encerrarão conexões meio-abertas após um tempo determinado. Este tipo de ataque deve ser observado quando da implementação dos protocolos de autenticação, como os que são propostos nesta dissertação através da utilização de temporizadores nos *sockets* e captura em nível de enlace (Libpcap, 2005).

Apêndice C

Criação dos pacotes a partir do CVS

```
!#/bin/bash

echo "Which system do you want to package [DHCP | TLS | CR | SRP]? "
read sysSelec

if [ "$sysSelec" != DHCP ] && [ "$sysSelec" != TLS ] &&
[ "$sysSelec" != CR ] &&
[ "$sysSelec" != SRP ]; then
echo "Invalid system!"
exit -1
fi

# Give the user some reference
pwd
ls -ld

echo "You must provide the passwords to access the CVS repository!"
echo "Press Ctrl-C to cancel or any other key to continue ..."
read k
```

```
# Delete previous directory installation
rm -rf AirStrike${sysSelec}
rm -rf AirStrike
rm -rf unpv12e
rm -rf AirStrike${sysSelec}.tar.gz

# Get the source code
echo "Grab unpv12e from CVS Repository ..."
cvs -q export -D tomorrow -d unpv12e unpv12e

echo "Grab AirStrike${sysSelec} from CVS Repository ..."
cvs -q export -D tomorrow -d AirStrike AirStrike${sysSelec}

# Steven's library shoul be under AirStrike directory for the sake
# of simplicity
mv unpv12e AirStrike/

# Change unp.h and libunp.a references in Makefile and common.h
cd AirStrike
cat Makefile | sed 's/\.\.\/unpv12e\/libunp.a/unpv12e\/libunp.a/' \\
> Makefile.tmp
mv Makefile.tmp Makefile

cat common.h | sed 's/\.\.\/unpv12e/unpv12e/' > common.h.tmp
mv common.h.tmp common.h

# Only necessary files must be packaged
make clean
cd unpv12e/lib
make clean
cd ../libfree
make clean
```

```
# Return to the 'script' directory and package it
cd ../../../

mv AirStrike AirStrike${sysSelec}
tar czvf AirStrike${sysSelec}.tar.gz AirStrike${sysSelec}

# Clean up
rm -rf AirStrike${sysSelec}

echo; echo "AirStrike${sysSelec} was packaged!"

cp AirStrike${sysSelec}.tar.gz /usr/local/apache2/htdocs

echo "AirStrike${sysSelec}.tar.gz copied to the webserver"
echo "on 10.10.0.10!"
```

Apêndice D

Varredura de segurança utilizando Flawfinder

Nenhum dos sistemas implementados apresentou falhas de segurança reportadas pelo Flawfinder como pode ser visto abaixo.

A saída completa retornada pelo Flawfinder pode ser vista no CD anexo, pois abaixo há apenas algumas das telas retornadas.

D.1 Varredura do Strikein-CR

```
Flawfinder version 1.25, (C) 2001-2004 David A. Wheeler.
Number of dangerous functions in C/C++ ruleset: 158
Examining ./authKEP.c
Examining ./common.h
Examining ./convert.c
Examining ./crClient.c
Examining ./credentials.c
Examining ./credentials.h
Examining ./isAlive.c
Examining ./isAlive.h
```

```
Examining ./processPayload.c
Examining ./processPayload.h
Examining ./strikeCrypt.c
Examining ./strikeCrypt.h
Examining ./strikeinCRClient.c
Examining ./strikeinCRClientGUI.c
Examining ./strikeinCRServer.c
./authKEP.c:41:  [2] (buffer) char:
  Statically-sized arrays can be overflowed. Perform bounds checking,
  use functions that limit length, or ensure that the size is larger
  than the maximum possible length.
./authKEP.c:63:  [2] (buffer) memcpy:
  Does not check for buffer overflows when copying to destination.
    Make sure destination can always hold the source data.

... Saída truncada neste ponto ...

Hits = 98
Lines analyzed = 2304 in 0.84 seconds (6790 lines/second)
Physical Source Lines of Code (SLOC) = 1430
Hits@level = [0]   0 [1]  33 [2]  65 [3]   0 [4]   0 [5]   0
Hits@level+ = [0+]  98 [1+]  98 [2+]  65 [3+]   0 [4+]   0 [5+]   0
Hits/KSLOC@level+ = [0+] 68.5315 [1+] 68.5315 [2+] 45.4545 [3+]   0
[4+]   0 [5+]   0
Minimum risk level = 1
Not every hit is necessarily a security vulnerability.
There may be other security vulnerabilities; review your code!
```

