UNIVERSIDADE SAGRADO CORAÇÃO

DEIVISON CARDOSO

CRIPTOANÁLISE EM REDES SEM FIO UTILIZANDO *LOOKUP TABLES*

BAURU
2013

DEIVISON CARDOSO

CRIPTOANÁLISE EM REDES SEM FIO UTILIZANDO *LOOKUP TABLES*

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Sociais Aplicadas como parte dos requisitos para obtenção do Título de Bacharel em Ciência da Computação, sob orientação do Prof. Esp. Henrique Pachioni Martins.

BAURU
2013

Cardoso, Deivison
C2684c

Criptoanálise em redes sem fio utilizando Lookup Table / Deivison Cardoso -- 2013.
78f. : il.

Orientador: Prof. Esp. Henrique Pachioni Martins.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Universidade do Sagrado Coração – Bauru – SP.

1. Redes sem fio. 2. Segurança. 3. Criptografia. 4, Criptoanálise. 5. Lookup Tables. I. Martins, Henrique Pachioni. II. Título.

DEIVISON CARDOSO

CRIPTOANÁLISE EM REDES SEM FIO UTILIZANDO *LOOKUP TABLES*

Projeto de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Sociais Aplicadas como parte dos requisitos para obtenção do Título de Bacharel em Ciência da Computação, sob orientação do Prof. Esp. Henrique Pachioni Martins.

Banca examinadora:

_____
Prof. Esp. Henrique Pachioni Martins
Universidade Sagrado Coração

_____
Prof. Me. Patrick Pedreira Silva
Universidade Sagrado Coração

_____
Prof.Dr. Élvio Gilberto da Silva
Universidade Sagrado Coração

Bauru, 2 de dezembro de 2013.

RESUMO

Com o avanço das tecnologias e crescimento exponencial das redes de computadores, tornou-se comum a utilização de radiofrequências como meio de transmissão de dados com as chamadas redes sem fio e a criptografia possui um papel importante neste ambiente, evitando que pessoas que não fazem parte de uma comunicação de dados possam ler o que está sendo trafegado. Apesar de tantas opções de criptografia apresentadas pelo mercado, este trabalho apresenta como quebrar algumas destas barreiras de segurança utilizando *Lookup Tables* e dicionários, comparando os resultados e apresentando formas de se proteger de ataques como estes citados.

Palavras Chave: Redes sem fio, Segurança, Criptografia, Criptoanálise, *Lookup Tables*.

ABSTRACT

With the advancement of technologies and the exponential growth of computer networks, it has become common to use radio as a means of data transmission with the known wireless networks and encryption has an important role by preventing people who are not part of a data communication can read what is being trafficked. Although many encryption options presented by the market, this paper presents how to break some of these security barriers using Lookup Tables and dictionaries, comparing the results and presenting ways to protect yourself from attacks like these mentioned.

Keywords: Wireless Networking, Security, Cryptography, Cryptanalysis, Lookup Tables.

LISTA DE FIGURAS

Figura 1 - Modelo OSI 17
Figura 2 - Modelo TCP/IP 18
Figura 3 - Topologia Ponto-a-Ponto e Barramento 20
Figura 4 - Topologia Anel e Estrela 21
Figura 5 - Independent Basic Service Set e Basic Service Set 22
Figura 6 - Extended Service Set 22
Figura 7 - Subcamadas MAC no padrão IEEE 802.11 24
Figura 8 - Fluxograma para o CSMA/CA 25
Figura 9 - Criptografia Simétrica 34
Figura 10 - Rodada individual do algoritmo DES 36
Figura 11 - Configuração AES 37
Figura 12 - Transformação AddRoundKey 37
Figura 13 - Transformação SubBytes 38
Figura 14 - Transformação ShiftRow 39
Figura 15 - Criptografia Assimétrica 40
Figura 16 - Método Diffie-Hellman 41
Figura 17 - Criptografia RSA: e=5, n=35 43
Figura 18 - Decriptação RSA: d=29, n=35 43
Figura 19 – MAC 44
Figura 20 - (a) Uma mensagem preenchida até um múltiplo de 512 *bits*. (b) As variáveis de saída. (c) O array de palavras 47
Figura 21 - Código-fonte utilizado para a geração das senhas 53
Figura 22 - Saída gerada pelo software 54
Figura 23 - *Script* utilizado para a geração dos arquivos de dicionário 55
Figura 24 - *Script* utilizado para a geração das *Lookup Tables* 55
Figura 25 - Airmon-ng 56
Figura 26 - Airodump-ng sem filtros 57
Figura 27 - Saída apresentada pelo Aireplay-ng 58
Figura 28 - Airodump-ng com a mensagem de captura de *handshake* 59
Figura 29 - Estrutura de arquivos 59
Figura 30 - Trecho do *script* utilizado para o ataque utilizando *Lookup Table* .. 60
Figura 31 - *Script* para juntar todos arquivos de dicionário em apenas um 61

Figura 32 - *Script* para realização de cinco ataques de dicionário seguidos.... 61
Figura 33 - Médias coletadas dos ataques utilizando Lookup Table 62
Figura 34 - Médias coletadas dos ataques utilizando Dicionário..... 62
Figura 35 - Configuração de segurança da rede sem fio 64
Figura 36 - Configuração de rede sem fio do roteador..... 65
Figura 37 - Representação de uma árvore para melhora de desempenho 67

LISTA DE ABREVIATURAS E SIGLAS

3DES	*Triple DES*
ABNT	Associação Brasileira De Normas Técnicas
ACK	*Acknowledgement*
AES	*Advanced Encryption Standard*
ANSI	*American National Standards Institute*
AP	*Access Point*
ARP	*Address Resolution Protocol*
ASCII	*American Standard Code For Information Interchange*
ASCII	*American Standard Code For Information Interchange*
BSA	*Basic Service Area*
BSS	*Basic Service Set*
BSSID	*Basic Service Set Identifcation*
CCMA	*Counter Cipher Mode With Block Chaining Message Authentication Code Protocol*
CERT	Centro De Estudo, Resposta E Tratamento De Incidentes De Segurança No Brasil
CSMA/CA	*Carrier Sense Multiple Access With Collision Avoidance*
CSMA/CD	*Carrier Sense Multiple Access With Collision Detection*
CTS	*Clear To Send*
DCF	*Distributed Coordination Function*
DES	*Data Encryption Standard*
DH	Diffie e Hellman
DIFS	*Distributed Interframe Space*
DNS	*Domain Name System*
EAP	*Extensible Authentication Protocol*
ESA	*Extended Service Area*
ESS	*Extended Service Set*
FTP	*File Transfer Protocol*
GNU	*Gnu Is Not Unix*
HMAC	*Keyed-Hashing For Message Authentication*
HTTP	*Hypertext Transfer Protocol*

IBSS	*Independent Basic Service Set*
IBSS	*Independent Basic Service Set*
ICMP	*Internet Control Message Protocol*
IEEE	*Institute Of Electrical And Electronics Engineers*
IGMP	*Internet Group Management Protocol*
IP	*Internet Protocol*
ISO	*International Organization For Standardization*
IV	*Inicialization Vector*
LAN	*Local Area Network*
LT	*LookupTables*
MAC1	*Medium Access Control*
MAC2	*Message Authentication Code*
MAN	*Metropolitan Area Network*
MD5	*Message Digest 5*
MDC	*Modification Detection Code*
MITM	*Man-in-the-Middle*
NAV	*Network Allocation Vector*
NIST	*National Institute Of Standards And Technology*
OSI	*Open System Interconnection*
PBKDF2	*Password-Based Key Derivation Functions Version 2*
PCF	*Point Coordnation Function*
PIFS	*PCF Interframe Space*
PSK	*Pre-Shared Key*
RAM	*Random Access Memory*
RARP	*Reverse Address Resolution Protocol*
RF	Radiofrequência
RSA	Rivest-Shamir-Adleman
RTS	*Request To Send*
SHA	*Secure Hash Algorithm*
SIFS	*Short Interframe Space*
SMTP	*Simple Mail Transfer Protocol*
SSID	*Service Set Identifier*
TCP	*Transmission Control Protocol*

TKIP	*Temporal Key Integrity Protocol*
UDP	*User Datagram Protocol*
VPN	*Virtual Private Network*
WAN	*Wide Area Network*
WEP	*Wired Equivalent Privacy*
WLAN	*Wireless Local Area Network*
WPA	*Wi-Fi Protected Access*
WPA2	*Wi-Fi Protected Access Versão 2*

SUMÁRIO

1 INTRODUÇÃO 12
2 OBJETIVOS 14
2.1 OBJETIVO GERAL 14
2.2 OBJETIVOS ESPECÍFICOS 14
3 REDES DE COMPUTADORES 14
4 PROTOCOLOS E PADRÕES 15
5 MODELOS DE REFERÊNCIA E DE PROTOCOLO 16
5.1 MODELO OSI 16
5.2 MODELO TCP/IP 18
6 REDES CABEADAS 20
7 REDES SEM FIO 21
7.1 ARQUITETURA 21
7.2 SERVIÇOS 23
7.3 SUBCAMADA MAC 23
7.3.1 DCF 24
7.3.2 PCF 26
8 MECANISMOS DE SEGURANÇA 27
8.1 WEP 27
8.2 WPA 28
8.3 WPA2 29
9 SEGURANÇA DA INFORMAÇÃO 29
9.1 PRINCÍPIOS 29
9.2 TIPOS DE ATAQUES 30
9.2.1 ATAQUES NÃO-ELETRÔNICOS 30
9.2.2 ATAQUES OFF-LINE 31
9.2.3 ATAQUES PASSIVOS 31
9.2.4 ATAQUES ATIVOS 31
9.3 SEGURANÇA FÍSICA 32
9.4 SEGURANÇA LÓGICA 33
10 CRIPTOGRAFIA 33
10.1 CRIPTOGRAFIA SIMÉTRICA 34
10.1.1 DES 35

10.1.2 3DES 36
10.1.3 AES 37
10.2 CRIPTOGRAFIA ASSIMÉTRICA 39
10.2.1 DH (DIFFIE, HELLMAN) 40
10.2.2 RSA 41
10.3 AUTENTICAÇÃO DE MENSAGEM 43
10.4 ALGORITMOS DE *HASH* E DE MAC 45
10.4.1 MD5 45
10.4.2 SHA-1 45
10.4.3 HMAC 47
10.4.4 PBKDF2 48
11 CRIPTOANÁLISE 48
12 LOOKUP TABLES 49
13 LINUX 49
13.1 BACKTRACK 50
13.1.1 CRUNCH 50
13.1.2 GENPMK 50
13.1.3 SUITE AIRCRACK-NG 51
13.1.3.1 AIRMON-NG 51
13.1.3.2 AIRODUMP-NG 51
13.1.3.3 AIREPLAY-NG 51
13.1.4 WIRESHARK 51
13.1.5 COWPATTY 52
14 METODOLOGIA 52
14.1 TIPO DE PESQUISA 52
14.2 RECURSOS 53
14.3 EXECUÇÃO 53
15 RESULTADOS 61
16 DIFICULDADES ENCONTRADAS 65
17 CONSIDERAÇÕES FINAIS 66
18 TRABALHOS FUTUROS 66
REFERÊNCIAS 68
ANEXO A – *SCRIPT* COMPLETO DE ATAQUE UTILIZANDO LOOKUP TABLE 71

ANEXO B - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO LOOKUP TABLE 76
ANEXO C - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO DICIONÁRIO 77

1 INTRODUÇÃO

Redes de computadores estão presentes hoje na maioria dos lugares. Seja nas casas, ambientes de trabalho ou na educação, as redes carregam consigo um papel tão importante que o seu mau funcionamento acaba gerando desordem na sociedade. Por definição, redes de computadores é a conexão de dois ou mais computadores com capacidade de trocar dados entre si. As redes mais primitivas eram capazes de apenas trocar informações de texto, de forma limitada, enquanto hoje em dia os dados trafegados na rede podem ser fluxos de voz, vídeos, textos e gráficos entre diversos sistemas computacionais – tais como celulares, *notebooks*, *tablets*, entre outros dispositivos – e não apenas entre computadores como antigamente.

Segundo informações publicadas no de 2013 pelo Centro de Estudo, Resposta e Tratamento de Incidentes de Segurança no Brasil – CERT.br (2013)– no ano de 2012 foram reportados ao centro 7815 invasões bem sucedidas em computadores e redes. Estes números tendem a subir enquanto os usuários não estejam cientes das vulnerabilidades que podem ser encontradas em suas redes.

Através destes números fica claro que a segurança da informação é um requisito básico para usuários em geral. A forma de segurança mais comum encontrada hoje são as senhas, conjunto de caracteres que tem por finalidade garantir acesso apenas ao proprietário da mesma. Porém, uma falha que se encontra em grande parte dos usuários é a forma como são criadas estas senhas, pois geralmente escolhem-se senhas simples para que possam lembrar futuramente, e esta simplicidade abre portas para pessoas sem autorização.

Dentro da área de segurança da informação, é trivial encontrarmos algum tópico relacionado à criptografia. De forma geral esta técnica se resume em escrever secretamente (do grego “*kryptos*”, ou seja, secreto e “*grapho*”, escrita), ou melhor, comunicar-se de maneira que nenhuma outra pessoa além do remetente e destinatário seja capaz de decifrar a mensagem. Com base neste conceito foram criados diversos algoritmos, fórmulas matemáticas para manipular a mensagem e deixá-la ilegível. Sendo assim, mesmo que esta mensagem seja interceptada por alguém, este não conseguirá ler a mensagem

original, a menos que este possua a informação que permite a encriptação e desencriptação da mensagem, também conhecida como chave.

Aprofundando ainda mais dentro da criptografia, podemos encontrar as chamadas funções *hash*, que resultam em uma mensagem única de tamanho fixo, independente do tamanho da informação original (considerado único porque apesar de teoricamente existir a possibilidade de dois *hashes* com resultados iguais a probabilidade de isto ocorrer é quase nula). Um detalhe importante deste resultado gerado, conhecido apenas como *hash*, é sua incapacidade de ser reprocessado inversamente gerando a informação original.

Esta característica aplicada às senhas implica em um crescimento de segurança a ponto de que uma senha interceptada, estará ilegível e de forma irreversível para o atacante, sendo em primeira instância inútil. Mas como não existe nada completamente seguro, *hackers*[1] surgem com novas técnicas a cada vulnerabilidade encontrada, algumas vezes para o bem, reportando o erro às empresas detentoras da programação, e às vezes para o benefício próprio, obtendo acesso indevido às redes, através de ataques às senhas criptografadas, ou simplesmente *hashes*.

Dentre vários tipos de ataques a senhas, um será destacado por ser o foco deste trabalho, o qual é denominado como ataque *off-line*, por ter a capacidade de ser executado na máquina local do atacante, sem qualquer tipo de conexão com o alvo. Diante desta situação este trabalho visa demonstrar as técnicas utilizadas para a quebra de senhas de redes sem fio, e formas de como se proteger destes ataques *off-line*, pois uma rede sem fio mal protegida é um alvo fácil para uma pessoa mal intencionada, e isto justifica este trabalho de forma que se consiga apresentar a sociedade, o risco que estas redes sem fio podem oferecer, e como é relativamente rápida a quebra de uma senha quando esta possui um nível de segurança alto.

[1] "[...] especialistas em computadores e tecnologia que estudam a fundos os sistemas operacionais, redes e protocolos de comunicação, seja para conhecê-los e melhorá-los" (ERICKSON, 2009)

2 OBJETIVOS

2.1 OBJETIVO GERAL

Realizar testes em laboratório com intuito de monitorar redes protegidas e demonstrar através de uma técnica *hacker,* como quebrar senhas criptografadas de redes sem fio utilizando *Lookup Tables* e apresentar formas de se proteger destes ataques.

2.2 OBJETIVOS ESPECÍFICOS

- Utilizar o sistema operacional *Backtrack5 R3* para a realização dos objetivos.
- Desenvolver um *software* para gerar combinações aleatórias de oito caracteres numéricos.
- Criar dicionário de palavras geradas por *software*.
- Criar *Lookup Tables* para serem usadas no ataque.
- Monitorar a rede e máquina alvo.
- Analisar pacotes e se necessário desassociar o usuário alvo de seu ponto de acesso.
- Utilizar *software*s para a quebra da senha com *Lookup Table* e com dicionário.
- Apresentar formas de como os usuários pode dificultar os ataques de Lookup Tables para a quebra de senha.

3 REDES DE COMPUTADORES

Nos dias atuais, graças ao avanço tecnológico, temos informações sobre diversos assuntos disponíveis praticamente de qualquer lugar. Esta disponibilidade deve-se à existência das redes de computadores que são capazes de trafegar informações de um dispositivo a outro. Existem algumas classificações para as redes, sendo as mais comumente utilizadas a *Local Area Network* (LAN), *Metropolitan Area Network* (MAN) e *Wide Area Network* (WAN).

As LANs, também conhecidas como redes locais, são redes que se delimitam a pequenos espaços físicos, com capacidade de compartilhamento de recursos e a troca de informações.

Já as MANs, ou simplesmente redes metropolitanas, tem uma abrangência maior, envolvendo uma cidade, como o nome sugere. Pouco conhecida e divulgada mas o seu melhor exemplo é a rede de televisão a cabo disponível em muitas cidades.

Por fim as WANs, ou redes geograficamente distribuídas, abrangem uma área geográfica grande de um país ou até mesmo continentes. De forma simples, trata-se de várias LANs interconectadas por uma sub-rede de roteadores que transmite as informações de uma rede local à outra.

Cada uma dessas classificações pode utilizar meios de transmissões guiados ou não-guiado. Stallings (2005) afirma que em ambos os meios de transmissão a comunicação está na forma de ondas eletromagnéticas, o que difere é que o meio de transmissão guiado as ondas são propagadas e guiadas por um meio sólido, enquanto no não-guiado são propagados pela atmosfera e não podem ser guiados.

4 PROTOCOLOS E PADRÕES

Para que as comunicações de dados nas redes sejam realizadas, é necessário que os dispositivos sigam uma série de protocolos. Forouzan (2008) cita que “protocolos são um conjunto de regras que controlam as comunicações de dados. Um protocolo define o que é comunicado, como isso é comunicado e quando deve ser comunicado”.

Complementando, a Cisco (2013) enfatiza a descrição das comunicações utilizando camadas, das quais possuem seus protocolos específicos, sendo as camadas inferiores relacionadas à movimentação de dados, e as camadas superiores são focadas no conteúdo da mensagem. Em geral, as camadas permitem dividir uma tarefa complexa em várias partes tornando-se simples a descrição de cada uma delas.

Em relação aos padrões, eles podem ser divididos em duas categorias: *de facto* e *de jure.* O padrão *de facto* (“de fato”, em português), segundo Tanenbaum (2003), “são aqueles que se consagram naturalmente, sem

nenhum plano formal", enquanto padrões *de jure* ("por lei" em português), "são padrões legais e formais adotados por uma instituição de padronização legalizada".

Forouzan (2008), também faz uma definição em sobre os padrões, sendo que estes

> "[...] são essenciais na criação e na manutenção de um mercado aberto e competitivo para fabricantes de equipamentos e na garantia de interoperabilidade nacional e internacional de dados e de tecnologia de telecomunicações e processos".

Os padrões são desenvolvidos por comitês, fóruns e orgãos governamentais e não-governamentais, dentre os mais conhecidos está uma organização voluntária independente entitulada como *International Organization for Standardization* (ISO), uma organização não-governamental conhecida como *American National Standards Institute* (ANSI), e a *Institute of Electrical and Electronics Engineers* (IEEE), a maior organização profissional do mundo.

5 MODELOS DE REFERÊNCIA E DE PROTOCOLO

Segundo a Cisco (2013), as diferenças entre um modelo de referência e um modelo de protocolo estão no conceito. O primeiro tem o objetivo de auxiliar o entendimento das funções e processos envolvidos, enquanto o segundo fornece um modelo que corresponde de perto à estrutura de um conjunto de protocolos. Os dois modelos de referência e de protocolo mais utilizados são respectivamente o modelo OSI (*Open System Interconnection*) e o modelo TCP/IP (*Transmission Control Protocol/Internet Protocol*).

5.1 MODELO OSI

O modelo OSI, criado pela ISO, é constituído por sete camadas, das quais cada camada recebe os dados de sua camada superior, incluem seus próprios dados, e enviam para a camada imediatamente inferior, repetindo o processo até chegar à camada mais inferior, onde o pacote com todos os dados estará pronto para ser transmitido (ULBRICH; VALLE, 2004). Como

mostra a Figura 1, as camadas do modelo OSI são: aplicação, apresentação, sessão, transporte, rede, enlace e física.

Figura 1 - Modelo OSI
Fonte: Elaborada pelo autor (2013)

Para cada camada existe uma função específica que deve ser executada segundo o modelo OSI. A Camada Física tem como papel principal comunicar cadeias de *bits*, assim como ativar, manter ou desativar as comunicações (ERICKSON, 2009).

