Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

PEDRO LUIS PINHEIRO SOLA

SEGURANÇA EM REDES WIRELESS

Assis/SP
2018

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

PEDRO LUIS PINHEIRO SOLA

SEGURANÇA EM REDES WIRELESS

Projeto de pesquisa apresentado ao curso de Ciência da Computação do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando: Pedro Luis Pinheiro Sola
Orientador: Prof. Me. Fábio Eder Cardoso

Assis/SP
2018

S684s SOLA, Pedro Luis Pinheiro

Segurança em redes wireless / Pedro Luis Pinheiro Sola. – Assis, 2018.

54p.

Trabalho de conclusão do curso (Ciência da Computação). –Fundação Educacional do Município de Assis-FEMA

Orientador: Ms. Fábio Éder Cardoso

1.Segurança 2.Redes 3.Wireless

CDD 005.8

SEGURANÇA EM REDES WIRELESS

PEDRO LUIS PINHEIRO SOLA

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação, avaliado pela seguinte comissão examinadora:

Orientador: Prof. Me. Fábio Eder Cardoso

Examinador: Prof. Dr. Luiz Carlos Begosso

Assis/SP
2018

DEDICATÓRIA

Dedico este trabalho a todos que me deram suporte ao longo dessa jornada.

AGRADECIMENTOS

Ao professor, Fábio Eder Cardoso pela orientação e pelo constante estimulo transmitido durante o trabalho.

Aos amigos, Carlos Roberto, Addam Caue, Pedro Foganholi, Leonardo Jubran, José Henrique e a todos que colaboraram direta ou indiretamente na execução deste trabalho e aos que cursaram junto durante esses quatro anos.

Aos familiares, minha mãe Marielle, meu padrasto Fabio, meu pai Emerson Luis, minha madrasta Giselle Abrahão e minha namorada Thalia Caroline Martins, pelo suporte e apoio durante toda essa aventura que tem sido a faculdade.

If brute force doesn't work, you aren't using enough.

Sherrilyn Kenyon (1965 – Atualidade)

RESUMO

As redes sem fio estão cada vez mais sendo utilizadas por pessoas e por empresas. Há certa preocupação em relação à confiabilidade de acesso e este trabalho tem como intuito explanar a respeito das funcionalidades de uma rede sem fio e como sua segurança pode ser testada e comprovada por meio de aplicativos desenvolvidos utilizando ferramentas de segurança dando o suporte necessário para que a rede deixe de ser vulnerável e corrigindo todos os possíveis defeitos. Redes sem fio utilizam radiofrequência para sua comunicação, seu padrão é o 802.11 ratificado pelo IEEE (*Institute of Eletrical and Eletronic Engineers).*

Como muitos usuários se conectam a estas redes para compartilharem recursos, é de extrema importância prover segurança nas comunicações que utilizam este padrão uma vez que tudo que se propaga, utilizando o ar como meio físico de comunicação, está passível de captura.

O presente trabalho tem como foco apresentar um estudo sobre a segurança em redes wireless, tendo como principal agente a demonstração de como explorar vulnerabilidades wireless em dispositivos que utilizam dos padrões 802.11bagnac, ou seja, roteadores wireless.

Palavras-chave: Segurança, Wireless, Sem Fio, Radiofrequência, 802.11, IEEE.

ABSTRACT

As wireless networks are more and more been used by people and companies, there is a certain concern regarding access reliability, this paper is intended to explain a relation of the functionalities of a wireless network and how their security can be tested and proven through applications developed with security tools, supporting a security system and all possible overruns. Wireless networks for radiofrequency for its communication, its standard and 802.11 ratified by the IEEE (Institute of Electronic and Electronic Engineers).

Many of the users connect to these networks to share resources, it is of utmost importance to provide security in the communications that use this standard since everything that is propagates, using the physical means of communication, is capturing.

The present work focuses on presenting a study on wireless network security, with the main agent demonstrating how to exploit wireless vulnerabilities in devices that use 802.11bagnac standards, that is, wireless routers.