D.2 Varredura do Strikein-TLS

```
Flawfinder version 1.25, (C) 2001-2004 David A. Wheeler.
```

```
Number of dangerous functions in C/C++ ruleset: 158
Examining ./common.h
Examining ./initializeCtx.c
Examining ./initializeCtx.h
Examining ./isAlive.c
Examining ./isAlive.h
Examining ./sigchldwaitpid.c
Examining ./sigchldwaitpid.h
Examining ./strikeCrypt.c
Examining ./strikeCrypt.h
Examining ./strikeinTLSClient.c
Examining ./strikeinTLSClientGUI.c
Examining ./strikeinTLSServer.c
Examining ./tlsClient.c
./common.h:69:  [2] (buffer) char:
  Statically-sized arrays can be overflowed. Perform bounds checking,
  use functions that limit length, or ensure that the size is larger than
  the maximum possible length.
./common.h:73:  [2] (buffer) char:
  Statically-sized arrays can be overflowed. Perform bounds checking,
  use functions that limit length, or ensure that the size is larger than
  the maximum possible length.

... Saída truncada neste ponto ...

Hits = 33
Lines analyzed = 1212 in 0.69 seconds (6388 lines/second)
Physical Source Lines of Code (SLOC) = 737
Hits@level = [0]   0 [1]   3 [2]  30 [3]   0 [4]   0 [5]   0
Hits@level+ = [0+]  33 [1+]  33 [2+]  30 [3+]   0 [4+]   0 [5+]   0
Hits/KSLOC@level+ = [0+] 44.7761 [1+] 44.7761 [2+] 40.7056 [3+]   0 [4+]
 0 [5+]   0
```

```
Minimum risk level = 1
Not every hit is necessarily a security vulnerability.
There may be other security vulnerabilities; review your code!
```