Já a Camada de Enlace, de acordo com a Cisco (2013), tem a função de controlar como o dado é colocado e recebido do meio em que trafega usando técnicas de controle de acesso ao meio e detecção de erros. A Camada de Rede por sua vez utiliza quatro processos básicos:

- Endereçamento: endereço atribuído a um dispositivo.
- Encapsulamento: criação de um pacote contendo os dados a serem transferidos, endereço de origem e destino, entre outras informações.
- Roteamento: quando os dispositivos não estão presentes na mesma rede, dispositivos intermediários, chamados de roteadores são necessários. “O papel do roteador é selecionar o caminho e direcionar os pacotes a seus destinos” (CISCO, 2013).

Na Camada de Transporte é oferecido confiabilidade para as próximas camadas se utilizando de um transporte transparente de dados entre sistemas. A Camada de Sessão, segundo Erickson (2009) “é responsável por estabelecer e manter as conexões entre aplicativos de redes”.

De acordo com Forouzan (2008), a Camada de Apresentação é responsável por traduzir informações de um formato específico para um genérico para o receptor, criptografar e comprimir os dados, enquanto a Camada de Aplicação possibilita ao usuário acesso aos recursos da rede, oferecendo interface e suporte a diversos serviços.

5.2 MODELO TCP/IP

Neste modelo existem apenas quatro camadas: aplicação, transporte, inter-redes, *host/rede*, como é representado pela Figura 2.

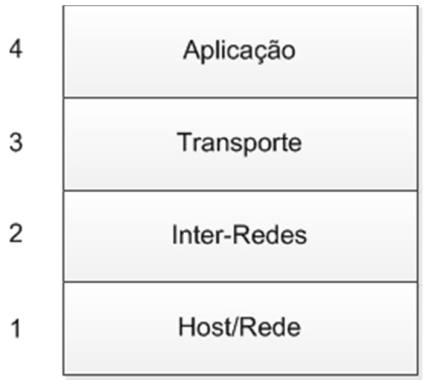

Figura 2 - Modelo TCP/IP
Fonte: Elaborada pelo autor (2013)

Na camada mais baixa, a *host/rede*, há uma falta de especificação, e livros e documentações que tratam do modelo TCP/IP raramente comentam sobre esta camada. No modelo a única especificação que existe é que “o *host* tem de se conectar a rede utilizando algum protocolo para que seja possível enviar pacotes IP” (TANENBAUM, 2003).

Na camada logo acima, denominada de interredes, ou apenas redes segundo alguns autores, é suportado o protocolo IP, o qual utiliza outros quatro protocolos auxiliares de suporte: ARP, RARP, ICMP e IGMP. Tais protocolos

não serão descritos em detalhes por não serem o foco deste trabalho. Esta camada possui a tarefa de entregar tais pacotes IP aonde forem necessários.

Forouzan (2008) afirma que o protocolo IP utiliza um serviço de entrada do tipo *best-effort* (melhor esforço possível), não necessita de conexão e não é confiável devido a falta de verificação e correção de erros. Este protocolo transporta dados divididos, e podem chegar ao destinatário fora de ordem. Porém estas limitações do IP não fazem dele um ponto negativo, mas sim o contrário, dando liberdade ao usuário de acrescentar funcionalidades necessárias alcançando uma máxima eficiência.

Na camada de transporte, dois protocolos são utilizados: o UDP – *User Datagram Protocol* – e o TCP. Tanenbaum (2003) afirma que o primeiro, assim como o IP, é um protocolo sem conexão e não confiável, onde seu objetivo está em uma entrega rápida, e não sobre uma entrega precisa, comumente utilizada em aplicações cliente/servidor com solicitação/resposta.

Já o TCP é um protocolo confiável e orientado à conexão, ou seja, em ambas as partes de uma transmissão é necessário estar conectado antes de qualquer ação tomada pelo protocolo. De modo geral, o TCP é transportado dentro de pacotes IP, em unidades denominadas segmentos, as quais possuem um número sequencial utilizado para reordenação quando o pacote é recebido pelo receptor, assim como um número de confirmação de segmentos recebidos (FOROUZAN, 2008).

Por fim, a camada de aplicação é equivalente a combinação das camadas de sessão, apresentação e aplicação do modelo de referência OSI. Vários protocolos podem ser utilizados nessa camada de nível mais alto, sendo alguns deles o protocolo de transferência de arquivos FTP (*File Transfer Protocol*), o protocolo de correio eletrônico SMTP (*Simple Mail Transfer Protocol*), protocolo DNS (*Domain Name System*), responsável por mapear nome de hosts para seus respectivos endereços de redes, e o HTTP (*Hypertext Transfer Protocol*) necessário para buscar páginas na *World Wide Web*.

6 REDES CABEADAS

Quando se trata de redes cabeadas, é necessário entendermos que há diversas formas de conectar cada nó da rede. Este *layout* lógico é chamado de topologia, sendo as quatro mais importantes a chamada ponto-a-ponto, barramento, anel e estrela. Cada uma dessas topologias, contam com características distintas que irão definir sua utilização ou não de acordo com a necessidade encontrada na rede.

De forma sucinta a topologia ponto-a-ponto conecta dois equipamentos em série tornando possível a comunicação entre eles, enquanto na topologia barramento é uma “versão estendida” da ponto-a-ponto por ser possível conectar diversos equipamentos em um único barramento, com uma característica importante que enquanto uma estação está mandando uma mensagem, as outras só podem escutar e receber, e caso algum equipamento durante o percurso esteja desligado, a informação não chegará até os próximos nós do barramento. Essas duas topologias citadas estão representadas na Figura 3.

Figura 3 - Topologia Ponto-a-Ponto e Barramento
Fonte: Elaborada pelo autor (2013)

As redes em anel, também conhecidas como *token-ring*, tiveram seu *layout* criado pela IBM e de acordo com Ulbrich e Valle (2004), estas redes ofereciam vantagens em relação às redes lineares anteriormente citadas, porém eram limitadas pelo número de equipamentos conectados na rede. Seu funcionamento utiliza o conceito de *token*, o qual fica circulando na rede enquanto seu conteúdo não for entregue ao remetente, e caso não tenha

conteúdo no mesmo, é possível que o dispositivo que possui o *token* carregá-lo com informações.

Por fim, na topologia estrela, toda informação deve ser passada por um equipamento central (*switch*[2], por exemplo) para que a informação seja enviada apenas para seu destinatário sem que as outras estações saibam da existência desta informação. Tanto a topologia anel quanto a estrela podem ser melhor visualizadas na representação da Figura 4.

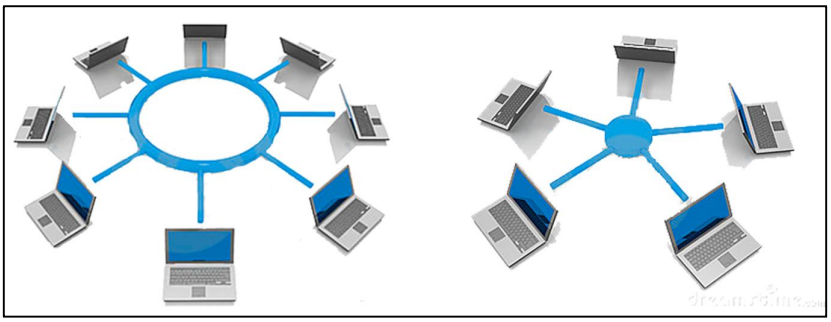

Figura 4 - Topologia Anel e Estrela
Fonte: Elaborada pelo autor (2013)

7 REDES SEM FIO

7.1 ARQUITETURA

Este tipo de rede, também conhecido como redes *wireless*, tem se tornado bastante presente nas casas dos usuários. De acordo com Forouzan (2008) o padrão para redes sem fio foi o IEEE 802.11 para que dispositivos possam ser acessados na ausência de uma estação-base, chamado de rede *ad-hoc*, conhecido também como IBSS (*Independent Basic Service Set*), ou na presença uma estação-base, podendo ser do tipo BSS (*Basic Service Set*) ou ESS (*Extended Service Set*).

Nas redes BSS é fornecida uma estrutura onde clientes móveis utilizam um único ponto de acesso. A área com cobertura de RF – radiofrequência – tanto para BSS, quanto para IBSS, é definida como *Basic Service Area* (BSA), ou ainda microcélula, representado na Figura 5.

[2] De acordo com Cisco (2013) o switch é um dispositivo de rede que filtra, encaminha e inunda quadros com base no endereço de destino de cada quadro e opera na camada de enlace de dados.

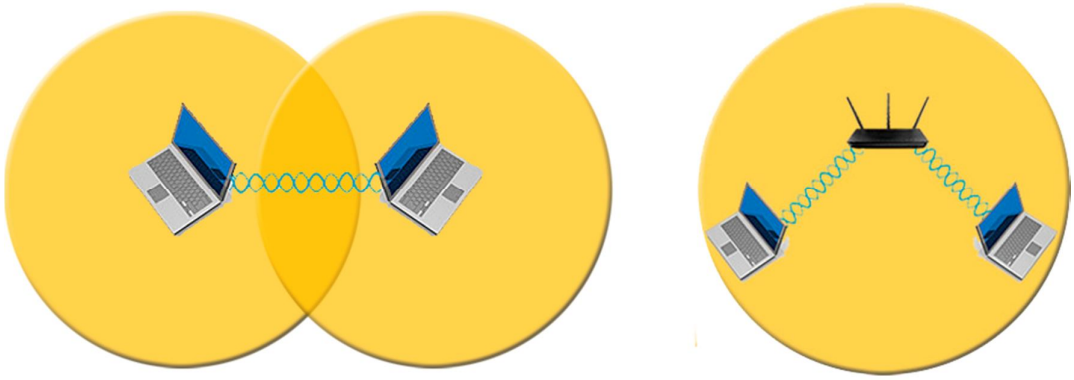

Figura 5 - Independent Basic Service Set e Basic Service Set
Fonte: Elaborada pelo autor (2013)

De acordo com a Cisco (2013), as redes ESS são necessárias quando um BSS não fornece uma cobertura suficiente de RF, sendo necessário unir um ou mais BSS por um sistema de distribuição comum, conforme mostra a Figura 6. Para diferenciar um BSS de outro dentro de um ESS, é verificado o identificador do BSS, chamado também de BSSID, que nada mais é que o endereço MAC (*Medium Access Control*) do *Access Point*, ou simplesmente AP, que serve o BSS. A área com cobertura de RF de uma ESS é conhecida como ESA – *Extended Service Area.*

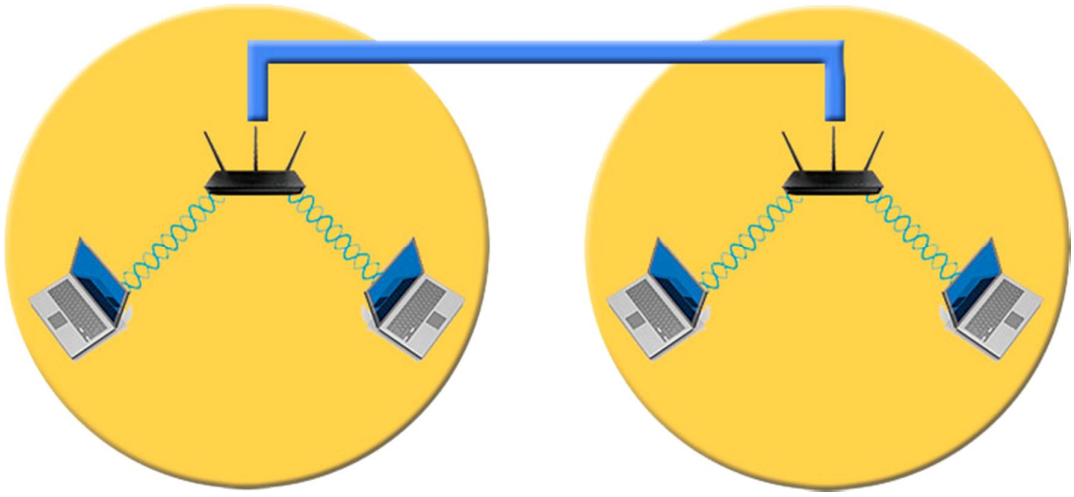

Figura 6 - Extended Service Set
Fonte: Elaborada pelo autor (2013)

7.2 SERVIÇOS

De acordo com Stallings (2005), a “IEEE 802.11 define diversos serviços que precisam ser fornecidos pela LAN sem fio para prover funcionalidade equivalente à que é inerente às LANs com fio”. Alguns destes serviços citados são:

- Associação: Antes de qualquer transmissão é necessário que a identidade da estação, assim como seu endereço seja conhecido. A partir do momento em que há uma associação da estação com o ponto de acesso, será possível comunicar-se com as outras estações da rede.
- Reassociação: Torna possível a transição da associação de um ponto de acesso já associado a outro, permitindo a mobilidade.
- Desassociação: Notificação de que uma associação entre um ponto de acesso e uma estação foi terminada.
- Autenticação: Estabelece identidade entre as estações ou pontos de acesso que querem se comunicar.
- Privacidade: Permite que apenas o destinatário pretendido possa ler o conteúdo de uma mensagem.

Assim como em redes cabeadas, as redes sem fio precisam tratar acessos múltiplos, para definir qual estação vai transmitir os dados sem que haja colisão de dados ou corromper os dados. Este tratamento é feito na subcamada MAC, que se encontra na camada de Enlace.

7.3 SUBCAMADA MAC

Dentro desta subcamada, se encontram dois protocolos de extrema importância em uma rede sem fio, sendo eles o DCF (*Distributed Coordination Function*) e o PCF (*Point Coordnation Function*), conforme demonstra a Figura 7.

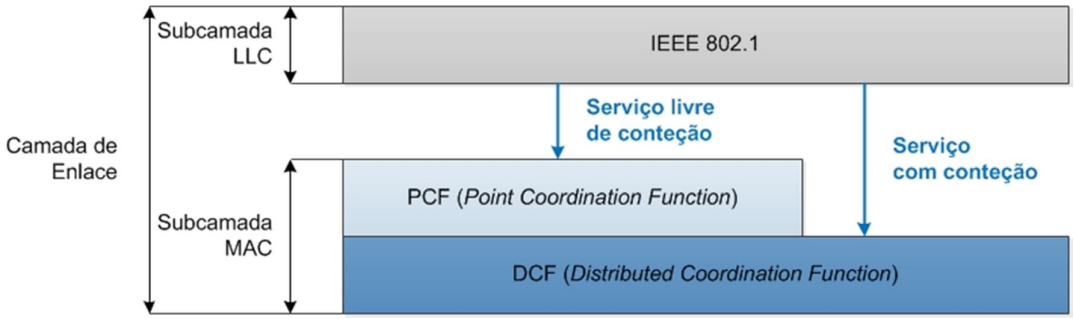

Figura 7 - Subcamadas MAC no padrão IEEE 802.11
Fonte: Forouzan (2008)
Nota: Adaptado pelo autor

7.3.1 DCF

O protocolo DCF utiliza o método CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*), por se tratar de uma rede do padrão IEEE 802.11 não sendo possível aproveitar o método que é usado em redes cabeadas que é chamado de CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*).

Ambos os métodos “exige que cada estação ‘escute’, primeiramente, a rede (ou verifique o estado do meio de transmissão) antes de iniciar uma transmissão” (FOROUZAN, 2008). O que difere um do outro é que o CSMA/CD possui maneiras de saber se houve uma colisão, enquanto o CSMA/CA, nas tecnologias atuais das placas de redes sem fio, não possui esta capacidade pelo fato que podem existir estações ocultas, assim como o problema de enfraquecimento de sinal durante a transmissão, ocultando as evidências de uma colisão.

O fluxograma do processo realizado pelo CSMA/CA está representado na Figura 8

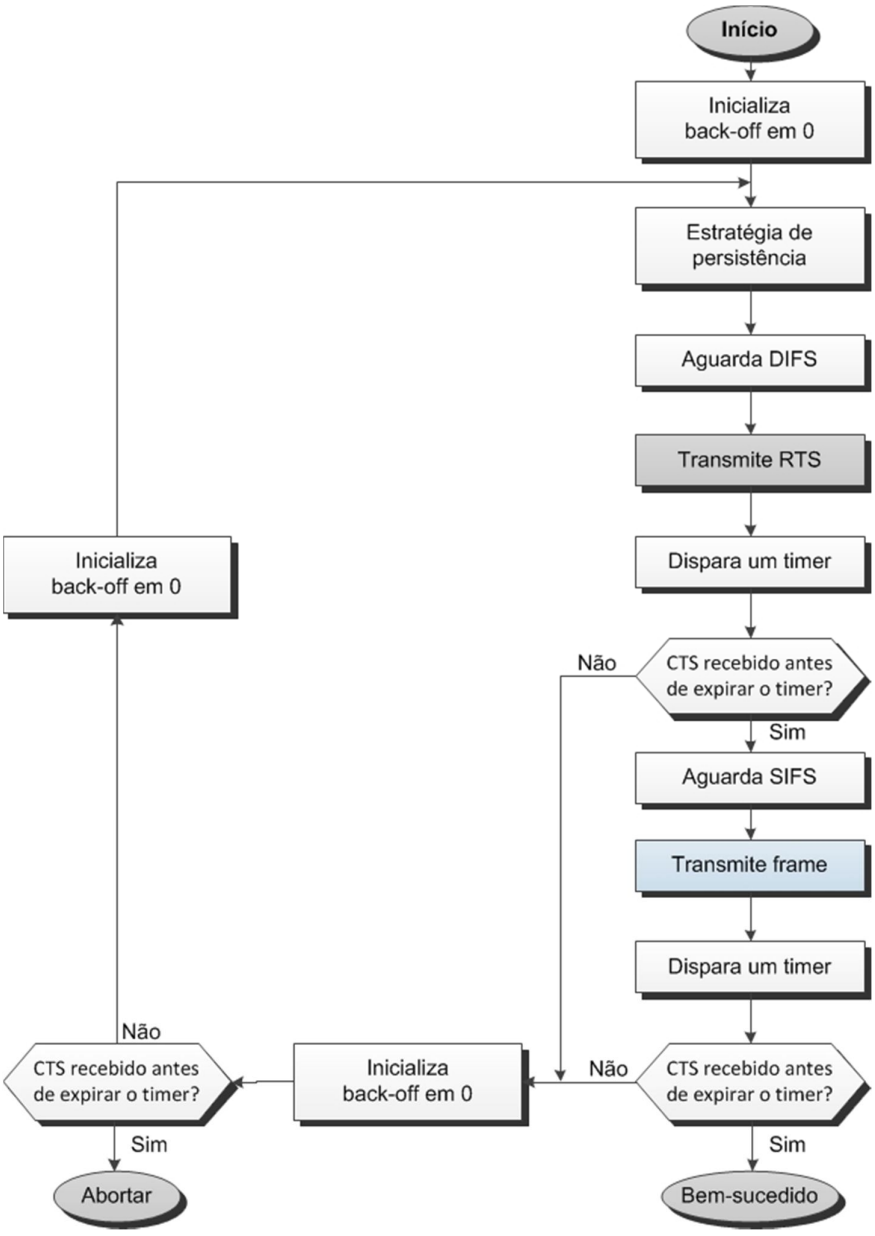

Figura 8 - Fluxograma para o CSMA/CA
Fonte: Forouzan (2008)
Nota: Adaptado pelo autor

De forma geral o método CSMA/CA antes de transmitir, a estação de origem escuta o meio de transmissão, caso este esteja ocupado, é utilizado a estratégia de *back-off*, que consiste na alocação de um número aleatório baseado no número de tentativas de transmissão, e enquanto o canal estiver ocioso, este número vai diminuindo, senão, fica estacionário. Caso o meio de transmissão esteja ocioso, a estação espera por um tempo denominado *Distributed Interframe Space* (DIFS), para realmente ter certeza que não há

nenhuma transmissão em processo. A partir daí, as estações começam o processo chamado de *handshake*[3].

Durante o *handshake* a estação de origem inicia uma transmissão de frame de controle chamado RTS (*Request to Send*). Quando a estação destino recebe o RTS, ela aguarda por um curto período de tempo, chamado de *Short Interframe Space* (SIFS) para que então seja retornada a estação origem um *frame* que indica que a estação destino está pronta para receber os dados. Este *frame* é conhecido como CTS (*Clear to Send*).

Após aguardar um espaço de tempo igual ao SIFS, a estação origem começa a transmissão de seus dados para a estação destino, a qual depois de receber estes dados, espera o curto período do SIFS e retorna um frame ACK (*Acknowledgement*) para a estação origem.

Todos estes frames de controles citados possuem grande importância no protocolo CSMA/CA. Stallings(2005) lembra que

> "o RTS alerta todas as estações dentro da área de recepção da origem que uma troca está em andamento; essas estações desistem da transmissão para evitar uma colisão entre dois quadros transmitidos ao mesmo tempo. Da mesma forma o CTS alerta todas as estações dentro da área de recepção do destino que uma troca está em andamento."

Forouzan (2008) aponta a importância do ACK ao término da transmissão "pois a estação de origem não tem nenhum outro meio para certificar a chegada bem-sucedida de seus dados no destino". O autor complementa explicando que as estações que estão fora da transação esperam um tempo determinado, tempo este necessário para a transação em andamento se completar, o qual está dentro do RTS. Este tempo fica carregado no NAV – *Network Allocation Vector*. Sendo assim as estações que tiverem algum tempo carregado em seu NAV, devem esperar por este tempo antes de começar a analisar os pacotes da rede novamente.