Keywords: Security, Wireless, radio frequency, 802.11.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AP – *Acess Point*

WEP – Wired Equivalency Protocol

WPA – Wi-fi Protected Acess

WPA2 – *Wi-fi Protected Acess, Version Two*

TKIP – *Temporal Key Integrity Protocol*

AES – *Advanced Encryption Standard*

IEEE – *Institute of Eletrical and Eletronic Engineers*

Mbps – *Megabytes per second*

Kbps – Kilobytes per second

DSSS – *Direct Sequence Spread Spectrum*

FHSS – *Frequency-hopping spread spectrum*

OFDM – *Orthogonal Frequency Division Multiplexing*

Mhz – *Mega Hertz*

Ghz – *Giga Hertz*

MIMO – *Multiple Input Multiple Output*

STAS – *Stations*

MIMO-OFDM – *Multiple Input Multiple Output - Orthogonal Frequency Division Multiplexing*

TxBF – *Beamforming*

CCK – *Complementary code keying*

EAP – *Extensible Authentication Protocol*

Radius – *Remote Authentication Dial-In User Server*

ICP – Infraestrutura de Chaves Públicas

PSK – *Pré-Shared Key*

IP – *Internet Protocol*

URL – *Uniform Resource Locator*

HTTP – *HyperText Transfer Protocol*

SUMÁRIO

1. INTRODUÇÃO

As redes sem fio têm se tornado cada vez mais utilizadas pelo público em geral e isso acontece por diversas vantagens que esse tipo de rede provê, como por exemplo, mobilidade, praticidade, flexibilidade, redução de custos dentre outros (ALVES, SOUZA, 2016). Porém, os dados que antes trafegaram utilizando cabos de rede, agora utilizam-se das ondas de radiofrequência que circulam pelo ar, sendo assim ela se torna bem mais vulnerável a ataques do que a rede cabeada.

Desta forma, os estudos de criptografia, segurança de dados e protocolos, devem ser aprimorados cada vez mais para que a rede esteja mais segura para ser utilizada na comunicação dos dados.

A maior preocupação de uma rede sem fio é, sua vulnerabilidade, com esse foco, será feito testes utilizando ferramentas do Kali Linux para complementar a segurança ao *Access Point* (AP), essas ferramentas farão a varredura de dados e leitura de registros.

1.1. OBJETIVOS

O presente trabalho tem como o objetivo utilizar um AP para a realização de testes de vulnerabilidades, aferindo se as mesmas redes estão seguras ou não, demonstrando suas falhas e os métodos para minimizar os riscos.

Os testes foram realizados com as ferramentas contidas no Kali Linux, para que qualquer pessoa possa realizar esses testes em seu próprio AP e ter uma maneira de checar as vulnerabilidades do mesmo.

1.2. PENTEST (*PENETRATION TEST*)

Pentest são bateria de testes metodológicos, que tem como objetivo descobrir, mapear e expor todas as possíveis vulnerabilidades de uma rede.

Existem diversos tipos de testes que podem ser realizados através de Pentests, como, testes em redes cabeadas, redes sem fio (Wireless), web, revisão de códigos fontes, desenvolvimento de programas que exploram vulnerabilidades em outros softwares (exploits) entre outras. (Moreno,2016)

O objetivo de Pentests, não é ter acesso a uma rede sem permissão, mas sim encontrar falhas e aplicar as devidas medidas possíveis para tornar essa rede menos ou se possível invulnerável.

Infelizmente, os mesmos testes que uma pessoa poderia efetuar em uma auditoria, podem ser realizados por criminosos virtuais, logo essas pessoas podem ter acesso a dados confidenciais.

Por conta do crescente número de ataques digitais, a legislação brasileira considera o ato de invadir computadores como crime passível de punição. Lei de nr 12.737, pena de 3 meses a 1 ano com multa. (Moreno,2016)

1.2.1. Distribuições de Pentest

Há diversas distribuições de sistemas operacionais para fazer auditoria em redes, sites, códigos-fontes, como, BackTrack, Kali Linux, BackBox, Samurai WTF, Wifislax entre outras.