D.3 Varredura do Strikein-SRP

```
Flawfinder version 1.25, (C) 2001-2004 David A. Wheeler.
Number of dangerous functions in C/C++ ruleset: 158
Examining ./authKEPClient.c
Examining ./authKEPServer.c
Examining ./common.h
Examining ./srpClient.c
Examining ./strikeinSRPClient.c
Examining ./strikeinSRPClientGUI.c
Examining ./strikeinSRPServer.c
Examining ./verifyVersionAndContext.c
Examining ./unpv12e/lib/addrinfo.h
Examining ./unpv12e/lib/connect_nonb.c
Examining ./unpv12e/lib/connect_timeo.c
Examining ./unpv12e/lib/daemon_inetd.c
Examining ./unpv12e/lib/daemon_init.c
Examining ./unpv12e/lib/dg_cli.c
Examining ./unpv12e/lib/dg_echo.c
Examining ./unpv12e/lib/error.c
Examining ./unpv12e/lib/get_ifi_info.c
Examining ./unpv12e/lib/gf_time.c
Examining ./unpv12e/lib/host_serv.c
Examining ./unpv12e/lib/hstrerror.c
Examining ./unpv12e/lib/if_indextoname.c
Examining ./unpv12e/lib/if_nameindex.c
Examining ./unpv12e/lib/if_nametoindex.c
```

```
Examining ./unpv12e/lib/in6addr_any.c
Examining ./unpv12e/lib/isfdtype.c
Examining ./unpv12e/lib/mcast_get_loop.c
Examining ./unpv12e/lib/mcast_join.c
Examining ./unpv12e/lib/mcast_set_if.c
Examining ./unpv12e/lib/mcast_set_ttl.c
Examining ./unpv12e/lib/my_addrs.c
Examining ./unpv12e/lib/pselect.c
Examining ./unpv12e/lib/read_fd.c
Examining ./unpv12e/lib/readable_timeo.c
Examining ./unpv12e/lib/readline.c
Examining ./unpv12e/lib/readn.c
Examining ./unpv12e/lib/rtt.c
Examining ./unpv12e/lib/signal.c
Examining ./unpv12e/lib/signal_intr.c
Examining ./unpv12e/lib/snprintf.c
Examining ./unpv12e/lib/sock_bind_wild.c
Examining ./unpv12e/lib/sock_cmp_addr.c
Examining ./unpv12e/lib/sock_cmp_port.c
Examining ./unpv12e/lib/sock_get_port.c
Examining ./unpv12e/lib/sock_ntop.c
Examining ./unpv12e/lib/sock_ntop_host.c
Examining ./unpv12e/lib/sock_set_addr.c
Examining ./unpv12e/lib/sock_set_port.c
Examining ./unpv12e/lib/sock_set_wild.c
Examining ./unpv12e/lib/sockatmark.c
Examining ./unpv12e/lib/sockfd_to_family.c
Examining ./unpv12e/lib/str_cli.c
Examining ./unpv12e/lib/str_echo.c
Examining ./unpv12e/lib/tcp_connect.c
Examining ./unpv12e/lib/tcp_listen.c
Examining ./unpv12e/lib/tv_sub.c
```

```
Examining ./unpv12e/lib/udp_client.c
Examining ./unpv12e/lib/udp_connect.c
Examining ./unpv12e/lib/udp_server.c
Examining ./unpv12e/lib/unp.h
Examining ./unpv12e/lib/unpifi.h
Examining ./unpv12e/lib/unprtt.h
Examining ./unpv12e/lib/unpthread.h
Examining ./unpv12e/lib/wraplib.c
Examining ./unpv12e/lib/wrappthread.c
Examining ./unpv12e/lib/wrapsock.c
Examining ./unpv12e/lib/wrapstdio.c
Examining ./unpv12e/lib/wrapunix.c
Examining ./unpv12e/lib/writable_timeo.c
Examining ./unpv12e/lib/write_fd.c
Examining ./unpv12e/lib/writen.c
Examining ./unpv12e/libfree/in_cksum.c
Examining ./unpv12e/libfree/inet_aton.c
Examining ./unpv12e/libfree/inet_ntop.c
Examining ./unpv12e/libfree/inet_ntop_ipv4.c
Examining ./unpv12e/libfree/inet_pton.c
Examining ./unpv12e/libfree/inet_pton_ipv4.c
Examining ./isAlive.h
Examining ./isAlive.c
./strikeinSRPServer.c:218:  [4] (format) printf:
  If format strings can be influenced by an attacker, they can be
  exploited. Use a constant for the format specification.
./unpv12e/lib/error.c:92:  [4] (format) vsnprintf:
  If format strings can be influenced by an attacker, they can be
  exploited, and note that sprintf variations do not always \0-terminate. Use a
  constant for the format specification.

... Saída truncada neste ponto ...
```

```
Hits = 133
Lines analyzed = 5744 in 0.99 seconds (11719 lines/second)
Physical Source Lines of Code (SLOC) = 3939
Hits@level = [0]   0 [1]  31 [2]  86 [3]   2 [4]  14 [5]   0
Hits@level+ = [0+] 133 [1+] 133 [2+] 102 [3+]  16 [4+]  14 [5+]   0
Hits/KSLOC@level+ = [0+] 33.7649 [1+] 33.7649 [2+] 25.8949 [3+] 4.06194 [4+]
3.5542 [5+]   0
Minimum risk level = 1
Not every hit is necessarily a security vulnerability.
There may be other security vulnerabilities; review your code!
```

D.4 Varredura do Strikein-DHCP

```
Flawfinder version 1.25, (C) 2001-2004 David A. Wheeler.
Number of dangerous functions in C/C++ ruleset: 158
Examining ./authKEPClient.c
Examining ./authKEPServer.c
Examining ./callIsAliveAP.c
Examining ./callIsAliveStation.c
Examining ./cleanup.c
Examining ./common.h
Examining ./convert.c
Examining ./credentials.c
Examining ./dhcpClient.c
Examining ./dhcpServer.c
Examining ./isAlive.c
Examining ./isAlive.h
Examining ./parseDHCPConf.c
Examining ./pcap.c
Examining ./pcap_common.h
```

```
Examining ./processPayloadClient.c
Examining ./processPayloadServer.c
Examining ./udpread.c
./dhcpClient.c:50:  [4] (shell) system:
  This causes a new program to execute and is difficult to use safely.
  try using a library call that implements the same functionality if
  available.
./dhcpClient.c:54:  [4] (shell) system:
  This causes a new program to execute and is difficult to use safely.
  try using a library call that implements the same functionality if
  available.

... Saída truncada neste ponto ...

Hits = 169
Lines analyzed = 2349 in 0.83 seconds (7034 lines/second)
Physical Source Lines of Code (SLOC) = 1561
Hits@level = [0]   0 [1]  55 [2]  96 [3]   2 [4]  16 [5]   0
Hits@level+ = [0+] 169 [1+] 169 [2+] 114 [3+]  18 [4+]  16 [5+]   0
Hits/KSLOC@level+ = [0+] 108.264 [1+] 108.264 [2+] 73.0301 [3+] 11.5311 [4+]
 10.2498 [5+]   0
Minimum risk level = 1
Not every hit is necessarily a security vulnerability.
There may be other security vulnerabilities; review your code!
```