7.3.2 PCF

Este protocolo é utilizado em casos onde existe uma transmissão de dados das quais não pode haver atraso, e só pode ser implementado em uma

[3] Processo que estabelece uma sessão TCP entre dois pontos finais (CISCO, 2013).

rede de infra-estrutura, ou seja, onde existe a presença de um *Access Point,* o qual irá fazer *polling*, uma técnica para fazer uma varredura de todas as estações possíveis dentro da rede, recebendo destas estações os dados que desejam transmitir (FOROUZAN, 2008).

Com os dados centralizados sobre o AP, um novo temporizador (assim como o DIFS e SIFS) é criado, chamado de PIFS (PCF *Interframe Space*), sendo este mais curto que o DIFS. Por consequência fica evidente que se uma estação quer usar o DCF e um AP quer utilizar o PCF, o último terá prioridade, pois seu tempo de espera antes de enviar o CTS é menor, logo enviará primeiro.

8 MECANISMOS DE SEGURANÇA

Nas redes cabeadas só é possível obter informações da rede se o atacante estiver conectado física ou remotamente a algum equipamento na rede, enquanto as redes sem fio a princípio não teriam esta segurança pelo fato dos dados estarem trafegando livremente pelo ar. Por este motivo foram criados protocolos de segurança para o padrão 802.11 da IEEE, sendo eles o WEP (*Wired Equivalent Privacy*), WPA (*Wi-fi Protected Access*) e WPA2. De acordo com Rufino (2011) estes protocolos deveriam, entre outras características, ser suficientemente forte, serem capaz de serem implementados em equipamentos com baixo poder de processamento, e por fim, terem autossincronísmo, ou seja, permitir o funcionamento de um equipamento com a mínima ou nenhuma intervenção manual assim que este entre na área de cobertura.

8.1 WEP

Este protocolo utiliza algoritmos simétricos para a codificação de seus dados. Seu funcionamento consiste na criação de quatro novas chaves geradas a partir da chave pública, geralmente de 104 *bits*. Para evitar ataques de dicionário e força bruta, é adicionado um segundo elemento de 24 *bits* gerado por uma função pseudoaleatória, chamado de *Inicialization Vector*, ou

simplesmente IV. “Entretanto, os 24 *bits* passam em claro pela rede, já que essa foi a forma encontrada para dar conhecimento desse valor, possibilitando que os elementos da rede estabeleçam a comunicação cifrada” (RUFINO, 2011).

Fica evidente então uma grande falha – de tantas outras – da WEP, concordando com a citação de Tanenbaum (2003), na qual a IEEE publicou em setembro de 2001 a violação completa do protocolo, com avisos que WEP não era melhor que a Ethernet (rede cabeada), recomendações para outros sistemas de segurança, entre outros tópicos.

8.2 WPA

O WPA primeiramente veio para substituir o fracasso do WEP, e muitas mudanças e avanços foram incorporados. Uma das mudanças é que redes *ad-hoc* não estão disponíveis no WPA, ao contrário do que era no WEP. Rufino (2011) aponta que algumas das mudanças encontradas no WPA são os tipos de protocolos usados, sendo um para pequenas redes e de uso doméstico, enquanto o outro é para redes com uma infraestrutura maior, onde seria necessário no mínimo um servidor de autenticação.

Para este trabalho, será dada uma maior atenção ao primeiro tipo, voltado às redes domésticas, conhecido como PSK (*Pre-Shared Key*, ou “chave pré-compartilhada”) ou também como WPA-PSK. Esta chave é responsável pelo reconhecimento do equipamento pelo concentrador (hub, switch ou roteador).

O TKIP – *Temporal Key Integrity Protocol* – é um protocolo que apareceu para melhorar algumas das vulnerabilidades presentes no WEP em relação aos IVs. Rufino (2011) aponta que a maior vantagem do TKIP é a troca constante de chaves, que no WEP eram estáticas e em texto puro. Outras vantagens são o aumento do tamanho dos IVs de 28 *bits* para 48 *bits*, assim como a possibilidade de substituir o IV a cada pacote, sessão ou por período.

8.3 WPA2

Sendo a versão mais nova e segura do WPA, o WPA2 traz consigo a implementação do protocolo CCMA (*Counter Cipher Mode with Block Chaining Message Authentication Code Protocol*), utilizando o algoritmo AES, o qual será mais bem explanado no capítulo 10.1.3. Com base nas afirmações de Haines (2010), tanto o WPA quanto o WPA2 quando estão no modo PSK, utiliza-se de chaves de 256 *bits* em hexadecimal, gerados pelo algoritmo PBKDF2 – *Password-Based Key Derivation Functions*, presente no capítulo 10.4.4 com um foco mais detalhista sobre ele. Isto foi proposto devido a problemas de padronização na entrada da chave, pois não era especificado se deviam estar no formato hexadecimal ou no mais comum ASCII.

Outra novidade apresentada no WPA2 é a vinda do protocolo *Extensible Authentication Protocol* (EAP), “que permite integrar soluções e de autenticação já conhecidas e testadas” (RUFINO, 2011). Como por exemplo, certificados digitais, senhas dinâmicas (*One Time Password*), e também associação do AP com os endereços MACs das estações permitidas.

9 SEGURANÇA DA INFORMAÇÃO

A Cisco (2013) aponta que assim como o tamanho das redes, a importância destas também cresceram, e seu comprometimento causaria grandes transtornos e as ameaças que podem causar tal comprometimento vem crescendo a cada dia, exigindo do atacante cada vez menos de conhecimento técnico devido às facilidades oferecidas pelas ferramentas.

9.1 PRINCÍPIOS

Na área da segurança da informação, existem diversos aspectos básicos, mas quatro deles são pilares que merecem uma atenção maior, sendo eles a confidencialidade, integridade e disponibilidade e a autenticação.

Confidencialidade é segundo Lyra (2008), “a capacidade de um sistema de permitir que alguns usuários acessem determinadas informações ao mesmo

tempo em que impede que outros, não autorizados, a vejam". Já a disponibilidade, o autor afirma que é ter acesso à informação em qualquer hora para todos que precisem dela e tenham autorização para isto.

Kurose e Ross (2006) apontam a integridade como a certeza que o conteúdo de uma comunicação chegue a seu remetente sem alteração durante a transmissão. Enquanto para autenticidade estes mesmos autores conceituam como a confirmação das partes envolvidas na comunicação.

9.2 TIPOS DE ATAQUES

Os ataques que são aqui citados são ataques restritos à captura de senhas, devido este ser parte do objetivo do trabalho. A disposição dos tipos de ataques está descrito de acordo com o que foi apresentado por Clavis (2013).

9.2.1 ATAQUES NÃO-ELETRÔNICOS

Estes ataques não utilizam o meio eletrônico para serem realizados, e sim as falhas nos hábitos das pessoas. São divididos em três categorias:

- Engenharia Social: Mitnick e Simon(2003) conceituam esta categoria como o uso da "influência e persuasão para enganar as pessoas, e convencê-las de que o engenheiro social é alguém que na verdade ele não é, ou pela manipulação". Isto resulta na obtenção de informações pelo engenheiro social aproveitando-se da ignorância ou ingênuidade da vítima.
- *Dumpster Diving*: Ou "mergulho do gari" (em tradução literal) é de acordo com Thomas (2007), quando atacantes utilizam "a prática de revirar o lixo de um escritório, ou instalação técnica para extrair dados confidenciais".
- *Shoulder Surfing*: consiste na observação do atacante sobre o usuário digitando sua senha.

9.2.2 ATAQUES OFF-LINE

Para este tipo de ataque, é necessário que o atacante possua um arquivo com os *hashes* (senhas criptografadas de forma irreversível). Existem duas modalidades para estes ataques, sendo a primeira quando o atacante possui uma lista de senha, e gera o *hash* de cada uma até encontrar o *hash* que está no arquivo. A segunda modalidade é quando o atacante possui uma lista de *hashes* para então descobrir qual a senha que gera o *hash* presente no arquivo.

Na primeira modalidade podem ser aplicados as técnicas de força bruta e dicionário. Erickson (2009) conceitua que é um ataque onde é testado cada combinação possível.

Já na segunda modalidade a técnica aplicada é a *Lookup Table*, a qual é tratada no tópico 12.

9.2.3 ATAQUES PASSIVOS

Ataques passivos não dependem da força da senha, e são difíceis de serem detectados por não produzirem ruídos (tráfego) na rede. Duas categorias podem ser descritas para este ataque de acordo com Clavis (2013).

- *Wire Sniffing*: Captura da senha sem criptografia durante uma escuta na rede.
- *Man-in-the-middle*: também conhecido pelo acrônimo MITM, esta categoria se caracteriza pelo posicionamento do atacante entre as estações que estão se comunicando, recebendo e enviando ao destino todo o tráfego de forma transparente para os usuários.

9.2.4 ATAQUES ATIVOS

Estes ataques produzem ruídos na rede, e a utilização de algumas categorias podem ser facilmente detectadas com *software*s apropriados. As categorias são:

- Dicionário: Utiliza um arquivo contendo diversas palavras para serem testadas como senha.
- Força-Bruta: Esta técnica tenta cada combinação possível, e segundo Erickson (2009), esse tipo de é capaz de encontrar qualquer senha possível, porém tomando um tempo enorme para ser realizado.
- Tentativa e Erro Senha padrão: O atacante com experiência, tenta acessar dispositivos (como um roteador), e testa as senhas padrões de fábrica para ter acesso à este dispositivo.
- Instalação de *malware*: Nessa categoria, é onde se encontra diversos *software*s maliciosos, inclusive os famosos *keyloggers*, um *malware* que capta cada tecla digitada pela vítima e envia ao atacante.

9.3 SEGURANÇA FÍSICA

De acordo com a ABNT (2005), onde são descritas as normas da ISO 17.799/2005 segurança física é definida como:

> "as instalações de processamento da informação críticas ou sensíveis sejam mantidas em áreas seguras, protegidas por perímetros de segurança definidos, com barreira de segurança e controles de acesso apropriados. Convém que sejam fisicamente protegidas contra o acesso não autorizado, danos e interferências".

De forma mais detalhada, os perímetros de segurança são uma área delimitada por uma linha imaginária que separa de outros espaços físicos por qualquer medida preventiva que dificultam o acesso não autorizado aos ativos da informação, como por exemplo, a biometria.

Estas e outras recomendações como arquivamento de maneira correta de dados impressos em papeis ou mídias de computadores, assim como prevenções que devem ser tomadas em relação a desastres naturais e acidentes.

9.4 SEGURANÇA LÓGICA

No campo da segurança lógica, pode-se destacar a segurança em redes, onde as empresas poderiam criar VPNs – *Virtual Private Network*. Outras recomendações também feitas pela ABNT (2005) seria o uso de *firewalls*, “recursos de segurança que têm o objetivo de controlar o acesso às redes de computadores”, de acordo com Lyra (2008), examinando todo o tráfego, bloqueando ou liberando de acordo com as normas configuradas para o mesmo.

Vale ressaltar também os antivírus, *softwares* que ajudam a impedir ataques, principalmente os provenientes de *malwares*, citados anteriormente. Por último e não menos importante, encontra-se a criptografia. Esta técnica é muito utilizada na segurança da informação e será explicada mais detalhadamente no capítulo 10, sendo praticamente um pré-requisito para um sistema seguro. Isto não implica que apenas a criptografia garanta a segurança necessária ao usuário.

10 CRIPTOGRAFIA

A criptografia é uma das áreas estudadas pela criptologia e está hoje presente em grande parte do nosso dia-a-dia de uma forma transparente para os usuários. Esta técnica permite que possamos comprar *on-line*, visualizar nossa conta bancária, transmitir e receber dados sigilosos, entre outros inúmeros exemplos.

De maneira sucinta, pode-se dizer que a criptografia é aplicada a uma mensagem em texto plano, ou seja, aquele que estamos acostumados e conseguimos ler pelo transmissor e enviado pela rede. O receptor por sua vez, com o uso de uma chave compartilhada entre ele e o transmissor, pode descriptografar a mensagem e recuperar o texto à sua forma simples e legível (PETERSON; DAVIE, 2004). Esta técnica além de fornecer privacidade entre as partes envolvidas, também provê alguns serviços importantes como autenticação e integridade.

O resultado de uma criptografia, chamado de texto cifrado, é concebido através de algoritmos matemáticos que podem ou não ser reversíveis, onde

recebe o texto cifrado e transforma em texto plano. Stallings (2008) salienta que a criptografia é a “ferramenta automatizada mais importante para a segurança da rede e das comunicações”. O mesmo também afirma que existem “duas formas de criptografia que são usadas normalmente: criptografia convencional, ou simétrica, e criptografia por chave pública, ou assimétrica”.

10.1 CRIPTOGRAFIA SIMÉTRICA

De acordo com Stallings (2008), um esquema de criptografia simétrica necessita de cinco elementos, sendo eles o texto plano, o algoritmo de criptografia, a chave secreta, o texto cifrado e o algoritmo de criptografia. A Figura 9 mostra o fluxo de uma criptografia simétrica.

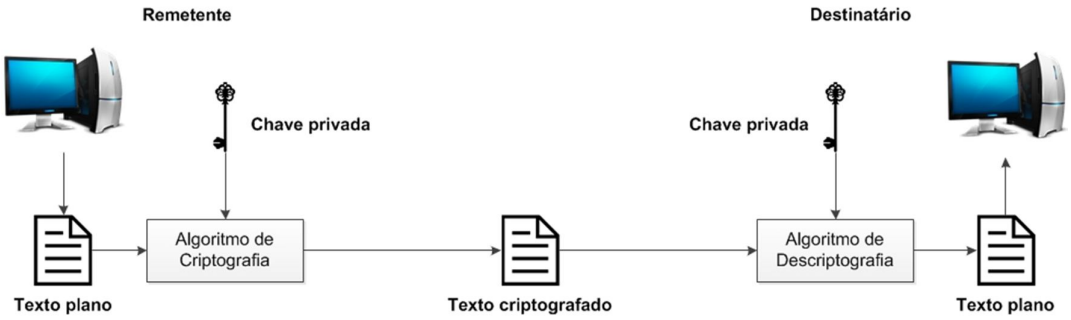

Figura 9 - Criptografia Simétrica
Fonte: Elaborada pelo autor (2013)

Na criptografia simétrica, tanto o remetente quanto o destinatário possuem a mesma chave secreta. Os algoritmos desenvolvidos para este tipo de criptografia são baseados em substituição, onde cada elemento do texto plano é substituído por outro, e transposição, que é a reorganização dos elementos do texto plano.

Existem também diferenças no modo como o texto plano é processado podendo ser dividido em cifra de bloco e cifra em fluxo.No primeiro, o texto plano é divido em blocos e este é criptografado, enquanto no segundo, de acordo com Stallings (2008) “processa os elementos de entrada continuamente, produzindo a saída de um elemento de cada vez”.

Os três principais algoritmos de criptografia simétrica utilizados para estudo são o DES (*Data Encryption Standard*), o 3DES (*Triple DES*) e o AES

(*Advanced Encryption Standard*) e todos serão abordados com mais detalhes neste trabalho.

10.1.1 DES

O DES utiliza o mesmo algoritmo tanto para criptografia quanto para descriptografia e exige duas entradas de 64 *bits* de extensão, sendo eles o texto plano e a chave. Peterson e Davie (2004) afirmam que apesar de a chave requerida ser de 64 *bits*, são utilizados apenas 56 deles sendo que “o último *bit* de cada um dos 8 *bytes* da chave é um *bit* de paridade para esse byte”.

Existem três fases que devem ser realizadas para que possa ser gerado um texto cifrado pelo DES, as quais estão representadas na Figura 10. Primeiramente os 64 *bits* do texto plano de entrada são permutados, ou seja, embaralhados, passando então para a segunda fase.

Nesta fase, logo após os *bits* serem permutados, os *bits* utilizam uma cifra conhecida como Feistel durante dezesseis rodadas. A rodada atual no algoritmo será representada pela letra “i”.

De acordo com Erickson (2009), os *bits* são divididos em dois grupos L (*Left*, ou esquerda) e R (*Right*, ou direita), e a cada rodada o novo L é definido para ser igual à antiga metade direita (R_{i-1}), e o novo R é definido para ser igual à antiga metade esquerda (L_{i-1}) com a operação lógica XOR (OU exclusivo) sobre a saída de uma função F, que usa a antiga metade direita (R_{i-1}) e a subchave para a volta chamada de K_i.

A geração desta subchave K_i também se inicia com uma permutação nos 56 *bits* iniciais e logo após é dividida em grupos de 28 *bits,* e um deslocamento esquerdo circular a estes de um ou dois *bits*, dependendo da rodada. Logo após o deslocamento, é feito uma permutação e contração, as quais não serão tratadas em detalhes neste trabalho, para retornar um valor de 48 *bits* que serve como entrada para a função F (STALLINGS, 2008).

Outro ponto importante a ser declarado é que a função F precisa que R_{i-1} também tenha 48 *bits* para que possa ser calculado o XOR entre estes *bits* e os *bits* da subchave. Para isso também é feito uma expansão e permutação dos *bits* contidos em R_{i-1} e para que então seja realizado o XOR. Este resultado será de 48 *bits*, e deve ser reduzido aos 32*bits* utilizados tanto para L quanto

para R. Para isto, é utilizado algo chamado de caixa de substituição (ou caixa-S), a qual reduz cada pedaço de 6 *bits* para 4 *bits*, retornando assim os 32 *bits* necessários para cada rodada.

Por fim, ao término da última rodada é realizada uma permutação inversa à inicial resultando assim no texto cifrado.

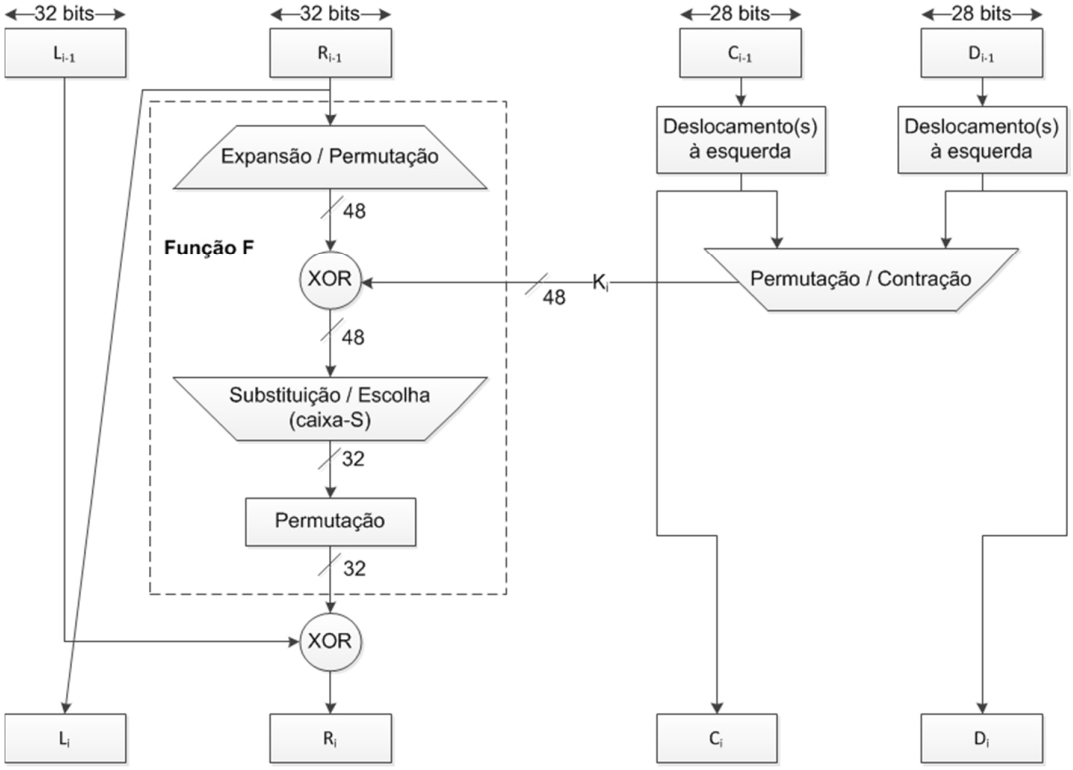

Figura 10 - Rodada individual do algoritmo DES
Fonte: Stallings (2008)
Nota: Adaptado pelo autor (2013)

10.1.2 3DES

Conclusões feitas sobre o DES resultaram que este não era seguro e que demoraria um tempo considerado pequeno para desvendar o DES. A única fraqueza realmente conhecida do DES era o tamanho de sua chave de 56 *bits*, de acordo com Erickson (2009). Sendo assim, surgiu o 3DES, ou seja, criptografar os dados três vezes. Peterson e Davie (2004) relatam que “isso pode ser feito com três chaves separadas, ou com duas chaves: a primeira é usada, depois a segunda e finalmente a primeira é usada novamente”.