1.2.1.1. BackTrack[1]

Baseado na distribuição Ubuntu, foi muito utilizado no passado para fazer auditorias, porém atualmente já está desatualizado, seu substituto é o Kali Linux, mesmo que seja uma distribuição que já parou de ser atualizada, vale a pena ser citada. (Moreno,2016)

[1] https://www.backtrack-linux.org

1.2.1.2. Samurai WTF[2] (*Web Testing Framework*)

O Samurai *Web Testing Framework é* uma distribuição de maquinas virtuais, suportadas pelo VirtualBox e pelo VMWare, ela é completamente configurada para finalidades de focar completamente em auditoria de web sites, apresentando diversas ferramentas para esse propósito. (Referenciar)

1.2.1.3. BackBox[3]

BackBox Linux é um teste de penetração e avaliação de segurança orientada para distribuição Linux, fornecendo um kit de ferramentas de análise de rede e sistemas. Ele inclui algumas das ferramentas de segurança e análise mais comumente conhecidas, visando uma ampla gama de metas, que vão desde análise de aplicações web a análise de rede, testes de estresse, sniffing, avaliação de vulnerabilidades, análise forense computacional, automotivo e exploração. Ele foi construído no sistema central do Ubuntu e totalmente customizado, projetado para ser um dos melhores testes de penetração e distribuição de segurança e muito mais. (Referenciar)

1.2.1.4. Wifislax[4]

Wifislax é uma distribuição GNU / Linux para realizar auditorias rápidas e fáceis de redes sem fio, para que você possa acessar as inúmeras ferramentas do sistema preparadas para ajudá-lo.

Atualmente, é uma das ferramentas mais usadas para auditar redes WiFi, tanto para a grande variedade de ferramentas incluídas para fazê-lo quanto para os inúmeros drivers para placas de rede que vêm instaladas na distribuição.

---

[2] http://www.samurai-wtf.org
[3] https://backbox.org/linux
[4] https://wifislax.en.uptodown.com/ubuntu

Como sempre, é importante ter em mente que esse tipo de distribuição deve ser usado para melhorar a segurança de sua rede, não para ações ilegais como roubar senhas de Wi-Fi ou interceptar dados, que são consideradas crimes graves pelas autoridades.

Para atender os objetivos propostos o presente trabalho foi dividido em seis capítulos.

Capitulo dois é sobre os Padrões de Redes Wireless para um entendimento maior sobre as redes wireless e seus protocolos.

Capítulo três é sobre os Modelos de Criptografia que acompanham os Padrões de Redes Wireless.

Capitulo quatro é sobre o Kali Linux que iremos utilizar como o sistema operacional no nosso trabalho e o capitulo explica sobre ele e as ferramentas que foram utilizadas no trabalho através dele.

Capitulo cinco será sobre a Metodologia do presente trabalho, explicando como foram efetuadas as utilizações das ferramentas.

O capítulo 6 é sobre a conclusão do presente trabalho.

2. PADRÕES DE REDES WIRELESS

As redes wireless possuem diversos padrões, com forma que, cada um atenda um tipo de necessidade. Os padrões que iremos pôr em pauta no presente trabalho serão: 802.11, 802.11b, 802.11a, 802.11g, 802.11n, 802.11ac.

A tabela 1 apresenta as principais diferenças entre os padrões que estão presentes no trabalho.