Apêndice E

Varredura de segurança utilizando BFB

Nenhum dos sistemas implementados apresentou falhas de segurança reportadas pelo BFB como pode ser visto abaixo.

E.1 Varredura do Strikein-CR

```
=> /home/carrion/codigoTese/AirStrikeCR/./startServer.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeCR/./startClient.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeCR/./configure
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
Cleaning up...might take a few seconds
```

E.2 Varredura do Strikein-TLS

```
=> /home/carrion/codigoTese/AirStrikeTLS/./startServer.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeTLS/./startClient.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeTLS/./configure
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
Cleaning up...might take a few seconds
```

E.3 Varredura do Strikein-SRP

```
=> /home/carrion/codigoTese/AirStrikeSRP/./startServer.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeSRP/./startClient.sh
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
=> /home/carrion/codigoTese/AirStrikeSRP/./configure
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
Cleaning up...might take a few secon
```

E.4 Varredura do Strikein-DHCP

```
=> /home/carrion/codigoTese/AirStrikeDHCP/./configure
   * Single argument testing
   * Multiple arguments testing
   * Environment variable testing
Cleaning up...might take a few seconds
```

Apêndice F

Header principal dos sistemas implementados

FO ram incluídos os arquivos common.h que é o header principal dos sistemas implementados, sendo eles o AirStrikeSRP, AirStrikeCR, AirStrikeDHCP e AirStrikeTLS, como referência rápida.

Estes arquivos contêm dados básicos das estruturas utilizadas na troca de mensagens e também parametrização geral de funcionamento dos protocolos, como por exemplo timeouts.

F.1 AriStrikeCR: Arquivo common.h

```
/*
 ************************************************************
 Authentication based on a challenge-response protocol

 Filename: common.h

 Description:
```

```
 This header has definitions, headers and structures used
 throughout the system.

 Author: Demetrio Carrion         31-Mar-2004
 Ravel Laboratory                 PESC/COPPE/UFRJ
 Rio de Janeiro, Brazil
 ************************************************************
 */

#include "unpv12e/lib/unp.h"
#include <mysql/mysql.h>

#define DEBUG 1
#define LINUX

#define SERVER_PORT 7781

/* Variables sizes */
#define PASSWD_SIZE 16+1
#define USERNAME_SIZE 16+1
#define MAX_HASH_SIZE 20
#define MAX_PAYLOAD_SIZE 255
#define NONCE_SIZE 32
#define DIRECTIVE_VALUE_SIZE 40
#define HEADER_SIZE 4

/* Status codes */
#define CREDENTIALS_NOT_ACCEPTED 0
#define CREDENTIALS_ACCEPTED 1
#define CONNECTION_ERROR 2
#define OK 3
#define ERROR 4
```

```
/* Protocol messages */
#define HELLO_CLIENT 10
#define HELLO_SERVER 11
#define SEND_CREDENTIALS 12
#define SERVER_FINISH 13
#define ALARM_VERSION_MISMATCH 50
#define ALARM_OUT_OF_CTX 51
#define THE_END 99

/*
 * Every message represents a diferent context.
 * Some messages have a fixe payload. Like those define here.
 */
#define PAYLOAD_HELLO_CLIENT_SIZE USERNAME_SIZE+MAX_HASH_SIZE

#ifdef OPENBSD
#define FILE_OPEN_FAIL NULL
#define DEV_RANDOM "/dev/srandom"
#elif defined LINUX
#define FILE_OPEN_FAIL -1
#define DEV_RANDOM "/dev/random"
#endif

/* Global variables */
MYSQL mysql;

struct payloadAlarm {
unsigned char alarmLevel;
unsigned char error;
};
```

```
struct payloadHelloClient {
char username[USERNAME_SIZE];
};

struct payloadHelloServer {
unsigned char nonceServer[NONCE_SIZE];
};

struct payloadSendCredentials {
unsigned char nonceClient[NONCE_SIZE];
unsigned char hashedCredentials[MAX_HASH_SIZE];
};

struct payloadServerFinish {
        unsigned char credentialStatus;
unsigned char hashedAck[MAX_HASH_SIZE];
};

union packetPayload {
struct payloadHelloClient helloClient;
struct payloadHelloServer helloServer;
struct payloadSendCredentials sendCredentials;
struct payloadServerFinish serverFinish;
};

struct protocolVersion {
unsigned char major;
unsigned char minor;
};

struct strikeInPacket {
struct protocolVersion version;
```

```
unsigned char context;
unsigned char length;
union packetPayload payload;
};

struct authVerifiers {
char username[USERNAME_SIZE];
unsigned char hashedPasswd[MAX_HASH_SIZE];
unsigned char nonceClient[NONCE_SIZE];
unsigned char nonceServer[NONCE_SIZE];
unsigned char credentialStatus;
};

struct strikeInConf {
        char user[DIRECTIVE_VALUE_SIZE];
        char password[DIRECTIVE_VALUE_SIZE];
        char host[DIRECTIVE_VALUE_SIZE];
        char database[DIRECTIVE_VALUE_SIZE];
};
```