10.1.3 AES

Também conhecido como algoritmo de Rijndael, o AES é uma cifra de bloco e foi concebido por meio de um concurso realizado pelo NIST (*National Institute of Standards and Technology* – Instituto Nacional de Padrões e Tecnologias), órgão americano responsável por aprovar padrões nacionais. A Figura 11 mostra as diferentes configurações apresentadas pelo AES.

Tamanho do Bloco de Dados	Número de Ciclos	Tamanho da Chave
128 *bits*	10	128 *bits*
	12	192 *bits*
	14	256 *bits*

Figura 11 - Configuração AES
Fonte: Forouzan (2008)

A princípio, Tanenbaum (2003) afirma que pelo fato do algoritmo utilizar blocos de dados de 128 *bits*, ou 16 *bytes*, é criado uma matriz de 4x4 *bytes* chamada de *state*, onde são armazenados todos os *bits*, de forma que os quatro primeiros *bytes* fiquem armazenados na primeira coluna, os quatro *bytes* seguintes na próxima coluna, e assim por diante. De acordo com Stallings (2008) são utilizados quatro processos para cada ciclo, exceto o último. Para cada processo, é utilizada uma chave distinta de 128 *bits*, recebida através de uma expansão da chave fornecida. Os processos realizados são respectivamente:

- *AddRoundKey*: Nesta transformação, todos os 128 *bits* do *state* sofrem uma alteração, passando por um XOR *bit* a *bit* com os 128 *bits* da chave da rodada, ilustrado pela Figura 12.

Figura 12 - Transformação AddRoundKey
Fonte: Stallings (2008)
Nota: Adaptado pelo autor

- *SubBytes*: O AES define uma matriz 16x16 *bytes* chamada de caixa-S, contendo todos os valores possíveis de permutação com 8 *bits*. Esta transformação de substituição utiliza os quatro primeiro *bits* para definir a linha dentro da caixa-S, enquanto a coluna é definida pelos quatro últimos *bits*. O algoritmo mapeia então o valor resultante para esta operação na caixa-S. Este processo é representado na Figura 13.

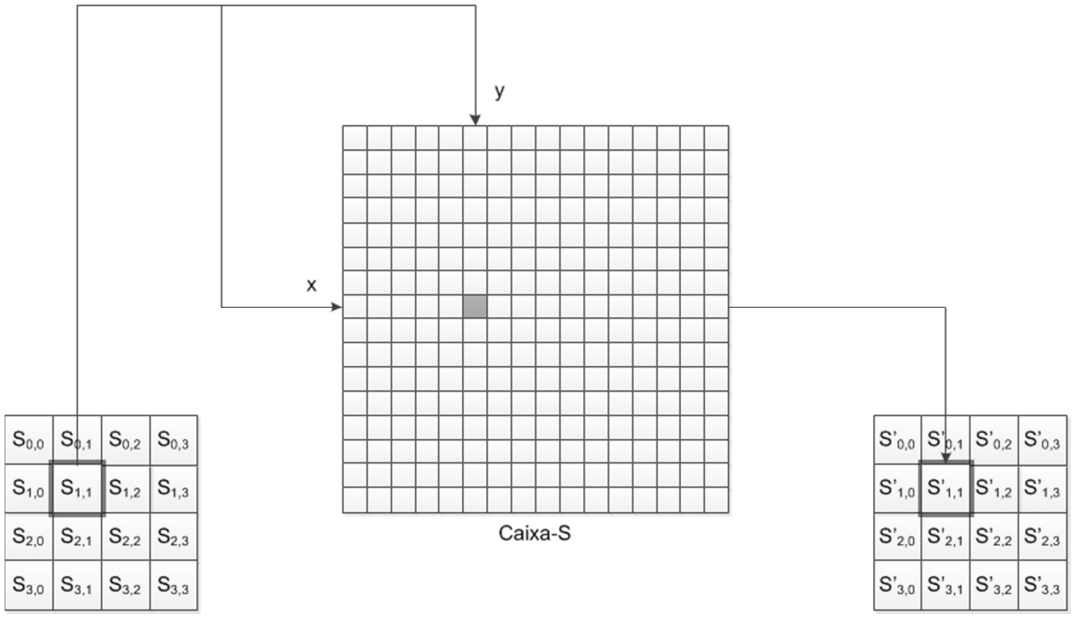

Figura 13 - Transformação SubBytes
Fonte: Stallings (2008)
Nota: Adaptado pelo autor

- *ShiftRow*: São deslocamentos circulares à esquerda nas linhas do *state* exercidas pelos *bytes*. Conforme mostra a Figura 14, a primeira linha, não sofre alteração, a segunda desloca circularmente um byte, a terceira desloca dois *bytes* e a quarta desloca três *bytes*.

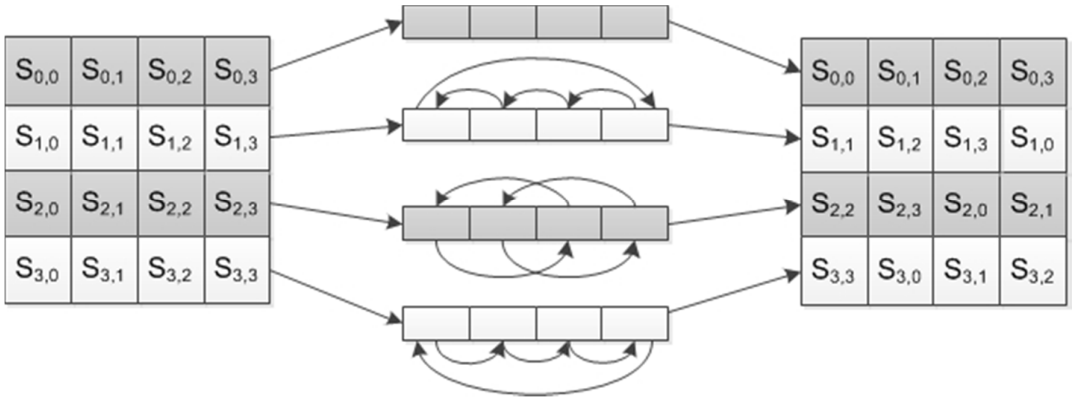

Figura 14 - Transformação ShiftRow
Fonte: Stallings (2008)
Nota: Adaptado pelo autor

- *MixColumns*: Cada coluna do *state* é misturado independente uma das outras. Essa mistura é uma "multiplicação da coluna antiga por uma matriz constante, sendo a multiplicação feita com o campo finito de Galois, GF(2^8)" (TANENBAUM, 2003). Para esta operação não será aprofundado nos detalhes devido ao grau de complexidade apresentado.

10.2 CRIPTOGRAFIA ASSIMÉTRICA

Os algoritmos de criptografia assimétrica utilizam chaves distintas, porém relacionadas, para a criptografia e descriptografia. De acordo com Stallings (2008), cada usuário gera um par de chaves, dentre as quais uma delas se torna pública, enquanto a outra permanece privada.

Os usuários quando vão enviar uma informação para outro, deve então escolher a chave pública do destinatário. Quando a mensagem chega ao seu destino, o destinatário utiliza a sua própria chave privada, conforme mostra a Figura 15.

Tanto o RSA – Rivest-Shamir-Adleman (sobrenomes dos autores), quanto o DH (Diffie e Hellman) são algoritmos que utilizam a criptografia assimétrica e serão explicados com maiores detalhes.

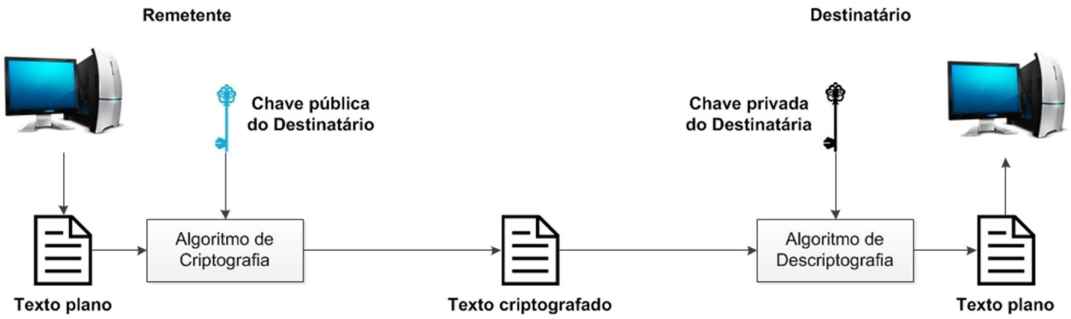

Figura 15 - Criptografia Assimétrica
Fonte: Elaborada pelo autor (2013)

10.2.1 DH (DIFFIE, HELLMAN)

O algoritmo DH foi o primeiro algoritmo de chave pública publicado. Consiste em cálculos em cima de números primos grandes que são públicos, ou seja, qualquer um pode ter acesso. De acordo com Forouzan (2008) existem seis passos, tomando como exemplo a Figura 16, uma comunicação entre Bob e Alice:

- Alice escolhe um número aleatório grande x e calcula $R_1 = g^x \ mod$ p.
- Bob escolhe outro número aleatório grande y e calcula $R_2 = g^y \ mod$ p.
- Alice envia R_1 para Bob.
- Bob envia R_2 para Alice.
- Alice calcula $K = (R_2)^x \ mod$ p.
- Bob calcula $K = (R_1)^y \ mod$ p.

Sobre os dois últimos passos é correto afirmar que “pelas leis da aritmética modular, ambos os cálculos produzem $g^{xy} \ mod$ p” (TANENBAUM, 2003). Com isto, as duas partes da comunicação compartilham a mesma chave, sem que um saiba o valor escolhido inicialmente pelo outro.

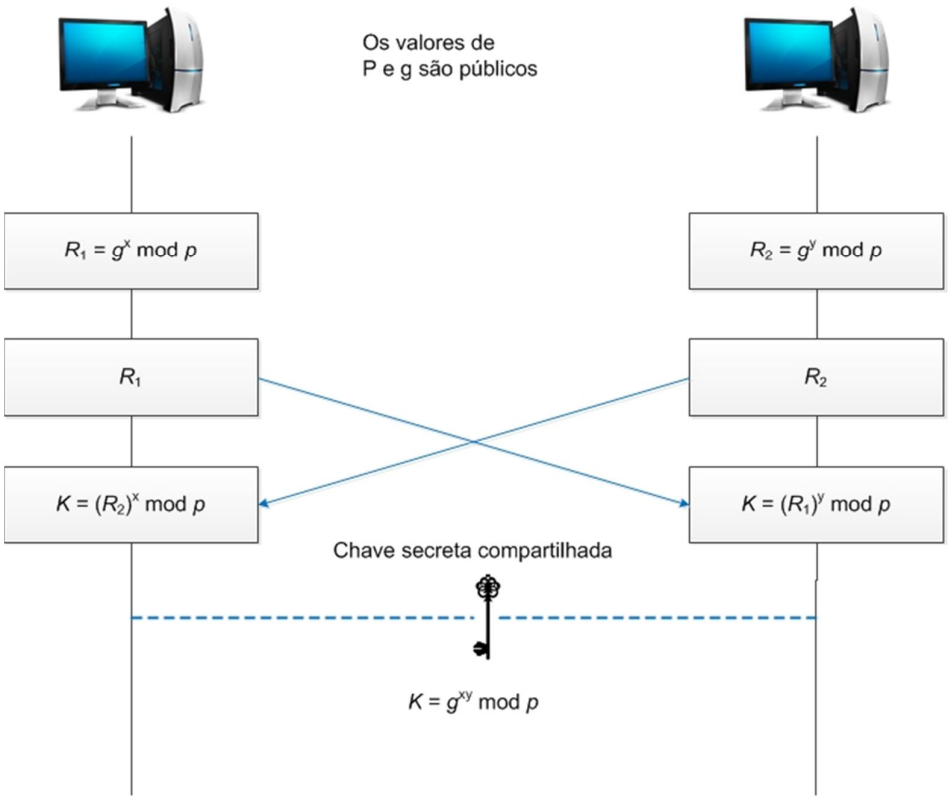

Figura 16 - Método Diffie-Hellman
Fonta: Forouzan (2008)
Nota: Adaptado pelo autor

10.2.2 RSA

Erickson (2009) afirma que “o RSA é um dos algoritmos assimétricos mais populares. A segurança da RSA é baseada na dificuldade de decompor números grandes”. Hoje este algoritmo é encontrado em *websites* como Gmail, Facebook e Twitter, por exemplo.

De acordo com Kurose e Ross (2006), para escolher as chaves públicas e privadas é necessário executar as seguintes etapas:

- Escolher dois números primos grandes, p e q, geralmente na ordem de 1024 *bits*, ou até mesmo de 2048 *bits*.
- Calcular n, o produto desses dois números, ou seja: n = p * q.
- Calcular z, sendo z = (p-1) * (q-1).

- Encontrar o número referente à letra e (*encryption*). Este número precisa ser menor que z e que sejam primos entre eles, quando não há fatores em comum exceto o número 1.
- Encontrar o número d, sabendo que a divisão [(e*d)-1]/z não sobre resto. Relembrando que a expressão *mod* significa o resto da divisão. Sendo assim, podemos dizer também que o número d escolhido tal que e*d *mod* z=1.
- Estes cálculos resultarão na chave pública K^+, denominada pelo par de números (n,e) e na chave privada K^-, sendo os números (n,d).

Como exemplo, podemos citar uma situação fictícia entre duas pessoas, João e Maria. João deseja enviar à Maria uma mensagem (M) contendo a palavra "*kryptos*". Logo, João terá que utilizar a chave pública de Maria (K_M^+) para resultar em $K_M^+(M)$.

Maria ao receber a mensagem irá utilizar a sua chave privada utilizada em seu algoritmo de decriptografia sobre a mensagem cifrada que João mandou, resultando na seguinte expressão: $K_M^-[K_M^+(M)] = (M)$.

De forma mais detalhada, seguiremos as etapas com os valores sugeridos por Kurose e Ross (2006). A letra 'p' receberá o número primo 5 enquanto 'q' receberá o número primo 7. Logo n=5*7 que resultará no valor 35. Já a letra 'z', resultaria no número 24.

Posteriormente, precisamos encontrar valores à letra e de forma que não tenha fatores em comum (a não ser pelo número um) e seja menor que z. Vamos supor então que e=5. Para o valor de d, sabendo que seu produto com e dividido por z deve sobrar 1, será adotado o número 29, pois 29*5*mod*24=1. Por fim, sabemos que a chave pública de Maria é (35,5) e a chave privada da mesma é (35,29).

Tanto a criptografia quanto a decriptografia da mensagem (M) – levando em consideração que os valores das letras variam de acordo com sua posição no alfabeto, ou seja, 'a' = 1 e 'z' = 26 – estarão presentes no Figura 17 e no Figura 18, respectivamente:

Letra texto plano	m: representação	m^e	Texto cifrado $c = m^e mod\ n$
K	11	161051	16
R	18	1889568	23
Y	25	9765625	30
P	16	1048576	11
T	20	3200000	20
O	15	759375	15
S	19	2476099	24

Figura 17 - Criptografia RSA: e=5, n=35
Fonte: Elaborada pelo autor (2013)

c	c^d	$M=c^d mod\ n$	Letra texto plano
16	830767497365572000000000000000000000	11	K
23	309105864309354000000000000000000000000	18	R
30	68630377364883000000000000000000000000000000	25	Y
11	158630929717149000000000000000	16	P
20	536870912000000000000000000000000000	20	T
15	127834039488589000000000000000000	15	O
24	106200365064067000000000000000000000000000	19	S

Figura 18 - Decriptação RSA: d=29, n=35
Fonte: Elaborada pelo autor (2013)

10.3 AUTENTICAÇÃO DE MENSAGEM

Segundo Stallings (2008), “a autenticação de mensagem é um mecanismo ou serviço utilizado para verificar a integridade de uma mensagem”. Existem duas técnicas mais comuns de autenticação de mensagem, sendo elas um código de autenticação de mensagem (MAC[4] – *Message Authentication Code*), ou o produto de uma função *hash*, em que o *hash*, gerado normalmente é denominado como MDC (*Modification Detection Code*, ou “Código de Detecção de Modificações” em português), pois “o código é capaz de detectar qualquer modificação na mensagem” (FOROUZAN, 2008).

Em resumo, o MAC é uma função calculada sobre a mensagem e uma chave secreta, a qual as duas partes da comunicação têm conhecimento (chave simétrica). O resultado desta função é anexado à mensagem, e enviada

[4] Neste capítulo de criptografia, MAC será tratado como *Message Authentication Code*. Não confundir com o MAC utilizado nos capítulos anteriores, o *Medium Access Control*.

ao destinatário, onde será realizado o mesmo procedimento. Após essa etapa, o destinatário irá comparar o seu MAC, com o MAC anexado à mensagem que foi enviada pelo remetente. Se ambos os MACs forem iguais, Stallings (2008) deduz que o destinatário tem certeza que a mensagem está íntegra, e que a mensagem veio do remetente declarado, supondo que apenas ambos sabiam a chave secreta. A Figura 19 apresenta os processos citados.

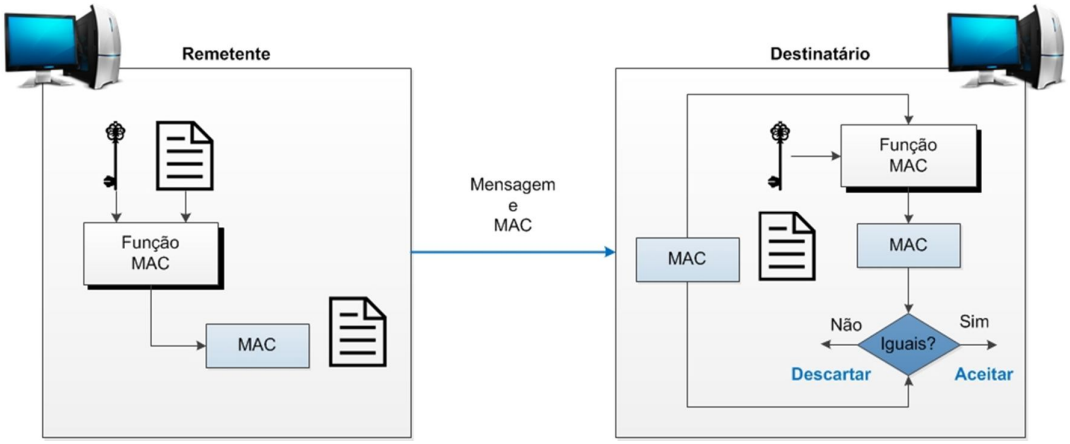

Figura 19 – MAC
Fonte: Forouzan (2008)
Nota: Adaptado pelo autor

Um *hash*, ou também conhecido como resumo de mensagem (*message digest*), não possui chaves e de acordo com Kurose e Ross (2006) é

> "muito parecido com uma soma de verificação. Algoritmos de resumo de mensagem pegam uma mensagem m, de comprimento arbitrário, e calculam uma 'impressão digital' dos dados, com comprimento fixo, conhecida como resumo de mensagem H(m)".

Uma afirmação importante dada por Moreno, Pereira e Chiaramonte (2005) é sobre as chamadas colisões, que consistem em resultados de uma função *hash* iguais para dados de entrada diferente, sendo este o ponto principal a ser evitado durante o desenvolvimento de uma função *hash*.

10.4 ALGORITMOS DE *HASH* E DE MAC

10.4.1 MD5

Com processamento em blocos de 512 *bits* e saída de 128 *bits*, o *Message Digest 5* (MD5) suporta uma entrada de comprimento arbitrário.Tanenbaum (2003) cita que o MD5 primeiramente expande o tamanho da mensagem com *bits* 0, com excessão do primeiro *bit* de expansão, até chegar em uma diferença de 448 para ser múltiplo de 512. Sendo assim, o tamanho do último bloco contendo 448 *bits* é preenchido com 64 *bits* resultantes do tamanho em *bits* da mensagem original e “caso o tamanho da mensagem não possa ser representado em 64 *bits*, apenas os *bits* menos significativos serão considerados” (MORENO; PEREIRA; CHIARAMONTE, 2005).

Logo após é criado um*buffer*de 128 *bits* com valores iniciais fixos para serem utilizados no processamento e cálculo do *hash*. Para questões de performance, o algoritmo cria uma tabela contendo valores fixos provenientes de uma função seno, as quais são utilizadas em funções diferentes dentro de quatro passos para cada bloco de entrada de 512 *bits*.

Após o cálculo de todas as operações de um bloco é somado o novo valor aos valores antigos do *buffer*. De acordo comTanenbaum (2003), o processo é repetido até que todos os blocos de entrada tenham sido processados, sendo que os 128 *bits* armazenados em *buffer* formam o resultado da função *hash*.

10.4.2 SHA-1

O SHA-1 (*Secure Hash Algorithm*) é um padrão federal norte-americano, e seu uso é exigido quando aplicações do âmbito federal precisam de um resumo de mensagem seguro Kurose e Ross (2006).

Stallings (2008) por sua vez, afirma que além do SHA-1 existem outras três variantes: SHA-256, SHA-384 e SHA-512, cada qual retorna o número de *bits* correspondentes em seus nomes. O SHA-1 produz um valor de apenas

160 *bits* e por esta razão, vulnerabilidades foram sendo encontradas e desde 2005 o NIST pretende tirar o SHA-1 do mercado, porém até hoje existem aplicações e funções que utilizam este algoritmo.