Protocolo	Ano de Lançamento	Frequência de Transmissão - Alcance Antena	Taxa de Transmissão - Velocidade	Modulação
802.11	1997	2.4 Ghz	1 a 2 Mb/s	DSSS,FHSS
802.11b	1999	2.4 Ghz	1, 2, 5.5, 11 Mb/s	DSSS,CCK
802.11ª	1999	3.7 ou 5.0Ghz	1 6,9,12,18,24,36,48,54 Mb/s	OFDM
802.11g	2003	2.4 Ghz	Mesmo padrão dos 802.11 e 802.11b	OFDM,DSSS
802.11n	2007	2.4 ou 5.0 Ghz	150 Mb/s (Por Antena)	M I M O - OFDM
802.11ac	2013	5.0 Ghz	433 Mb/s (Por Antena)	MU-MIMO
802.11ad	2014	60 Ghz	Opera em torno dos 6.75 Gb/s	OFDM

Tabela 1: Comparação entre os protocolos 802.11

Fonte: (MORENO, 2016)

2.1. PADRÃO 802.11

O padrão 802.11 foi o primeiro padrão wireless, lançado em 1997, com uma taxa de transferência que opera entre 1Mbps a 2Mbps, uma taxa de transferência realmente baixa para as demandas atuais. Neste padrão as transmissões de wireless são realizadas por radiofrequência. (MORENO, 2016)

Todos padrões de redes wireless possuem técnicas de modulação, elas são utilizadas para dividir a frequência de operação em até 14 canais. O padrão 802.11 utiliza as técnicas DSSS e FHSS para fazer a modularização. O DSSS é utilizado para dividir a operação em até 14 canais e o FHSS é utilizado para transmitir informações em várias frequências, de forma que, de tempos em tempos ele mudará a frequência transmitida, para evitar interferência. (MORENO, 2016)

2.2. PADRÃO 802.11B

O Padrão 802.11b foi lançado em 1999, como uma atualização do padrão 802.11. A principal característica que diferencia estas duas versões é a possibilidade de estabelecer conexões nas seguintes velocidades de transmissão: 1Mb/s, 2Mb/s, 5.5 Mb/s e 11 Mb/s. A frequência utilizada pelo padrão 801.11, é mantido no 802.11b que é de 2,412GHz a 2,484 Ghz, a tabela 2 mostra todas as frequências que esses padrões funcionam. (MORENO, 2016)

Canal	Frequência
1	2.412 Ghz
2	2.417 Ghz
3	2.422 Ghz
4	2.427 Ghz
5	2.432 Ghz
6	2.437 Ghz
7	2.442 Ghz
8	2.447 Ghz
9	2.452 Ghz
10	2.457 Ghz
11	2.462 Ghz
12	2.467 Ghz
13	2.472 Ghz
14	2.484 Ghz

Tabela 2: Canais e respectivas frequências do padrão 802.11b

Fonte: (MORENO, 2016)

A área de cobertura de uma transmissão do padrão 802.11b pode variar por vários fatores, como, a taxa de dados necessária, a capacidade, as fontes de interferências de radiofrequência e o meio físico contando também as características de potência, conectividade e modelo de antena. Teoricamente, a área é vista como um círculo com diâmetro de aproximadamente 60 metros a 11Mbps em áreas fechadas, em áreas abertas pode alcançar até 400 metros de diâmetro a 1Mbps. Entretanto, na pratica consegue-se verificar que em ambientes fechados a conectividade chega a 50 metros de distância do AP mais próximo. Já em áreas abertas, com a utilização de antenas de alta potência

pode-se alcançar uma conectividade de aproximadamente um quilometro de distância. (AMARAL, MAESTRELLI, 2005)

Um fator interessante do padrão 802.11b e seus sucessores é que, para manter a transmissão funcionando o maior tempo possível ela diminui sua taxa de transmissão de dados até o ponto de seu limite mínimo (1 MB/s) à medida em que a estação fica mais longe do ponto de acesso, o caminho contrário também ocorre, quanto mais perto do ponto de acesso, maior será a velocidade de transmissão de dados. (Amaral, Maestrelli, 2005)

O padrão 802.11b foi o primeiro a ser adotado em larga escala, sendo, portanto, um dos responsáveis pela popularização das redes Wi-Fi. (ALECRIM, 2013)