F.2 AriStrikeTLS: Arquivo common.h

```
/*
 ************************************************************
 Authentication based on a TLS protocol

 Filename: common.h

 Description:

 This header has definitions, headers and structures used
```

```
 throughout the system.

 Author: Demetrio Carrion         31-Mar-2004
 Ravel Laboratory                 PESC/COPPE/UFRJ
 Rio de Janeiro, Brazil
 ************************************************************
 */

#define DEBUG 1

#define LINUX

#define SERVER_PORT 7778
#define PASSWD_SIZE 16+1
#define USERNAME_SIZE 16+1
#define MAX_HASH_SIZE 20
#define MAX_PAYLOAD_SIZE 255
#define NONCE_SIZE 32
#define DIRECTIVE_VALUE_SIZE 40
#define CERT_FILE_NAME_SIZE 100
#define PRIV_KEY_FILE_NAME_SIZE 100

#define CREDENTIALS_NOT_ACCEPTED 0
#define CREDENTIALS_ACCEPTED 1
#define CONNECTION_ERROR 2
#define SSL_ERROR 3
#define GENERAL_ERROR 4
#define OK 10

// Every packet has a fixed header
#define HEADER_SIZE 4
```

```
// Protocol messages
#define HELLO_CLIENT 10
#define HELLO_SERVER 11
#define SEND_CREDENTIALS 12
#define SERVER_FINISH 13
#define ALARM_VERSION_MISMATCH 50
#define ALARM_OUT_OF_CTX 51
#define THE_END 99

// Every message represents a diferent context.
// Some messages have a fixe payload. Like those define here.
#define PAYLOAD_HELLO_CLIENT_SIZE USERNAME_SIZE+MAX_HASH_SIZE

#ifdef OPENBSD
#define FILE_OPEN_FAIL NULL
#define DEV_RANDOM "/dev/srandom"
#elif defined LINUX
#define FILE_OPEN_FAIL -1
#define DEV_RANDOM "/dev/random"
#endif

struct payloadAlarm {
unsigned char alarmLevel;
unsigned char error;
};

struct payloadHelloClient {
char username[USERNAME_SIZE];
};

struct payloadHelloServer {
unsigned char nonceServer[NONCE_SIZE];
```

```
};

struct payloadSendCredentials {
unsigned char nonceClient[NONCE_SIZE];
unsigned char hashedCredentials[MAX_HASH_SIZE];
};

struct payloadServerFinish {
        unsigned char credentialStatus;
unsigned char hashedAck[MAX_HASH_SIZE];
};

union packetPayload {
struct payloadHelloClient helloClient;
struct payloadHelloServer helloServer;
struct payloadSendCredentials sendCredentials;
struct payloadServerFinish serverFinish;
};

struct protocolVersion {
unsigned char major;
unsigned char minor;
};

struct strikeInPacket {
struct protocolVersion version;
unsigned char context;
unsigned char length;
union packetPayload payload;
};

struct authVerifyers {
```

```
char username[USERNAME_SIZE];
unsigned char hashedPasswd[MAX_HASH_SIZE];
unsigned char nonceClient[NONCE_SIZE];
unsigned char nonceServer[NONCE_SIZE];
unsigned char credentialStatus;
};

struct strikeInConf {
        char user[DIRECTIVE_VALUE_SIZE];
        char password[DIRECTIVE_VALUE_SIZE];
        char host[DIRECTIVE_VALUE_SIZE];
        char database[DIRECTIVE_VALUE_SIZE];
};
```