Assim com o MD5, é feita uma expansão de *bits* até que o tamanho seja múltiplo de 512 *bits*. Logo após, de acordo com Tanenbaum (2003), “um número de 64 *bits* contendo o tamanho da mensagem antes do preenchimento é submetido a uma operação OR nos 64 *bits* de baixa ordem”.

Tanenbaum (2003) continua explicando que são criados cinco variáveis de 32 *bits*, de H_0 a H_4, as quais são inicializadas como constantes específicadas no padrão. Estas variáveis é onde o *hash* se acumula. Agora, após estes processos, os blocos de 512 *bits* são processados, sendo que para cada bloco as 16 palavras (*words* – unidade que contém 32 *bits*) são copiadas dentro de um vetor de 80 posições, restando 64 os quais são preenchidos utilizando uma determinada fórmula que faz rotação circular à esquerda.

Para completar as operações executadas em um único bloco, são criadas mais cinco variáveis, chamadas de A, B, C, D e E, contendo os valores de H_0 a H_4 respectivamente. Oitenta iterações de cálculos são realizados utilizando funções diferentes que mudam a cada 20 iteração. Após o término dos cálculos, os valores de A até E são somados aos valores de H_0 a H_4.

Com o término das operações neste bloco, o próximo é preparado para fazer o mesmo processo, com os vetores de palavras reinicializado, e com o vetor H com os valores obtidos do processo anterior, seguindo esta sequência até que todos os blocos tenham sido processados. Ao término do último bloco, “as cinco palavras de 32 *bits* no [vetor] H são transmitidas como saída, formando o *hash* criptográfico de 160 *bits*” (TANENBAUM, 2003).

Para melhor visualização, a Figura 20 apresenta um esquema mostrando os blocos de 512 *bits* da mensagem, as variáveis H_0 a H_4 e também o vetor de 80 posições W.

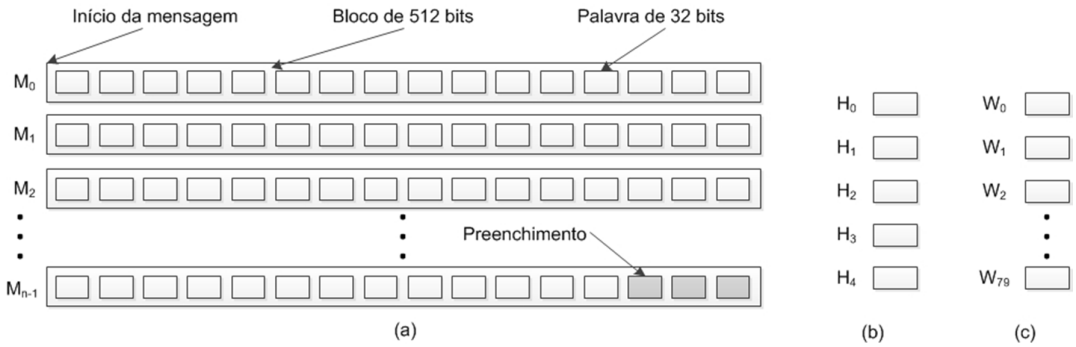

Figura 20 - (a) Uma mensagem preenchida até um múltiplo de 512 *bits*. (b) As variáveis de saída. (c) O array de palavras
Fonte: Tanenbaum (2003)
Nota: Adaptado pelo autor

10.4.3 HMAC

A IETF (2013a) cita que os objetivos principais (traduzidos para o português) por trás do HMAC são:

- Usar, sem modificações, as funções de *hash* disponíveis, particularmente aquelas que tenham bom desempenho em *software*, e para as quais o código esteja ampla e gratuitamente disponível.
- Permitir a substituição fácil da função de *hash* embutida caso sejam encontradas ou exigidas função *hash* mais rápidas ou mais seguras.
- Preservar o desempenho original da função de *hash* sem incorrer em degradação significativa.
- Ter uma análise criptografia bem compreendida quanto à força do mecanismo de autenticação com base em suposições razoáveis sobre a função de *hash* embutida.

De forma simples o HMAC – *Keyed-Hashing for Message Authentication* – é uma implementação do MAC baseado em funções *hash* sem chave, como por exemplo, o MD5 ou o SHA-1.

De acordo com Forouzan, (2008), é aplicado primeiramente uma função *hash* sem chave à mensagem concatenada com a chave simétrica. O resultado será um HMAC intermediário, o qual é novamente processado da mesma forma, concatenando agora o próprio HMAC intermediário, e a mesma chave utilizada no processo anterior resultando então no HMAC. Ao receber o HMAC,

o receptor irá calcular seu próprio HMAC e comparar com os HMACs recebidos para validar a integridade da mensagem e autenticar a origem dos dados.

10.4.4 PBKDF2

Benton (2010) afirma que o PBKDF2 é utilizado devido à falta de proteção por parte dos usuários na escolha da senha, ou PSK, não importando a proteção usada, CCMP ou TKIP. Esta afirmação também é apoiada pelo NIST (2010), o qual também especifica que o PBKDF2 utiliza-se de HMAC junto com SHA-1, além do uso do *salt,* e o contador de iteração que a função deve conter, “devendo ser selecionada o maior possível, desde que o tempo requerido para gerar a chave usando a senha de entrada seja aceitável pelos usuários” (em tradução literal), sendo que o mínimo recomendado pelo órgão é de 1000 iterações enquanto que para sistemas mais poderosos, e que necessitam de muita proteção este número chega a dez milhões.

Os processos executados pelo PBKDF2 descritos pela IETF (2013b) são a adição do *salt* à senha que o usuário digitou, onde este resultado é o dado de entrada para um algoritmo geralmente de *hash*, podendo ser também de uma função pseudoaleatória, durante uma quantidade de iterações estipulada na entrada dos dados, assim como o número de *bits* desejável na saída.

Já Haines (2010) focando nas redes sem fio, afirma que na senha digitada pelo usuário além do *salt*, que seria o SSID, entra também na adição o tamanho do SSID. O algoritmo SHA-1 é o escolhido para processar a soma dos três elementos anteriores, e então utilizar seus dados de saída como dados de entrada para a próxima iteração, durante todas as suas 4096 iterações, retornando ao final um *hash* de 256 *bits*.

11 CRIPTOANÁLISE

A criptoanálise consiste em outra área dos estudos realizados na criptologia, tendo como foco a obtenção dos textos planos sem conhecimento das chaves utilizadas ou ainda como minimizar o esforço computacional para ser realizada esta obtenção (FLORIANO, 2013).

Stallings (2008) reforça esta sentença afirmando que na área de criptoanálise se encontra “as técnicas empregadas para decifrar uma mensagem sem qualquer conhecimento dos detalhes de criptografia”. Continua sua afirmação dizendo que “a criptoanálise é o que os leigos chamam de ‘quebrar o código’”.

Existem diversos ataques criptoanalíticos que podem ser utilizados em diversas criptografias, uns com maiores vantagens outros com menos. Floriano (2013) cita que o criptoanalista deve levar em consideração fatores complexos na realização de uma criptoanálise, como por exemplo, o tempo tomado tendo como base a quantidade de operações que serão necessárias, a memória necessária para ser realizado a criptoanálise, e a quantidade de dados que podem ser gerados.

12 LOOKUP TABLES

Lookup Tables (LT), ou tabelas de pesquisa em tradução literal, são tabelas pré-computadas limitadas pelo tamanho, e os caracteres presentes na tabela. As LTs são geralmente usadas para a quebra de senhas armazenadas em *hash*.

As LTs usadas contra as senhas de redes sem fio possuem um tratamento a mais: o valor do *salt*, uma espécie de “tempero” na senha, resultando em um *hash* diferente da senha pura. O *salt*, no caso de redes sem fio nada mais é que o ESSID, o nome da rede sem fio. Para as redes WPA/PSK e WPA2/PSK é computado o algoritmo PBKDF2 para ser gerado a *Lookup Table*.

Erickson (2009) afirma que com esta estrutura de dados “qualquer senha poderia ser *crackeada* no tempo que se leva para pesquisar”. Porém, para que seja possível esta rapidez na pesquisa, é necessário aceitar uma troca por espaço, sendo que o espaço ocupado é consideravelmente grande.

13 LINUX

De acordo com Debian (2013), o Linux na verdade é o *kernel* de um sistema operacional, que tem com modelo o sistema operacional Unix. O *kernel*

sozinho não possui muita funcionalidade, é necessário uma série de ferramentas que são capazes de utilizar tal *kernel*. Estas ferramentas são providas através do GNU Project, projeto iniciado por Richard Stallman com intuito de criar um sistema operacional. Sendo assim, o nome do sistema operacional que a grande maioria chama de Linux, na verdade chama-se GNU/Linux.

Existem também as chamadas distribuições, que são o conjunto do *kernel* e ferramentas escolhidas pela equipe de desenvolvimento, cada qual com seus interesses. As mais conhecidas hoje são *Debian, openSuse, Fedora* e *Ubuntu*. A distribuição *Ubuntu* por exemplo, é totalmente baseada na *Debian*, enquanto que a distribuição que iremos utilizar neste trabalho chamada *BackTrack*, é baseada no *Ubuntu*.

13.1 BACKTRACK

Distribuição Linux baseada nas distribuições *Whoppix, IWHAX* e *Auditor* e mantida pela Offensive Security, a qual declara que foi feito para *pentesters* com o objetivo de ser um pacote completo para ser usado em auditorias. De acordo com Giavaroto e Santos(2013), a distribuição conta hoje com mais de 300 ferramentas voltadas para pentest com o adicional que algumas certificações na área de segurança utilizam o BackTrack como ferramenta principal.

13.1.1 CRUNCH

Este *software* é capaz de criar arquivos de texto contendo todas as combinações e permutações possíveis de um grupo de caracteres determinado pelo usuário, assim como o tamanho mínimo e máximo dos resultados.

13.1.2 GENPMK

GenPMK tem a função de criar *Lookup Tables* a partir de um dicionário informado pelo usuário e também o SSID da rede, pois este é usado como *salt* durante o calculo dos *hashes*.

13.1.3 SUITE AIRCRACK-NG

Esta suíte é um conjunto de *softwares* desenvolvido para auditoria de redes sem fio. No total possui mais de 15 *softwares* dentro do pacote, onde apenas três deles serão utilizados nesta pesquisa, sendo eles o *aireplay-ng*, *airmon-ng* e *airodump-ng*.

13.1.3.1 AIRMON-NG

Habilita e desabilita o modo monitor de uma placa de rede sem fio. No modo monitor, a placa de rede é capaz de monitorar todo o tráfego recebido em uma rede *wireless*, sem precisar de associação a esta rede.

13.1.3.2 AIRODUMP-NG

Utilizado para captura de pacotes, com diversos detalhes, como o endereço MAC do AP e das estações conectadas, força do sinal, número de pacotes de dados trafegados no AP e enviados de cada estação, protocolo de autenticação utilizado, o SSID do AP, entre outros dados. Uma opção bastante importante que será utilizado durante a pesquisa é a de salvar os dados monitorados.

13.1.3.3 AIREPLAY-NG

Sua principal função é de gerar tráfegos e injetar pacotes. Existem hoje dez tipos de ataques possíveis com esta ferramenta, sendo a desassociação, autenticação falsa e teste de injeção apenas alguns exemplos.

13.1.4 WIRESHARK

Anteriormente conhecido como *Etheral*, o *Wireshark* é o principal analisador de tráfego de rede do mundo, com organização por protocolos em tempo real com interação durante a captura. De acordo com o *site* oficial do

software wireshark.org, o analisador foi desenvolvido originalmente por Gerald Combs e conta hoje com mais de oitocentas contribuições para que o *software* continue atualizado.

13.1.5 COWPATTY

O *software coWPAtty* é projetado para auditar a seleção de chave pré-compartilhada (PSK) para redes WPA baseado no protocolo TKIP, segundo definição do próprio desenvolvedor Joshua Wright (WIRELESSDEFENCE.ORG, 2013), autor de alguns capítulos do livro Wireshark & Ethereal Network Protocol Analyzer Toolkit.

14 METODOLOGIA

14.1 TIPO DE PESQUISA

O trabalho inicialmente envolveu uma pesquisa exploratória, pois de acordo com Gil (2008), o objetivo desta é familiarizar-se com um assunto ainda pouco conhecido, pouco explorado. Ao final de uma pesquisa exploratória, o pesquisador estando mais familiarizado com o assunto, permite que uma maior compreensão e precisão sejam alcançadas no assunto pouco explorado. Sendo assim, foram realizados testes laboratoriais onde tudo foi documentado para que posteriormente os resultados fossem consistentes com a realidade observada.

Por ser um tipo de pesquisa muito específica, e um assunto pouco explorado, existe uma carência de informações específicas nesta área com características acadêmicas, consequentemente uma maior dificuldade durante a pesquisa bibliográfica. Tal dificuldade deve ser vista pelo pesquisador como uma forma de ajudar o meio acadêmico e não como motivo para desistência. A contribuição com o meio acadêmico pode permitir futuramente que mais pesquisas sejam realizadas nesta área tão pouco explorada.

14.2 RECURSOS

Foi utilizado para a realização deste trabalho um notebook da marca Lenovo modelo G550, com processador *Dual Core* T4300 2.10GHz da Intel, com 4Gb de memória RAM, disco rígido de 320Gb, com *Windows Seven Ultimate* 64 *bits* e *Backtrack 5 R3* instalados em *dual-boot*.

O laboratório contou também com um roteador da marca TP-Link, modelo número TL-WR741ND, 150Mbps, 2.4GHz o qual foi configurado com senhas contendo apenas números com 8 posições, por ser o mínimo de caracteres necessários para a configuração de uma senha para o protocolo de rede sem fio escolhido, o WPA2/PSK que supostamente deveria ser o mais seguro dentre os protocolos existentes.

14.3 EXECUÇÃO

Para a execução do ataque a rede sem fio foi utilizado exclusivamente o sistema operacional *Backtrack 5 R3*, já que este possui todas as ferramentas necessárias em sua instalação padrão. Foram realizados ataques ao roteador, onde suas senhas serão definidas a partir de um *software* simples que foi desenvolvido em Java para retornar oito caracteres numéricos aleatórios utilizando o código-fonte apresentado na Figura 21, que cria um vetor de oito posições para cada senha que for gerada, e alimenta este vetor com números aleatórios e ao final, imprime esta sequência de números na tela.

```
class GeradorNumerosAleatorios {

    public static void main(String[] args) {

        for (int qtdeSenhas = 0; qtdeSenhas < 10; qtdeSenhas++) {
            // Vetor que armazenará os 8 números da senha
            int vetor[] = new int[8];

            for (int qtdeCaracter = 0; qtdeCaracter < 8; qtdeCaracter++) {
                // Coloca no vetor um número aleatório
                vetor[qtdeCaracter] = (int) (Math.random() * 10);
            }

            for (int indice : vetor) {
                // Percorre o vetor, imprimindo o valor de cada índice
                System.out.print(indice);
            }

            // Quebra de linha
            System.out.println();
        }
    }
}
```

Figura 21 - Código-fonte utilizado para a geração das senhas
Fonte: Elaborada pelo Autor (2013)

Com o código-fonte pronto, o arquivo foi salvo, e no terminal do *Backtrack 5 R3* foi utilizado o comando “javac GeradorNumerosAleatorios.class” para compilar o programa e então “java GeradorNumerosAleatorios” para a execução do programa. A Figura 22 mostra a saída que o programa forneceu.

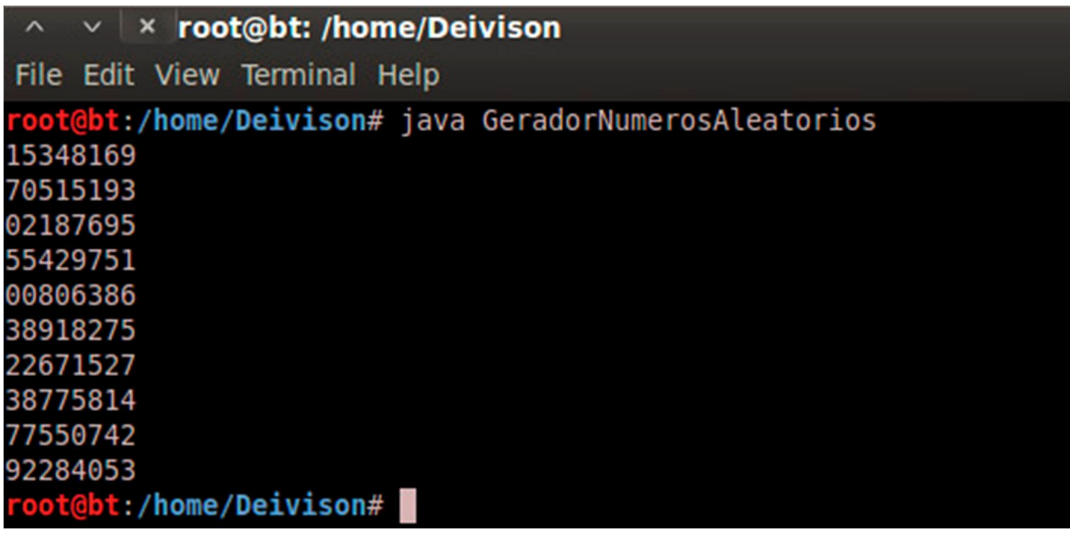

Figura 22 - Saída gerada pelo software
Fonte: Elaborada pelo Autor (2013)

Como foi utilizado o conceito de ataque de dicionário, foi criado então o dicionário utilizando o *software Crunch*. A execução deste *software* requer alguns parâmetros como o tamanho mínimo e máximo de caracteres e quais caracteres que serão utilizados. Outros parâmetros são opcionais tais como o arquivo de saída onde irá conter os dados, qual o tamanho lógico máximo para o arquivo, dividindo o arquivo de saída, entre outros. Este *software* cria todas as combinações possíveis sobre as regras que foram passadas como parâmetro. Para esta pesquisa, foi utilizado um tamanho fixo de oito caracteres, sendo estes caracteres apenas numéricos, resultando em cem milhões de combinações. A razão desta restrição é pelo fato de que o dicionário a ser criado deixaria o arquivo muito grande, aumentando ainda mais este tamanho com o processamento que irá receber. O programa Crunch foi executado a partir de um *script* conforme mostra a Figura 23, passando parâmetros para dividir a saída em arquivos com cinco milhões de palavras cada, pois devido a grande quantidade de informações causaria problemas na geração das *Lookup Tables* posteriormente. Os *scripts* utilizados neste trabalho utilizam o *software*

Scrot para tirar uma *screenshot*, para que então fosse possível calcular o tempo tomado durante a execução do *software* em questão.

```
#!/bin/bash

clear;
scrot /home/Deivison/tcc/screenshots/wordlistBegin.png;

/pentest/passwords/crunch/crunch 8 8 0123456789 -o START -c 5000000;

scrot /home/Deivison/tcc/screenshots/wordlistEnd.png;
```

Figura 23 - *Script* utilizado para a geração dos arquivos de dicionário
Fonte: Elaborada pelo Autor (2013)

Tendo o dicionário em mãos foi gerado as *Lookup Tables* com o *software genPMK*. Para esta pesquisa, foi obrigatório o envio de três parâmetros ao *software*, sendo eles o arquivo de dicionário criado após o comando “-f”, o arquivo de saída com o comando “-d” e o comando “-s” seguido pelo nome do ponto de acesso que será utilizado como *salt* durante a criação dos *hashes*.

Nesta etapa, foram utilizados *scripts* para a geração dos arquivos devido ao tempo que esta seria necessário para ser concluído. Sendo assim, foram criados vários *scripts* para gerar duas *Lookup Tables* por vez, cada uma contendo cinquenta mil possíveis senhas. A Figura 24 mostra um arquivo de *script* utilizado para gerar as *Lookup Tables*.

```
#!/bin/bash

clear;
scrot /home/Deivison/tcc/screenshots/hashesBegin19.png;

genpmk -f /home/Deivison/tcc/wordlist/90000000-94999999.txt -d
/home/Deivison/tcc/hashed\ wordlist/hashes19.txt -s LabTCC;

sleep 3;
scrot /home/Deivison/tcc/screenshots/hashesEnd19.png;

sleep 2;

clear;
scrot /home/Deivison/tcc/screenshots/hashesBegin20.png;

genpmk -f /home/Deivison/tcc/wordlist/95000000-99999999.txt -d
/home/Deivison/tcc/hashed\ wordlist/hashes20.txt -s LabTCC;

sleep 3;
scrot /home/Deivison/tcc/screenshots/hashesEnd20.png;
```

Figura 24 - *Script* utilizado para a geração das *Lookup Tables*
Fonte: Elaborada pelo Autor (2013)

Após a criação da *Lookup Table*, foi utilizada a suíte Aircrack-ng, que nada mais é que um conjunto de *software*s com objetivos voltados às redes sem fio. Mas antes, foi necessário habilitar a placa *wireless* do *notebook* no modo monitor com a ajuda do *software* airmon-ng apresentado na Figura 25, para que esta possa captar os pacotes trafegando. Os parâmetros requeridos deste *software* são apenas dois: a ação e a interface de rede. Tomando como base o comando utilizado, o parâmetro “start” indica que será iniciado um processo de monitoramento sobre o segundo parâmetro “wlan0” que no caso era a interface de rede que foi trabalhada. Uma nova interface é criada especificamente para a utilização no modo monitor, que no caso foi “mon0”.

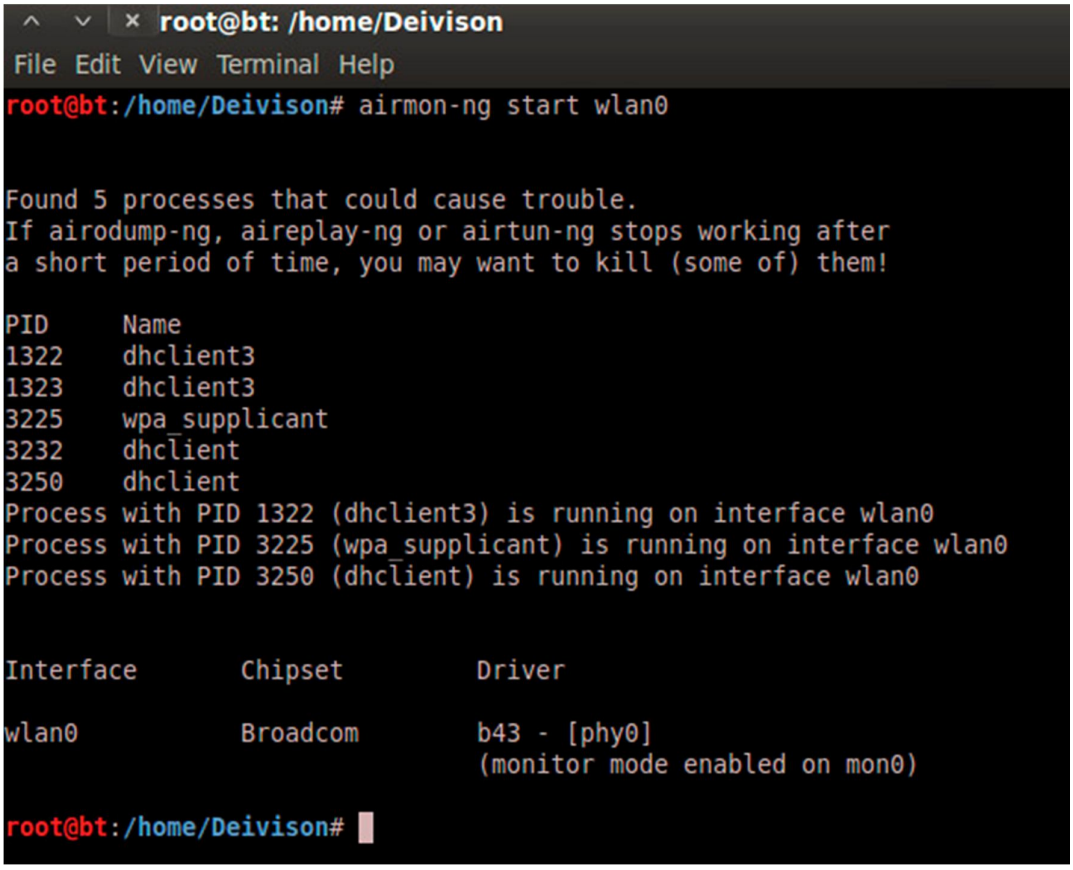

Figura 25 - Airmon-ng
Fonte: Elaborada pelo Autor (2013)

Com a placa em modo monitor, foi utilizado o programa airodump-ng, monitorando todas as redes sem fio no alcance da placa, a quantidade de tráfego em cada uma delas, a quantidade de tráfego, e também estações conectadas juntamente com a rede que pertence. Todas essas informações

podem ser vistas na Figura 26. O comando utilizado foi apenas “airodump-ng mon0”, onde “mon0” é a interface em modo monitor que está sendo utilizada.

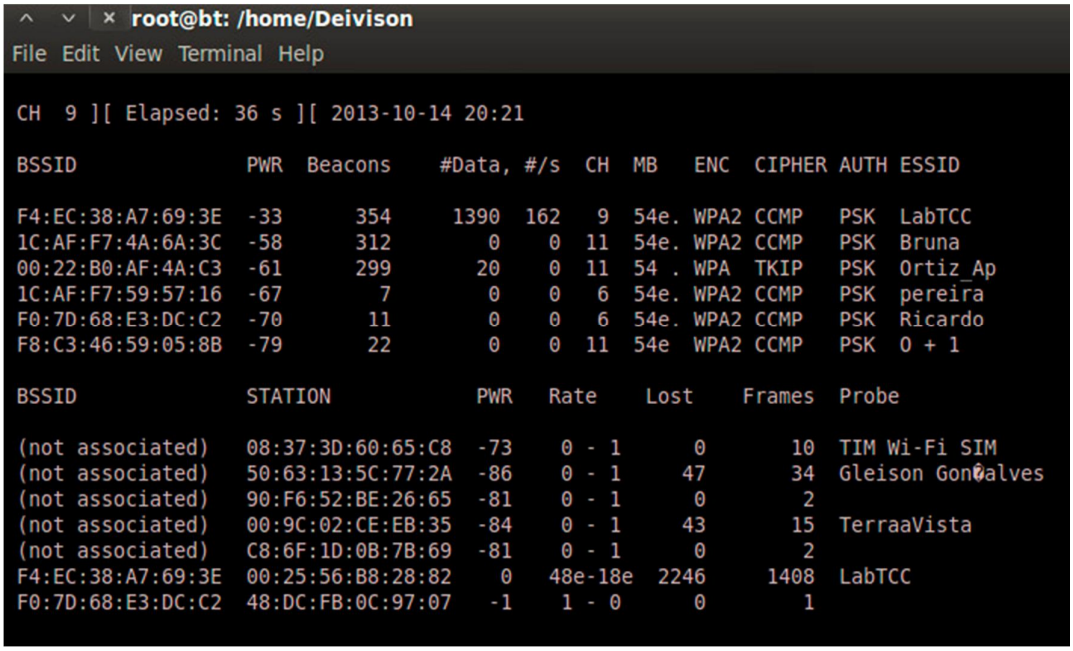

Figura 26 - Airodump-ng sem filtros
Fonte: Elaborada pelo Autor (2013)

Encontrando o alvo na lista de redes apresentada, o programa airodump-ng é novamente utilizado passando os parâmetros para somente monitorar especificamente a rede alvo com objetivo de capturar o *handshake* com a utilização do comando “-bssid”, especificando o BSSID da rede alvo, salvar todo o tráfego de dados em um arquivo através do comando “-w” seguido pelo caminho do arquivo e por último é indicado qual interface de rede será utilizada. O comando para esta ação foi: “airodump-ng -bssid F4:EC:38:A7:69:3E -w ./tcc/passwords/00806386/hanshake/withoutAircrack-ng mon0”.

Neste momento, é necessário que uma estação se conecte a rede para que seja possível capturar o *handshake*. Neste trabalho foram utilizadas as duas formas possíveis de captura do *handshake*. A primeira consiste na estação que está conectada à rede, se desconectar por livre e espontânea vontade e conectar-se novamente, enquanto a segunda forma seria o atacante enviar pacotes de desautenticação para a estação com o auxílio do programa aireplay-ng. Isto fará com que a mesma se reconecte com o ponto de acesso,

sendo necessário ser realizado o processo de *handshake* entre os dois dispositivos. A Figura 27 ilustra a saída apresentada pelo programa com o envio dos pacotes de desautenticação. Para este trabalho foi utilizado o seguinte comando: “aireplay-ng -0 10000 -a F4:EC:38:A7:69:3E -c 00:25:56:B8:28:82 mon0”, onde o parâmetro “-0” significa sinal de desautenticação, o valor “10000” é a quantidade máxima de pacotes a serem enviados, “-a” é o Endereço MAC do AP da rede, e “-c” é o Endereço MAC da estação que está sendo monitorada para ser capturado o *handshake*.

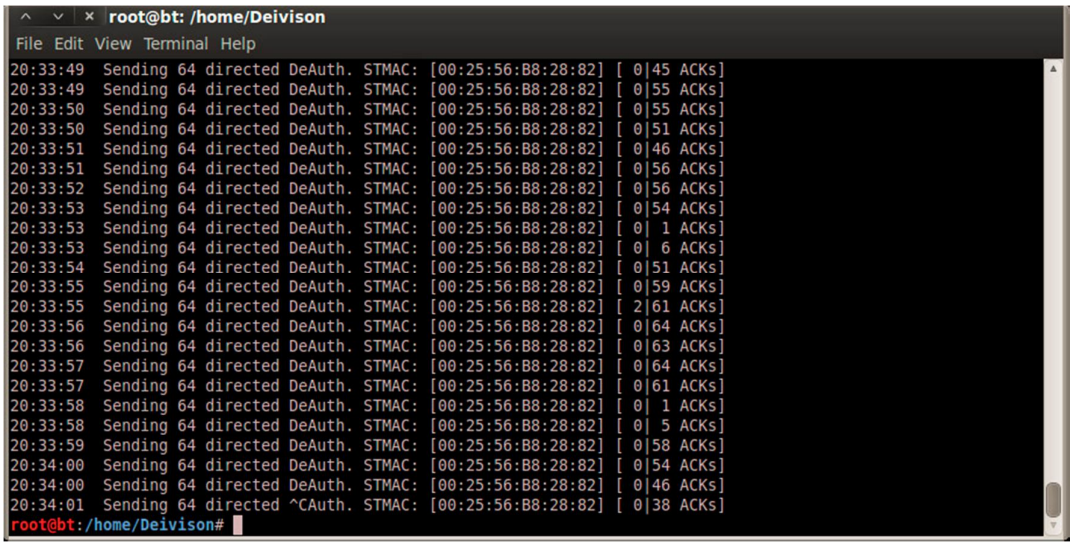

Figura 27 - Saída apresentada pelo Aireplay-ng
Fonte: Elaborada pelo Autor (2013)

Quando uma estação se conecta a rede, é capturado o *handshake* e uma mensagem é apresentada no programa airodump-ng confirmando a captura conforme visto na Figura 28. Deste ponto em diante, o programa airodump-ng pode ser encerrado e o arquivo contendo o *handshake* será salvo no caminho especificado anteriormente.

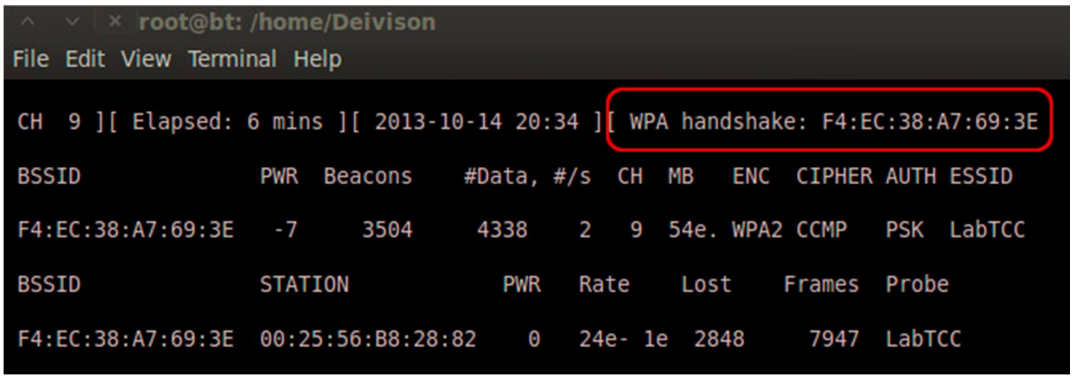

Figura 28 - Airodump-ng com a mensagem de captura de *handshake*
Fonte: Elaborada pelo Autor (2013)

Para a realização dos ataques, foi montada uma estrutura de arquivos para melhor organização. A Figura 29 demonstra esse arquivo, contendo 10 diretórios, cada uma com uma senha gerada. Dentro de cada um desses diretórios existem outros dois diretórios, sendo eles: *screenshots*, *handshake*. No primeiro estão armazenados todos os *screenshots* disparados automaticamente pelos *scripts* desenvolvidos. No diretório *handshake* está armazenado os arquivos gravados pelo programa airodump-ng, contendo o *handshake* capturado durante o monitoramento da rede alvo.

Figura 29 - Estrutura de arquivos
Fonte: Elaborada pelo Autor (2013)

A Figura 30 apresenta parte do *script* utilizado para a quebra de senha utilizando *Lookup Table*. Como as *Lookup Tables* estão divididas em 20 arquivos, foi necessário construir um *script* com uma estrutura de "if" aninhados, para quando o *script* encontrasse a senha desejada, então todo o restante do código não fosse executado. O comando executado para ser realizado o ataque necessita de três parâmetros: "-d" indicando qual o arquivo da *Lookup Table*; "-r" para apontando para o arquivo com a extensão ".cap" contendo o *handshake* capturado; e "-s" para informar o SSID da rede. O *script* completo pode ser visualizado no anexo ANEXO A – *SCRIPT* COMPLETO DE ATAQUE UTILIZANDO LOOKUP TABLE.

Para cada senha gerada, foram realizados dez ataques, para posteriormente serem calculadas as médias de tempo e velocidade e comparadas com as médias dos ataques utilizando dicionário.

```
if cowpatty -d ./tcc/hashed\ wordlist/hashes19.txt -r
./tcc/passwords/00806386/handshake/withoutAircrack-ng-01.cap -s LabTCC == true
then
 exit
else
 if cowpatty -d ./tcc/hashed\ wordlist/hashes20.txt -r
./tcc/passwords/00806386/handshake/withoutAircrack-ng-01.cap -s LabTCC == true
 then
  exit
 else
  echo "SENHA NÃO ENCONTRADA"
 fi
fi
```

Figura 30 - Trecho do *script* utilizado para o ataque utilizando *Lookup Table*
Fonte: Elaborada pelo Autor (2013)

Para os ataques utilizando dicionário, foram utilizados os mesmos arquivos de *handshake* utilizados para os ataques de *Lookup Tables*. Os arquivos gerados pelo software Crunch também foram reaproveitados utilizando um *script* que contém o comando "cat" redirecionando a saída padrão para um arquivo, conforme mostra a Figura 31. Isto fará com que o conteúdo de todos os 20 arquivos que foram gerados anteriormente, seja passado para um único arquivo.

```
#!/bin/bash

clear;
scrot /home/Deivison/tcc/screenshots/joinWordlistBegin.png;

cat *.txt >> wordlist.lst

scrot /home/Deivison/tcc/screenshots/joinWordlistEnd.png;
```

Figura 31 - *Script* para juntar todos arquivos de dicionário em apenas um
Fonte: Elabora pelo Autor (2013)

Para a realização dos ataques de dicionário também foi utilizado o programa coWPAtty, mudando apenas os parâmetros passando o arquivo de texto contendo as possíveis senhas com o parâmetro “-f”, em vez de passar as *Lookup Tables* com o parâmetro “-d”. A Figura 32 apresenta o comando utilizado para os ataques de dicionário.

```
#!/bin/bash

COUNT=1 while [ $COUNT -le 5 ] do

 clear;
 echo "Iniciando Teste - Dicionário - $COUNT/5";
 sleep 1;
 scrot /home/Deivison/tcc/passwords/00806386/screenshots/dictionary_begin_0$COUNT.png;

 cowpatty -f ./tcc/wordlist/wordlist.lst -r
./tcc/passwords/00806386/handshake/withoutAircrack-ng-01.cap -s LabTCC

 sleep 1;
 scrot /home/Deivison/tcc/passwords/00806386/screenshots/dictionary_end_0$COUNT.png;
 sleep 2;
 ((COUNT++))

done
```

Figura 32 - *Script* para realização de cinco ataques de dicionário seguidos
Fonte: Elaborada pelo Autor (2013)

15 RESULTADOS

Para a coleta dos dados, foi analisada cada *screenshot* disparada automaticamente pelos *scripts* de ataque e calculado a diferença de tempo entre o início e o fim do ataque e tabulado para uma melhor visualização dos dados.

A Figura 33 apresenta os dados coletados dos ataques utilizando *Lookup Tables*, enquanto a Figura 34 apresenta dados dos ataques de dicionário, observando que as senhas seguidas de um asterisco significam que foram resultados calculados com base nas médias dos ataques realizados.

Para detalhes, veja os Anexos ANEXO B - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO LOOKUP TABLE e ANEXO C - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO DICIONÁRIO.

Médias – Lookup Table		
Senhas	**Tempo (segundos)**	**Velocidade (senhas/segundo)**
00806386	6	134.397,67
02187695	16,3	134.214,42
15348169	114,5	134.045,14
22671527	169,3	133.913,33
38775814	287	135.107,37
38918275	289,4	134.479,18
55429751	410,9	134.898,40
70515193	520,9	135.371,84
77550742	575,6	134.730,27
92284053	683,6	134.997,15

Figura 33 - Médias coletadas dos ataques utilizando Lookup Table
Fonte: Elaborada pelo Autor (2013)

Médias – Dicionário		
Senhas	**Tempo**	**Velocidade (senhas/segundo)**
00806386	01:01:46 h	217,57
02187695	02:47:22 h	217,85
15348169*	19:34:58 h	217,71
22671527*	1 dia 04:55:36 h	217,71
38775814*	2 dia 01:28:27 h	217,71
38918275*	2 dias 01:39:22 h	217,71
55429751*	2 dias 22:43:23 h	217,71
70515193*	3 dias 17:58:15 h	217,71
77550742*	4 dias 02:56:51 h	217,71
92284053*	4 dias 21:44:45 h	217,71

Figura 34 - Médias coletadas dos ataques utilizando Dicionário
Fonte: Elaborada pelo Autor (2013)

Com base nos dados acima, fica evidente a diferença discrepante entre os dois métodos. Os ataques utilizando *Lookup Tables* comparados com os ataques utilizando dicionário possuem uma velocidade acima de seiscentas

* Senhas das quais os valores de tempo foram calculados com base na média dos testes realizados.

vezes, devido ao fato de as senhas já estarem pré-computadas (com seus respectivos *hashes* já gerados), tornando o tempo de pesquisa muito mais rápido comparado com os ataques de dicionário em que a geração do *hash* é feita em tempo real. Os dados apresentados em relação às *Lookup Tables* não consideram o tempo de geração das mesmas que tomaram em média seis horas para cada cinqüenta milhões de palavras.

Com os resultados obtidos dos ataques, foram feitas pesquisas para encontrar formas de se proteger de ataques utilizando *Lookup Tables* e também dicionário. As imagens a seguir deste capítulo estão baseadas nas configurações do roteador da marca TP-Link, modelo número TL-WR741ND, 150Mbps, 2.4GHz que foi utilizado durante o trabalho. Outros modelos de roteadores de uso domésticos possuem configurações semelhantes.

Durante o processo de geração de uma senha, o proprietário da rede deve primeiramente procurar sempre por uma senha grande com a mistura de letras, números e se possível caracteres especiais. Essas condições aliadas com a falta de padronização das senhas dificultam qualquer ataque malicioso contra a rede, principalmente se o ataque for utilizando *Lookup Tables*, já que estes são baseados em um dicionário, que geralmente encontram-se palavras comuns com poucas permutações envolvendo números. Dicionários criados pelo próprio atacante exigem muito tempo para que se torne eficiente, caso contrário, o dicionário terá um alcance bastante limitado de senhas.

Outro ponto importante para manter uma senha segura, é não se acomodar com a mesma senha por muito tempo, ou seja, de tempo em tempo é viável a troca de senha para quando uma senha for comprometida sem que os usuários da rede percebam, uma nova senha possa substituir a antiga e continuar oferecendo um ambiente seguro. Para a configuração de senha, é necessário acessar o painel de segurança da rede sem fio, conforme apresentado na Figura 35.

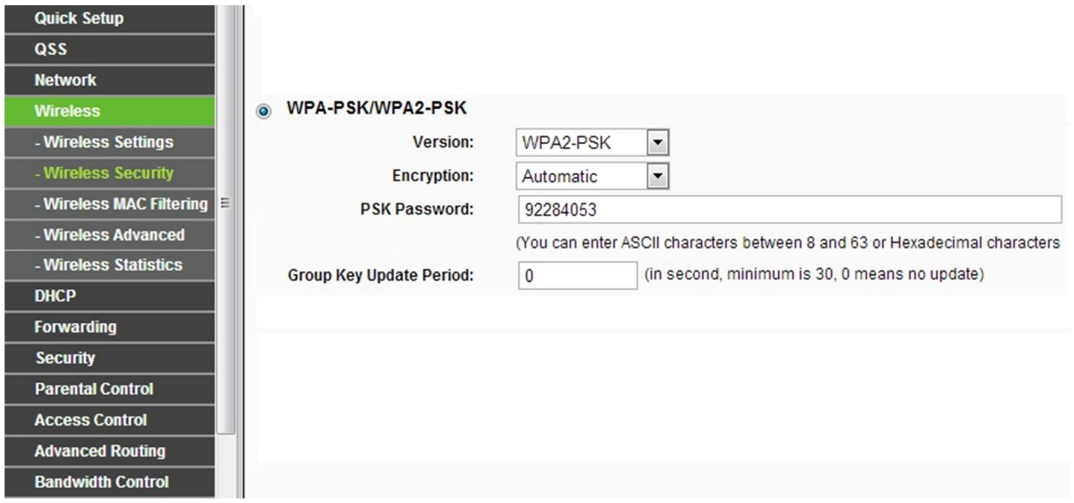

Figura 35 - Configuração de segurança da rede sem fio
Fonte: Elaborada pelo Autor (2013)

Para os ataques específicos que utilizam *Lookup Tables*, renomear o nome da rede (SSID) é uma maneira simples de tornar o ataque ineficiente, pois as *Lookup Tables* são criadas especificamente para o SSID escolhido pelo atacante. Como as gerações das *Lookup Tables* consomem um tempo considerável, é inviável para o atacante ficar gerando *Lookup Tables* para cada SSID que for alterado.