F.3 AriStrikeSRP: Arquivo common.h

```
#define LINUX

/*
 ************************************************************
 Authentication based on the SRP protocol

 Filename: common.h

 Description:

 This header has definitions, headers and structures used
 throughout the system.

 Author: Demetrio Carrion         31-Mar-2004
```

```
 Ravel Laboratory                  PESC/COPPE/UFRJ
 Rio de Janeiro, Brazil
 ************************************************************
 */

#include "unpv12e/lib/unp.h"
#include </usr/local/include/srp.h>

#ifndef NET_RT_IFLIST
#define NET_RT_IFLIST 3
#endif

#define DEBUG 1

/* DEBUG MODES */
#define DEBUG_NO 0
#define DEBUG_VERBOSE 1
#define DEBUG_VERY_VERBOSE 2

/* Variables sizes */
#define SERVER_PORT 7773
#define PASSWD_SIZE 16+1
#define USERNAME_SIZE 16+1
#define MAX_PAYLOAD_SIZE 65535
#define SHA_SIZE 20
#define DEV_SIZE 5
#define NONCE_SIZE 16
#define SESSION_KEY_SIZE 40

#define VERSION_MAJOR 1
#define VERSION_MINOR 0
```

```
/* Status codes */
#define CREDENTIALS_NOT_ACCEPTED 0
#define CREDENTIALS_ACCEPTED 1
#define CONNECTION_ERROR 2
#define OK 3
#define ERROR 4
#define ERROR_CATCH_SIGNAL 5

/* Every packet has a fixed header */
#define HEADER_SIZE 4
#define SUB_HEADER_PARAMS_SIZE sizeof(int)*3

/* Protocol messages */
#define SRP_USERNAME 20
#define SRP_PARAMETERS 21
#define SRP_PUB_A 22
#define SRP_PUB_B 23
#define SRP_RESP_CLIENT 24
#define SRP_RESP_SERVER 25
#define ALARM_VERSION_MISMATCH 50
#define ALARM_OUT_OF_CTX 51
#define THE_END 99

/* Global variables */
unsigned char debugMode;

struct payloadSrpUsername {
char username[USERNAME_SIZE];
};

struct payloadSrpParameters {
int modulusLen;
```

```
int generatorLen;
int saltLen;
unsigned char modulus[128];
unsigned char generator[1];
unsigned char salt[10];
};

struct payloadSrpPub {
unsigned char pubSessionValue[128];
};

struct payloadSrpResp {
unsigned char respSessionValue[SHA_SIZE];
};

union packetPayload {
struct payloadSrpUsername srpUsername;
struct payloadSrpParameters srpParameters;
struct payloadSrpPub srpPubA;
struct payloadSrpPub srpPubB;
struct payloadSrpResp srpResp;
};

struct protocolVersion {
unsigned char major;
unsigned char minor;
};

struct strikeInPacket {
struct protocolVersion version;
unsigned char context;
unsigned char length;
```

```
union packetPayload payload;
};
```