Outra dica em relação ao SSID é desativar nas configurações do roteador o envio de pacotes em broadcast conforme mostra na Figura 36, ou seja, qualquer usuário fora da rede não conseguirá saber o nome da rede de imediato. Embora existam técnicas para descobrir o nome da rede sem fio em situações como esta descrita, o atacante precisará de mais tempo até que se consiga o SSID da rede que deseja atacar, podendo levar até a desistência do ataque devido ao trabalho que será necessário para a quebra de senha.

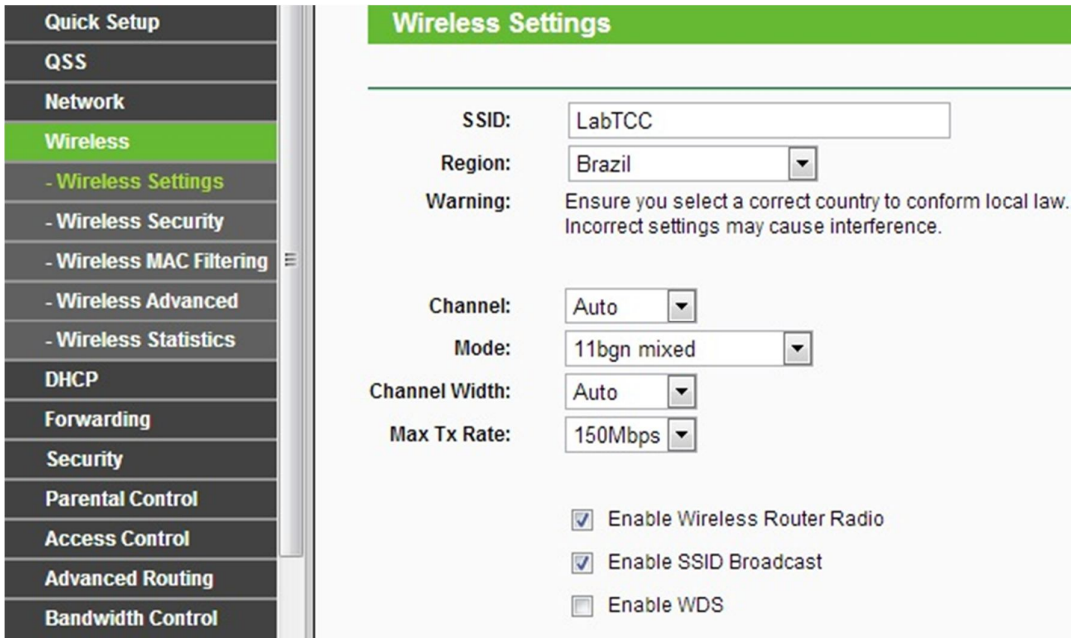

Figura 36 - Configuração de rede sem fio do roteador
Fonte: Elaborada pelo Autor (2013)

Outro ponto importante é alterar a senha padrão de administrador da página de configuração do roteador, porque uma vez que o atacante invade a rede o mesmo pode tentar alterar as configurações do roteador que em caso de estar configurado com a senha padrão de administração pode ser facilmente adivinhada pelo atacante.

16 DIFICULDADES ENCONTRADAS

Durante o processo de preparação e também de ataques, foram encontradas algumas dificuldades que ao fim puderam ser resolvidas de outros modos.

A primeira dificuldade encontrada foi técnica, pois a princípio foi criado o dicionário com o software Crunch sem dividir em vinte arquivos, e durante o processo de geração das *Lookup Tables*, o notebook utilizado no trabalho não suportava todo o processamento e acaba esquentando a ponto de reiniciar o *notebook*.

Outra dificuldade encontrada foi após a decisão de dividir o arquivo de dicionário em vinte arquivos e então fazer os *scripts* de forma que fossem

percorridos apenas os arquivos necessários até encontrar a senha. Devido à falta de conhecimento com Shell *Script*, foi necessário um pouco de pesquisa e testes isolados para verificar a eficiência do *script*.

Por fim, a última dificuldade encontrada foi durante os ataques de dicionário, como exigem muito processamento, não era possível realizar ataques para todas as senhas geradas, então para encontrar o tempo tomado para cada senha, foram realizados cálculos tomando como base a média dos ataques bem sucedidos.

17 CONSIDERAÇÕES FINAIS

Após os testes realizados e analisado os resultados obtidos dos mesmos foi possível alcançar os objetivos deste trabalho, verificando que a quebra de senha utilizando *Lookup Tables* é incrivelmente mais rápido, porém existe o tempo para sua geração, que levou em torno de 13 horas para a geração de cem mil combinações. Sendo assim, se as *Lookup Tables* criadas forem utilizadas para apenas um ataque, então esse tipo de ataque torna-se ineficiente. Entretanto se a mesma *Lookup Table* for utilizada para diversos ataques, então existe uma grande vantagem de utilizá-las em relação aos dicionários.

Com o uso das recomendações demonstradas no Capítulo **Erro! Fonte de referência não encontrada.**, a segurança de uma rede sem fio é elevada consideravelmente contra os ataques utilizando *Lookup Tables*, já que estes estão restritos para um único SSID, e não contém todas as possibilidades de senhas possíveis.

18 TRABALHOS FUTUROS

Com a utilização de clusters, é possível que a velocidade tanto para a geração das *Lookup Tables* quanto para a quebra da senha aumente

drasticamente, devido ao grande poder de processamento fornecido, podendo ser verificado então a eficiência de utilizar clusters para este fim.

É possível também ser feito um comparativo entre *Lookup Tables* e *Rainbow Tables*, outra técnica utilizada que geralmente ocupa menos espaço em disco, e tem um sistema de busca mais complexo que as *Lookup Tables*.

Pelo fato de os *softwares* utilizados neste trabalho serem de código-fonte aberto, a verificação, estudo do seu funcionamento e alteração dos programas é possível. Sendo assim, uma alteração para melhorar o desempenho seria no *software* genpmk para que ao término a *Lookup Table* seja organizada em estruturas de árvores, sendo cada nó um caractere do *hash*. Já no programa cowpatty alterar para não comparar o *hash* inteiro, mas sim por partes, começando pelo primeiro caractere do *hash* encontrado no *handshake*. Este caractere vai ser analisado para ver em qual árvore será feito a pesquisa, eliminando todos os *hashes* que não começam com aquele mesmo caractere, conforme mostra a Figura 37.

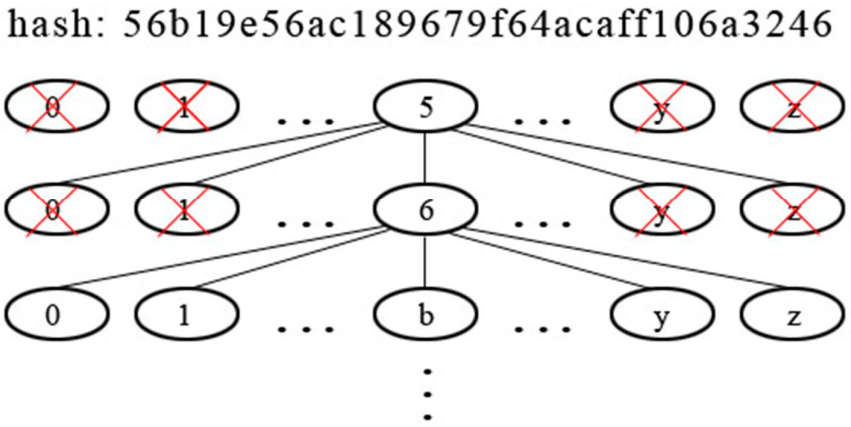

Figura 37 - Representação de uma árvore para melhora de desempenho
Fonte: Elaborada pelo Autor (2013)

REFERÊNCIAS

ABNT - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR ISO/IEC 17799**: Tecnologia da Informação - Técnicas de segurança - código de prática para a gestão da segurança da informação. [S. l.]: Associação Brasileira de Normas Técnicas, 2005. 120 p.

CISCO – NETWORKING ACADEMY. **CCNA Exploration 4.0**. Disponível em <https://135603.netacad.com/courses/24731>. Acesso em 02 mar. 2013.

CERT.BR. **Incidentes Reportados ao CERT.br:** Janeiro a Dezembro de 2012. Disponível em: <http://www.cert.br/stats/incidentes/2012-jan-dec/total.html>. Acesso em: 15 mar. 2013.

CLAVIS. **Webinar # 16 – Ataques de Força Bruta:** Método Dicionário, Híbridos e Rainbow Tables. Disponível em: <http://www.blog.clavis.com.br/webinar-16-ataques-de-forca-bruta-metodo-dicionario-hibridos-e-rainbow-tables/>. Acesso em: 04 mar. 2013.

DEBIAN. **O que é GNU/Linux?** Disponível em: <http://www.debian.org/releases/stable/s390/ch01s02.html.pt>. Acesso em: 29 maio 2013.

DREAMSTIME. **Royalty Free Stock Photography:** Linear bus with laptops. Disponível em: <http://www.dreamstime.com/royalty-free-stock-photography-linear-bus-laptops-image17092467>. Acesso em: 30 maio 2013a.

DREAMSTIME. **Royalty Free Stock Photography:** Ring network topology with laptops. Disponível em: <http://www.dreamstime.com/royalty-free-stock-images-ring-network-topology-laptops-image17238509>. Acesso em: 30 maio 2013b.

DREAMSTIME. **Royalty Free Stock Photography:** Star network topology with laptops. Disponível em: <http://www.dreamstime.com/royalty-free-stock-photography-star-network-topology-laptops-image17238537>. Acesso em: 30 maio 2013c.

DREAMSTIME. **Royalty Free Stock Photography:** Two connected laptops. Disponível em: <http://www.dreamstime.com/royalty-free-stock-images-two-connected-laptops-image21218169>. Acesso em: 30 maio 2013d.

ERICKSON, Jon. **Hacking.** São Paulo: Digerati Books, 2009.

FLORIANO, Guilerme Martinez. **Camouflagedsecurity system:** protótipo de *software* que emprega técnicas de criptografia, assinatura digital e esteganografia para comunicação segura. Porto Alegre, 2007. Disponível em: < http://www.uniritter.edu.br/graduacao/informatica/sistemas/downloads/tcc2k7-

2/GuilhermeMartinezFloriano.pdf>. Acesso em: 03 mar 2011.

FOROUZAN, Behrouz A.. **Comunicação de Dados e Redes de Computadores.** 4. ed. São Paulo: Mcgraw-hill, 2008.

GIAVAROTO, Silvio César Roxo; SANTOS, Gerson Raimundo Dos.**Backtrack Linux:** Auditoria e Testes de Invasão em Redes de Computadores. Rio de Janeiro: Ciência Moderna, 2013.

GIL, Antonio Carlos. **Como elaborar projetos de pesquisa**. 5. ed. São Paulo: Atlas, 2008.

HAINES, Brad. **Seven Deadliest Wireless Technologies Attacks**. Burlington: Elsevier, 2010. Disponível em <http://em3rgency.com/wp-content/uploads/2012/12/Seven-Deadliest-Wireless-Technologies-Attacks.pdf> Acesso em: 20 maio 2013.

IEEE STANDARDS ASSOCIATION. **IEEE Standard for Information technology - Telecommunications and information exchange between systems Local and metropolitan area networks - Specific requirements:** Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Disponível em: <http://standards.ieee.org/about/get/802/802.11.html>. Acesso em: 21 abr. 2013.

IETF. **RFC 2104:** HMAC: Keyed-Hashing for Message Authentication. Disponível em: <http://www.ietf.org/rfc/rfc2104.txt>. Acesso em: 04 mar. 2013a.

IETF. **RFC 2898:** PKCS #5: Password-Based Cryptography Specification Version 2.0. Disponível em: <http://www.ietf.org/rfc/rfc2898.txt>. Acesso em: 26 maio 2013b.

KEVIN BENTON. **The Evolution of 802.11 Wireless Security.** Disponível em: <http://itffroc.org/pubs/benton_wireless.pdf>. Acesso em: 27 fev. 2013.

KUROSE, James F.; ROSS, Keith W.. **Redes de Computadores e a Internet:** Uma abordagem top-down. São Paulo: Pearson Addison Wesley, 2006.

LYRA, Maurício Rocha. **Segurança e Auditoria em Sistema de Informação.** Rio de Janeiro: Ciência Moderna, 2008.

MITNICK, Kevin D.; SIMON, William L.. **MITNICK - A Arte de Enganar:**Ataques de Hackers: Controlando o Fator Humano na Segurança da Informação. São Paulo: Pearson Education, 2003.

MORENO, Edward David; PEREIRA, Fábio Dacêncio; CHIARAMONTE, Rodolfo Barros. **Criptografia em Software e Hardware.** São Paulo: Novatec, 2005.

NIST. **Recommendation for Password-Based Key Derivation:** Part 1: Storage Applications. Disponível em: <http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf>. Acesso em: 27 fev. 2013.

OFFENSIVE SECURITY LTD. **Offensive Security Training and Services.** Disponível em: <http://www.offensive-security.com/community-projects/backtrack-linux/>. Acesso em: 1 maio 2013.

PETERSON, Larry L.; DAVIE, Bruce S.. **Redes de Computadores:** Uma Abordagem de Sistemas. Rio de Janeiro: Elsevier, 2004.

RUFINO, Nelson Murilo de O.. **Segurança em Redes sem Fio:** Aprenda a proteger suas informações em ambientes Wi-fi e Bluetooth. 3. ed. São Paulo: Novatec, 2011.

STALLINGS, William. **Criptografia e Segurança de Redes.** São Paulo: Pearson Prentice Hall, 2008.

STALLINGS, William. **Redes e Sistemas de Comunicação de Dados:**Teoria e Aplicações Corporativas. Rio de Janeiro: Elsevier, 2005.

TANENBAUM, Andrew S.. **Redes de Computadores.** 4. ed. Rio de Janeiro: Elsevier, 2003.

THOMAS, Tom. **Segurança de Redes:** Primeiros Passos. Rio de Janeiro: Ciência Moderna, 2007.

ULBRICH, Henrique Cesar; DELLA VALLE, James. **Universidade H4ck3r.** São Paulo: Digerati Books, 2004.

WIRELESSDEFENCE.ORG. **CoWPAtty MAIN.** Disponível em: <http://www.wirelessdefence.org/Contents/coWPAttyMain.htm>. Acesso em: 04 mar. 2013.

ANEXO A – *SCRIPT* COMPLETO DE ATAQUE UTILIZANDO LOOKUP TABLE

```
#!/bin/bash

if cowpatty -d ./tcc/hashed\ wordlist/hashes.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
then
 exit
else
 if cowpatty -d ./tcc/hashed\ wordlist/hashes2.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
 then
  exit
 else
  if cowpatty -d ./tcc/hashed\ wordlist/hashes3.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
  then
   exit
  else
  if cowpatty -d ./tcc/hashed\ wordlist/hashes4.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
  then
   exit
  else
   if cowpatty -d ./tcc/hashed\ wordlist/hashes5.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
   then
```

```
    exit
   else
    if cowpatty -d ./tcc/hashed\ wordlist/hashes6.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
    then
     exit
    else
     if cowpatty -d ./tcc/hashed\ wordlist/hashes7.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
     then
      exit
     else
      if cowpatty -d ./tcc/hashed\ wordlist/hashes8.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
       then
        exit
       else
        if cowpatty -d ./tcc/hashed\ wordlist/hashes9.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
         then
          exit
         else
         if cowpatty -d ./tcc/hashed\ wordlist/hashes10.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
         then
          exit
         else
```

```
        if cowpatty -d ./tcc/hashed\ wordlist/hashes11.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
        then
         exit
        else
         if cowpatty -d ./tcc/hashed\ wordlist/hashes12.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
         then
          exit
         else
          if cowpatty -d ./tcc/hashed\ wordlist/hashes13.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
          then
           exit
          else
           if cowpatty -d ./tcc/hashed\ wordlist/hashes14.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
           then
            exit
           else
            if cowpatty -d ./tcc/hashed\ wordlist/hashes15.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
            then
             exit
            else
             if cowpatty -d ./tcc/hashed\ wordlist/hashes16.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
              then
```

```
                exit
            else
             if cowpatty -d ./tcc/hashed\ wordlist/hashes17.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
              then
             exit
            else
             if cowpatty -d ./tcc/hashed\ wordlist/hashes18.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
                then
                 exit
            else
             if cowpatty -d ./tcc/hashed\ wordlist/hashes19.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
                 then
                  exit
           else
            if cowpatty -d ./tcc/hashed\ wordlist/hashes20.txt -r
./tcc/passwords/92284053/handshake/withoutAircrack-ng-01.cap -s LabTCC ==
true
                  then
                   exit
            else
               echo "SENHA NÃƒO ENCONTRADA"
             fi
            fi
           fi
          fi
         fi
        fi
       fi
```

fi
fi
fi
fi
fi
fi
fi
fi
fi
fi
fi
fi

ANEXO B - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO LOOKUP TABLE

											Média	
Senha	**Tempo em segundos (aproximado)**										**Tempo (segundos)**	**Velocidade (senhas/segundos)**
00806386	6	6	6	6	6	6	6	6	6	6	6,00	134.397,67
02187695	16	16	16	16	16	16	17	16	18	16	16,30	134.214,42
15348169	113	116	114	115	115	114	115	114	114	115	114,50	134.045,14
22671527	170	172	169	168	168	173	168	170	167	168	169,30	133.913,33
38775814	288	287	290	285	287	287	288	286	287	285	287,00	135.107,37
38918275	288	293	288	293	291	286	292	287	287	289	289,40	134.479,18
55429751	410	412	413	410	407	414	409	411	414	409	410,90	134.898,40
70515193	521	522	521	519	521	521	519	520	526	519	520,90	135.371,84
77550742	578	578	573	581	574	573	576	577	573	573	575,60	134.730,27
92284053	682	686	686	683	683	682	683	683	684	684	683,60	134.997,15

ANEXO C - TABELA COMPLETA DOS DADOS DOS ATAQUES UTILIZANDO DICIONÁRIO

											Médias	
Senhas	**Tempo em segundos (aproximado)**										**Tempo**	**Velocidade (senhas/segundos)**
00806386	3.688	3.679	3.800	3.683	3.682	3.688	3.679	3.800	3.683	3.682	01:01:46 h	217,57
02187695	10.065	10.019	10.079	10.031	10.016	10.065	10.019	10.079	10.031	10.016	02:47:22 h	217,85
15348169*											19:34:58 h	217,71
22671527*											1 dia 04:55:36 h	217,71
38775814*											2 dias 01:28:27 h	217,71
38918275*											2 dias 01:39:22 h	217,71
55429751*											2 dias 22:43:23 h	217,71
70515193*											3 dias 17:58:15 h	217,71
77550742*											4 dias 02:56:51 h	217,71
92284053*											4 dias 21:44:45 h	217,71

* Senhas das quais os valores de tempo foram calculados com base na média dos testes realizados.

Criptoanálise em Redes Sem Fio Utilizando *Lookup Tables*

Deivison Cardoso, Henrique P. Martins, Élvio G. da Silva, Patrick P. Silva

Centro de Ciências Exatas e Sociais Aplicadas – Universidade Sagrado Coração (USC)
Bauru – SP – Brasil

deivisondc@gmail.com, henmartins@gmail.com, egsilva@usc.br,
patrickpsilva@gmail.com

Abstract. *With the advancement of technologies and the exponential growth of computer networks, it has become common to use radio frequencies as a means of transmitting data with the known wireless networks that, different from traditional networks, they need greater security, where the encryption has a important role in the safety of the parties involved. Although many encryption options presented by the market, this paper aims to break some of these barriers using Lookup Tables, revealing the password used to access the wireless network showing the weaknesses of this type of network, and at the end present methods to reduce the vulnerability of users in wireless environments.*

Resumo. *Com o avanço das tecnologias e crescimento exponencial das redes de computadores, tornou-se comum a utilização de radiofreqüências como meio de transmissão de dados com as chamadas redes sem fio que diferente das redes tradicionais, estas precisam de maior segurança, onde a criptografia possui um papel importante na segurança das partes envolvidas. Apesar de tantas opções de criptografia apresentadas pelo mercado, este trabalho visa quebrar algumas destas barreiras utilizando Lookup Tables, revelando a senha utilizada para acessar a rede sem fio mostrando as fragilidades deste tipo de rede, e ao fim apresentar métodos para diminuir a vulnerabilidade dos usuários em ambientes sem fio.*

1. Introdução

Redes de computadores estão presentes hoje na maioria dos lugares. Seja nas casas, ambientes de trabalho ou na educação, as redes carregam consigo um papel tão importante que o seu mau funcionamento acaba gerando desordem na sociedade. Por definição, redes de computadores é a conexão de dois ou mais computadores com capacidade de trocar dados entre si. As redes mais primitivas eram capazes de apenas trocar informações de texto, de forma limitada, enquanto hoje em dia os dados trafegados na rede podem ser fluxos de voz, vídeos, textos e gráficos entre diversos sistemas computacionais – tais como celulares, *notebooks*, *tablets*, entre outros dispositivos – e não apenas entre computadores como antigamente.

Segundo informações publicadas no de 2013 pelo Centro de Estudo, Resposta e Tratamento de Incidentes de Segurança no Brasil – CERT.br (2013)– no ano de 2012 foram reportados ao centro 7815 invasões bem sucedidas em computadores e redes. Estes números tendem a subir enquanto os usuários não estejam cientes das vulnerabilidades que podem ser encontradas em suas redes.

Através destes números fica claro que a segurança da informação é um requisito

básico para usuários em geral. A forma de segurança mais comum encontrada hoje são as senhas, conjunto de caracteres que tem por finalidade garantir acesso apenas ao proprietário da mesma. Porém uma falha que se encontra em grande parte dos usuários é a forma como são criadas estas senhas, pois geralmente escolhem-se senhas simples para que possam lembrar futuramente, e esta simplicidade abre portas para pessoas sem autorização.

Dentro da área de segurança da informação, é trivial encontrarmos algum tópico relacionado à criptografia. De forma geral esta técnica se resume em escrever secretamente, ou melhor, comunicar-se de maneira que nenhuma outra pessoa além do remetente e destinatário seja capaz de decifrar a mensagem. Com base neste conceito foram criados diversos algoritmos, fórmulas matemáticas para manipular a mensagem e deixá-la ilegível. Sendo assim, mesmo que esta mensagem seja interceptada por alguém, este não conseguirá ler a mensagem original, a menos que este possua a informação que permite a encriptação e desencriptação da mensagem, também conhecida como chave.

Aprofundando ainda mais dentro da criptografia, podemos encontrar as chamadas funções *hash*, que resultam em um resultado único de tamanho fixo, independente do tamanho da informação original (considerado único porque apesar de teoricamente existir a possibilidade de dois *hashes* com resultados iguais a probabilidade de isto ocorrer é quase nula). Um detalhe importante deste resultado gerado, conhecido apenas como *hash*, é sua incapacidade de ser reprocessado inversamente gerando a informação original.

Esta característica aplicada às senhas implica em um crescimento de segurança a ponto de que uma senha interceptada, estará ilegível e de forma irreversível para o atacante, sendo em primeira instância inútil. Mas como não existe nada completamente seguro, *hackers*[1] surgem com novas técnicas a cada vulnerabilidade encontrada, algumas vezes para o bem, reportando o erro às empresas detentoras da programação, e às vezes para o benefício próprio, obtendo acesso indevido às redes, através de ataques às senhas criptografadas, ou simplesmente *hashes*.

Dentre vários tipos de ataques a senhas, um será destacado por ser o foco deste trabalho, o qual é denominado como ataque *off-line*, por ter a capacidade de ser executado na máquina local do atacante, sem qualquer tipo de conexão com o alvo. Diante desta situação este trabalho visa demonstrar as técnicas utilizadas para a quebra de senhas de redes sem fio, e formas de como se proteger destes ataques *off-line*.

2. Redes Sem Fio

Por motives de segurança foram criados protocolos para o padrão 802.11 da IEEE, sendo eles o WEP (*Wired Equivalent Privacy*), WPA (*Wi-fi Protected Access*) e WPA2. De acordo com Rufino (2011) estes protocolos deveriam, entre outras características, ser suficientemente forte, serem capaz de serem implementados em equipamentos com baixo poder de processamento, e por fim, terem autossincronísmo, ou seja, permitir o funcionamento de um equipamento com a mínima ou nenhuma intervenção manual assim que este entre na área de cobertura.

O WPA2, como sendo foco deste trabalho, traz consigo a implementação do

[1] "[...] especialistas em computadores e tecnologia que estudam a fundos os sistemas operacionais, redes e protocolos de comunicação, seja para conhecê-los e melhorá-los" (ERICKSON, 2009)

protocolo CCMA (*Counter Cipher Mode with Block Chaining Message Authentication Code Protocol*), utilizando o algoritmo de criptografia AES. Com base nas afirmações de Haines (2010), tanto o WPA quanto o WPA2 quando estão no modo PSK, utiliza-se de chaves de 256 *bits* em hexadecimal, gerados pelo algoritmo PBKDF2 – *Password-Based Key Derivation Functions*. Isto foi proposto devido a problemas de padronização na entrada da chave, pois não era especificado se deviam estar no formato hexadecimal ou no mais comum ASCII.

3. Tipos de Ataques

Os tipos de ataques utilizados durante a realização deste trabalho, são denominados como ataques off-line. Para este tipo de ataque, é necessário que o atacante possua um arquivo com os *hashes* (senhas criptografadas de forma irreversível). Existem duas modalidades para estes ataques, sendo a primeira quando o atacante possui uma lista de senha, e gera o *hash* de cada uma até encontrar o *hash* que está no arquivo. A segunda modalidade é quando o atacante possui uma lista de *hashes* para então descobrir qual a senha que gera o *hash* presente no arquivo.

Na primeira modalidade podem ser aplicados as técnicas de força bruta e dicionário. Erickson (2009) conceitua que é um ataque onde é testado cada combinação possível. Já na segunda modalidade a técnica aplicada é a *Lookup Tables* ou tabelas de pesquisa em tradução literal, que são tabelas pré-computadas limitadas pelo tamanho, e os caracteres presentes na tabela.

As *Lookup Tables* usadas contra as senhas de redes sem fio possuem um tratamento a mais: o valor do *salt*, uma espécie de "tempero" na senha, resultando em um *hash* diferente da senha pura. O *salt*, no caso de redes sem fio nada mais é que o ESSID, o nome da rede sem fio. Para as redes WPA/PSK e WPA2/PSK é computado o algoritmo PBKDF2 para ser gerado a *Lookup Table*.

Erickson (2009) afirma que com esta estrutura de dados "qualquer senha poderia ser *crackeada* no tempo que se leva para pesquisar". Porém, para que seja possível esta rapidez na pesquisa, é necessário aceitar uma troca por espaço, sendo que o espaço ocupado é consideravelmente grande.

4. Criptografia

De maneira sucinta, pode-se dizer que a criptografia é aplicada a uma mensagem em texto plano, ou seja, aquele que estamos acostumados e conseguimos ler pelo transmissor e enviado pela rede. O receptor por sua vez, com o uso de uma chave compartilhada entre ele e o transmissor, pode descriptografar a mensagem e recuperar o texto à sua forma simples e legível (PETERSON; DAVIE, 2004). Esta técnica além de fornecer privacidade entre as partes envolvidas, também provê alguns serviços importantes como autenticação e integridade.

O resultado de uma criptografia, chamado de texto cifrado, é concebido através de algoritmos matemáticos que podem ou não ser reversíveis, onde recebe o texto cifrado e transforma em texto plano. Stallings (2008) salienta que a criptografia é a "ferramenta automatizada mais importante para a segurança da rede e das comunicações".

5. SHA-1

O SHA-1 (*Secure Hash Algorithm*) é um padrão federal norte-americano de criptografia, e seu uso é exigido quando aplicações do âmbito federal precisam de um resumo de mensagem seguro Kurose e Ross (2006).

Primeiramente é feita uma expansão de *bits* até que o tamanho seja múltiplo de 512 *bits*. Logo após, de acordo com Tanenbaum (2003), "um número de 64 *bits* contendo o tamanho da mensagem antes do preenchimento é submetido a uma operação OR nos 64 *bits* de baixa ordem".

Tanenbaum (2003) continua explicando que são criados cinco variáveis de 32 *bits*, de H_0 a H_4, as quais são inicializadas como constantes específicadas no padrão. Estas variáveis é onde o *hash* se acumula. Agora, após estes processos, os blocos de 512 *bits* são processados, sendo que para cada bloco as 16 palavras (words – unidade que contém 32 *bits*) são copiadas dentro de um vetor de 80 posições, restando 64 os quais são preenchidos utilizando uma determinada fórmula que faz rotação circular à esquerda.

Para completar as operações executadas em um único bloco, são criadas mais cinco variáveis, chamadas de A, B, C, D e E, contendo os valores de H_0 a H_4 respectivamente. Oitenta iterações de cálculos são realizados utilizando funções diferentes que mudam a cada 20 iteração. Após o término dos cálculos, os valores de A até E são somados aos valores de H_0 a H_4.

Com o término das operações neste bloco, o próximo é preparado para fazer o mesmo processo, com os vetores de palavras reinicializado, e com o vetor H com os valores obtidos do processo anterior, seguindo esta sequência até que todos os blocos tenham sido processados. Ao término do último bloco, "as cinco palavras de 32 *bits* no [vetor] H são transmitidas como saída, formando o *hash* criptográfico de 160 *bits*" (TANENBAUM, 2003).

A Figura 1 apresenta um esquema mostrando os blocos de 512 *bits* da mensagem, as variáveis H_0 a H_4 e também o vetor de 80 posições W.

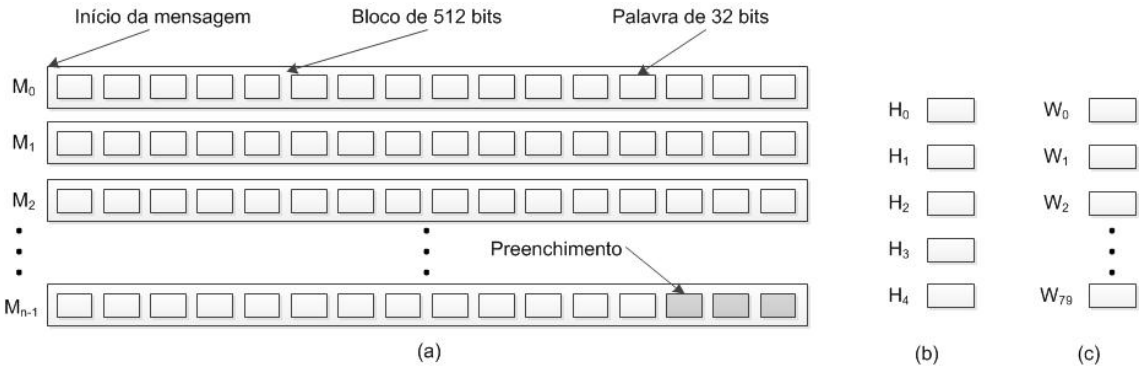

Figura 1. (a) Uma mensagem preenchida até um múltiplo de 512 *bits*. (b) As variáveis de saída. (c) O array de palavras.

6. MAC

O MAC é uma função calculada sobre a mensagem e uma chave secreta, a qual as duas partes da comunicação têm conhecimento (chave simétrica). O resultado desta função é anexado à mensagem, e enviada ao destinatário, onde será realizado o mesmo procedimento. Após essa etapa, o destinatário irá comparar o seu MAC, com o MAC anexado à mensagem que foi enviada pelo remetente. Se ambos os MACs forem iguais, Stallings (2008) deduz que o destinatário tem certeza que a mensagem está íntegra, e que a mensagem veio do remetente declarado, supondo que apenas ambos sabiam a chave secreta

7. HMAC

A IETF (2013a) cita que os objetivos principais por trás do HMAC são:

- Usar, sem modificações, as funções de *hash* disponíveis, particularmente aquelas que tenham bom desempenho em *software*, e para as quais o código esteja ampla e gratuitamente disponível.
- Permitir a substituição fácil da função de *hash* embutida caso sejam encontradas ou exigidas função *hash* mais rápidas ou mais seguras.
- Preservar o desempenho original da função de *hash* sem incorrer em degradação significativa.
- Ter uma análise criptografia bem compreendida quanto à força do mecanismo de autenticação com base em suposições razoáveis sobre a função de *hash* embutida.

De forma simples o HMAC – Keyed-Hashing for Message Authentication – é uma implementação do MAC baseado em funções *hash* sem chave, como por exemplo, o MD5 ou o SHA-1.

De acordo com Forouzan, (2008), é aplicado primeiramente uma função *hash* sem chave à mensagem concatenada com a chave simétrica. O resultado será um HMAC intermediário, o qual é novamente processado da mesma forma, concatenando agora o próprio HMAC intermediário, e a mesma chave utilizada no processo anterior resultando então no HMAC. Ao receber o HMAC, o receptor irá calcular seu próprio HMAC e comparar com os HMACs recebidos para validar a integridade da mensagem e autenticar a origem dos dados.

8. PBKDF2

Benton (2010) afirma que o PBKDF2 é utilizado devido à falta de proteção por parte dos usuários na escolha da senha, ou PSK, não importando a proteção usada, CCMP ou TKIP. Esta afirmação também é apoiada pelo NIST (2010), o qual também especifica que o PBKDF2 utiliza-se de HMAC junto com SHA-1, além do uso do *salt*, e o contador de iteração que a função deve conter, "devendo ser selecionada o maior possível, desde que o tempo requerido para gerar a chave usando a senha de entrada seja aceitável pelos usuários", sendo que o mínimo recomendado pelo órgão é de 1000 iterações enquanto que para sistemas mais poderosos, e que necessitam de muita proteção este número chega a dez milhões.

Os processos executados pelo PBKDF2 descritos pela IETF (2013b) são a

adição do *salt* à senha que o usuário digitou, onde este resultado é o dado de entrada para um algoritmo geralmente de *hash*, podendo ser também de uma função pseudoaleatória, durante uma quantidade de iterações estipulada na entrada dos dados, assim como o número de *bits* desejável na saída.

Já Haines (2010), focando nas redes sem fio, afirma que na senha digitada pelo usuário além do *salt*, que seria o SSID, entra também na adição o tamanho do SSID. O algoritmo SHA-1 é o escolhido para processar a soma dos três elementos anteriores, e então utilizar seus dados de saída como dados de entrada para a próxima iteração, durante todas as suas 4096 iterações, retornando ao final um *hash* de 256 *bits*.

9. Metodologia

Foi estabelecido para este trabalho dez senhas diferentes geradas por um *software* desenvolvido na linguagem Java. Para cada senha gerada, foram realizados dez testes para que fosse calculada uma média do tempo tomado para a quebra da senha. Com as senhas geradas, foi utilizado o *software Crunch* que cria um dicionário com todas as combinações possíveis de acordo com as regras que forem passadas como parâmetro. Para esta pesquisa, foi utilizado um tamanho fixo de oito caracteres, sendo estes caracteres apenas numéricos, resultando em cem milhões de combinações. A razão desta restrição é pelo fato de que o dicionário a ser criado deixaria o arquivo muito grande, aumentando ainda mais este tamanho com o processamento que irá receber.

O dicionário gerado foi utilizado como base para a geração das *Lookup Tables* com o *software genPMK*, que exige um parâmetro muito importante, o *salt*. Este parâmetro é muito importante, pois um erro na digitação deste pode inutilizar toda a *Lookup Table*, já que esta é específica para um único SSID.

Com estes arquivos preparados, foi dado início aos ataques utilizando *Lookup Tables*. Para isto, é necessário que a interface de rede sem fio do atacante esteja em modo monitor, o que pode ser providenciado com o *software airmon-ng*. Com a placa em modo monitor é possível monitorar todo o tráfego de dados de rede sem fio que esta interface consegue captar assim como filtrar os dados apenas da rede que se deseja atacar e salvar todo o tráfego. Isto é necessário pois para durante o *handshake* de uma estação com um ponto de acesso, está armazenado a senha da rede criptografada. Para a captura do *handshake* pode ser utilizado o *software aireplay-ng* para forçar a desautenticação de uma estação, ou apenas esperar que uma estação qualquer se conecte a rede.

Com o *handshake* capturado e as *Lookup Tables* geradas, é possível realizar a quebra de senha de modo off-line utilizando o *software coWPAtty*, que exige que seja passado como parâmetro a *Lookup Table*, o arquivo contendo o *handshake*, e o SSID da rede alvo. É possível também executar ataques de dicionário simples com o mesmo *software*, apenas alterando o arquivo da *Lookup Table* para o arquivo de dicionário. Neste trabalho foram realizados os dois tipos de ataques para fazer uma comparação de tempo entre os dois métodos.

10. Resultados Coletados

Para a coleta dos dados, foi analisado cada *screenshot* disparada automaticamente pelos scripts de ataque e calculado a diferença de tempo entre o início e o fim do ataque e tabulado para uma melhor visualização dos dados.

A Tabela 1**Erro! Fonte de referência não encontrada.** apresenta os dados coletados dos ataques utilizando *Lookup Tables*, enquanto a Tabela 2 apresenta dados dos ataques de dicionário, observando que as senhas seguidas de um asterisco significam que foram resultados calculados com base nas médias dos ataques realizados.

Tabela 1. Médias coletadas dos ataques utilizando *Lookup Table*

Médias – *Lookup Table*		
Senhas	**Tempo (segundos)**	**Velocidade (senhas/segundo)**
00806386	6	134.397,67
02187695	17,2	127.191,57
15348169	114,5	134.045,14
22671527	169,3	133.913,33
38775814	287	135.107,37
38918275	289,4	134.479,18
55429751	410,9	134.898,40
70515193	520,9	135.371,84
77550742	575,6	134.730,27
92284053	683,6	134.997,15

Tabela 2. Médias coletadas dos ataques utilizando Dicionário

Médias – Dicionário		
Senhas	**Tempo**	**Velocidade (senhas/segundo)**
00806386	01:01:46 h	217,57
02187695	02:47:22 h	217,85
15348169*	19:34:58 h	217,71
22671527*	1 dia 04:55:36 h	217,71
38775814*	2 dia 01:28:27 h	217,71
38918275*	2 dias 01:39:22 h	217,71
55429751*	2 dias 22:43:23 h	217,71
70515193*	3 dias 17:58:15 h	217,71
77550742*	4 dias 02:56:51 h	217,71
92284053*	4 dias 21:44:45 h	217,71

Com base nos dados acima, fica evidente a diferença discrepante entre os dois métodos. Os ataques utilizando *Lookup Tables* comparados com os ataques utilizando dicionário possuem uma velocidade acima de seiscentas vezes.

* Senhas das quais os valores de tempo foram calculados com base na média dos testes realizados.

11. Considerações Finais

Após os testes realizados e analisado os resultados obtidos dos mesmos foi possível verificar que a quebra de senha utilizando *Lookup Tables* é incrivelmente mais rápido, porém existe o tempo para sua geração, que levou em torno de 13 horas para a geração de cem mil combinações. Sendo assim, se as *Lookup Tables* criadas forem utilizadas para apenas um ataque, então esse tipo de ataque torna-se ineficiente. Entretanto se a mesma *Lookup Table* for utilizada para diversos ataques, então existe uma grande vantagem de utilizá-las em relação aos dicionários.

References

CERT.BR. Incidentes Reportados ao CERT.br: Janeiro a Dezembro de 2012. Disponível em: <http://www.cert.br/stats/incidentes/2012-jan-dec/total.html>. Acesso em: 15 mar. 2013.

RUFINO, Nelson Murilo de O.. Segurança em Redes sem Fio: Aprenda a proteger suas informações em ambientes Wi-fi e Bluetooth. 3. ed. São Paulo: Novatec, 2011.

ERICKSON, Jon. Hacking. São Paulo: Digerati Books, 2009

PETERSON, Larry L.; DAVIE, Bruce S.. Redes de Computadores: Uma Abordagem de Sistemas. Rio de Janeiro: Elsevier, 2004

STALLINGS, William. Criptografia e Segurança de Redes. São Paulo: Pearson Prentice Hall, 2008.

KUROSE, James F.; ROSS, Keith W.. Redes de Computadores e a Internet: Uma abordagem top-down. São Paulo: Pearson Addison Wesley, 2006.

TANENBAUM, Andrew S.. Redes de Computadores. 4. ed. Rio de Janeiro: Elsevier, 2003.

IETF. RFC 2104: HMAC: Keyed-Hashing for Message Authentication. Disponível em: <http://www.ietf.org/rfc/rfc2104.txt>. Acesso em: 04 mar. 2013a.

IETF. RFC 2898: PKCS #5: Password-Based Cryptography Specification Version 2.0. Disponível em: <http://www.ietf.org/rfc/rfc2898.txt>. Acesso em: 26 maio 2013b.

FOROUZAN, Behrouz A.. Comunicação de Dados e Redes de Computadores. 4. ed. São Paulo: Mcgraw-hill, 2008.

KEVIN BENTON. The Evolution of 802.11 Wireless Security. Disponível em: <http://itffroc.org/pubs/benton_wireless.pdf>. Acesso em: 27 fev. 2013.

NIST. Recommendation for Password-Based Key Derivation: Part 1: Storage Applications. Disponível em: <http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf>. Acesso em: 27 fev. 2013.