F.4 AriStrikeDHCP: Arquivo common.h

```
#define LINUX
/*
 ************************************************************
 Authentication based on the DHCP \& Challenge-Response protocol

 Filename: common.h

 Description:

 This header has definitions, headers and structures used
 throughout the system.

 Author: Demetrio Carrion         31-Mar-2004
 Ravel Laboratory                 PESC/COPPE/UFRJ
 Rio de Janeiro, Brazil
 ************************************************************
 */

/* See /usr/include/netinet/udp.h - It's necessay for Linux */

#include        "unpv12e/lib/unp.h"
#include  "unpv12e/lib/unpthread.h"
#include <pcap.h>

#include        <netinet/in_systm.h>    /* required for ip.h */
#include        <netinet/in.h>
```

```
#include        <netinet/ip.h>
#include        <netinet/udp.h>
#include        <net/if.h>
#include        <netinet/if_ether.h>

#ifndef NET_RT_IFLIST
#define NET_RT_IFLIST 3
#endif

/* Variables sizes */
#define SERVER_PORT 7777
#define PASSWD_SIZE 16
#define USERNAME_SIZE 16
#define XID_SIZE 12
#define SHA_SIZE 20
#define NONCE_SIZE 16
#define DEV_SIZE 5
#define IP_ADDR_SIZE 15+1
#define DOMAIN_SIZE 44
#define SESSION_KEY_SIZE 16
#define SESSION_KEY_ENC_SIZE 24
#define DIRECTIVE_VALUE_SIZE 40

/* Protocol version */
#define VERSION_MAJOR '1'
#define VERSION_MINOR '0'
#define PATCH_LEVEL '0'

/* Operations */
#define CTX_DISCOVER 20
#define CTX_CHALLENGE 21
#define CTX_RESPONSE 22
```

```
#define CTX_OFFER 23
#define CTX_THE_END 99

/* Status codes */
#define CREDENTIALS_NOT_ACCEPTED 0
#define CREDENTIALS_ACCEPTED 1
#define CONNECTION_ERROR 2
#define OK 3
#define ERROR 4
#define ERROR_CATCH_SIGNAL 5

/* DEBUG MODES */
#define DEBUG_NO 0
#define DEBUG_VERBOSE 1
#define DEBUG_VERY_VERBOSE 2

#define SNIFFER_TIMEOUT 1
#define SNIFFER_TRIES 3
#define IP_LEASED 1
#define IP_NOT_LEASED 0
#define STRIKEIN_CONF "/usr/local/AirStrike/DHCP/conf/strikein.conf"
#define DHCP_CONF "/usr/local/AirStrike/DHCP/conf/dhcp.conf"

struct protocolVersion {
        unsigned char major;
        unsigned char minor;
unsigned char patchLevel;
};

struct dhcpPacket {
struct protocolVersion version;
unsigned char operation;
```

```
unsigned char xid[XID_SIZE];
unsigned char username[USERNAME_SIZE];
unsigned char nonce[NONCE_SIZE];
unsigned char packetHash[SHA_SIZE];
unsigned char yourIP[IP_ADDR_SIZE];
unsigned char yourNetmask[IP_ADDR_SIZE];
unsigned char yourBroadcast[IP_ADDR_SIZE];
unsigned char yourGW[IP_ADDR_SIZE];
unsigned char yourDNS[IP_ADDR_SIZE];
unsigned char yourDomain[DOMAIN_SIZE];
unsigned char sessionKeyEnc[SESSION_KEY_ENC_SIZE];
};

#define DHCP_PACKET_SIZE sizeof(struct dhcpPacket)
/* The first 3 messages work with a small dhcp packet. */
#define DHCP_SMALL_PACKET_SIZE (4+XID_SIZE+USERNAME_SIZE+NONCE_SIZE+SHA_SIZE)

/* Global variables */
unsigned char debugMode;
char device[DEV_SIZE];
int datalink;
pcap_t *pd;
int *leasedIPS;
int addrInPool;

struct authVerifiers {
        char username[USERNAME_SIZE];
        unsigned char hashedPasswd[SHA_SIZE];
        unsigned char nonceClient[NONCE_SIZE];
        unsigned char nonceServer[NONCE_SIZE];
unsigned char xid[XID_SIZE];
unsigned char sessionKey[SESSION_KEY_ENC_SIZE];
```

```
};

struct strikeInConf {
        char user[DIRECTIVE_VALUE_SIZE];
        char password[DIRECTIVE_VALUE_SIZE];
        char host[DIRECTIVE_VALUE_SIZE];
        char database[DIRECTIVE_VALUE_SIZE];
};

/* top - bottom : Range of IPs to be leased */
struct dhcpConfInfo {
unsigned char netmask[IP_ADDR_SIZE];
unsigned char gateway[IP_ADDR_SIZE];
unsigned char dns[IP_ADDR_SIZE];
unsigned char broadcast[IP_ADDR_SIZE];
unsigned char domain[DOMAIN_SIZE];
unsigned char top[IP_ADDR_SIZE];
unsigned char bottom[IP_ADDR_SIZE];
};

struct threadParameters {
struct sockaddr_in cliaddr;
socklen_t cliaddrLen;
char mesg[DHCP_PACKET_SIZE+1];
};

/* Function prototypes */
void cleanup(int);
void udp_write(char *, int);
int udp_read(struct dhcpPacket *);
int udp_check(char *ptr, int len);
```