2.3. PADRÃO 802.11A

O padrão 802.11a foi disponibilizado no final do ano de 1999, quase na mesma época que o padrão 802.11b, porém ele veio após o 802.11b. Sua principal característica é a possibilidade de operar em taxas de transmissão de dados maiores que seu antecessor nos seguintes valores: 6Mb/s, 9Mb/s 12Mb/s, 18Mb/s, 24Mb/s, 36 Mb/s, 48 Mb/s e 54Mb/s. (ALECRIM, 2013)

O alcance de sua transmissão é de cerca de 50 metros, porém a sua frequência de operação é diferente do padrão 802.11, ela opera em 5 GHz, portanto, são redes que utilizam canais elevados, 120, 140 e todos AP que utilizarem essa faixa de canal, estarão utilizando o padrão 802.11a. O uso desta frequência é de grande conveniência por apresentar menos possibilidades de interferência, afinal, este valor é pouco usado, porém pode trazer alguns problemas, já que muitos países não possuem regulamento para esta frequência. (ALECRIM, 2013) e (MORENO, 2016)

Além disso, esta característica pode trazer algumas dificuldades em relação a comunicação com outros dispositivos que operam nos padrões 802.11 (2.4GHz) e 802.11b (2.4 GHz).

Outro detalhe importante é que em vez de utiliza-se DSSS (*Direct Sequence Spread Spectrum*) ou FHSS (*Frequency-hopping spread spectrum*), o padrão 802.11ª utiliza-se de uma técnica conhecida como OFDM (*Orthogonal Frequency Division Multiplexing*), nela a

informação a ser trafegada é dividida em vários pequenos conjuntos de dados que são transmitidos simultaneamente em diferentes frequências, de forma que uma não interfira uma na transmissão da outra, fazendo com que esta técnica funcione de uma maneira bem eficiente. (MORENO, 2016)

Apesar de oferecer taxas de transmissão maiores, o padrão 802.11a não chegou a ser tão popular quanto o padrão 802.11b.

2.4. 802.11G

O padrão 802.11g foi lançado em 2003, e é tido como o sucessor da versão 802.11b, pois ele é totalmente compatível com a sua versão antiga. Com isso em mente um dispositivo que opera em 802.11b pode fazer transmissões de arquivos para quaisquer dispositivos que operem em 802.11g, sem qualquer tipo de problema, além do fato de eles possuírem taxas de transmissão diferentes, sendo assim a taxa apenas seria limitada pelo protocolo 802.11b. (ALECRIM, 2013)

O principal atributo do padrão 802.11g é o fato de que ele pode trabalhar com taxas de transmissão de até 54 MB/s, assim como acontece com o padrão 802.11a. Entretanto, o 802.11g opera com frequências na faixa de 2,4GHz (canais de 20 MHz) e possui praticamente o mesmo poder de cobertura do seu antecessor, o padrão 802.11b.

A transmissão deste padrão utiliza-se da mesma técnica que seus antecessores a OFDM, entretanto, quando a comunicação a ser efetuada será com um dispositivo 802.11b, a técnica de transmissão passa a ser o DSSS, com isso ele é capaz de comunicar-se tanto com AP que utilizam os padrões 802.11a e 802.11b. (MORENO, 2016)

2.5. 802.11N

O início do desenvolvimento da especificação 802.11n iniciou-se em 2004 e foi finalizado em setembro de 2007. Durante este período de desenvolvimento, foram lançados vários dispositivos compatíveis com a versão não terminada do padrão, estamos falando do sucessor do protocolo 802.11g tal como este foi sucessor do protocolo 802.11b.

O padrão 802.11n tem como principal característica o uso de uma técnica chamada MIMO (*Mutiple-Input Multiple Output*), capaz de aumentar de maneira considerável as taxas de transferência de dados por meio da combinação de várias vias de transmissão. Com isso em mente é possível, por exemplo, usar dois, três ou quatro emissores e receptores para o funcionamento da rede. Uma das configurações mais comuns neste caso é o uso de APs (*Access Points*) que utilizam três antenas e STAs (*Stations*) com a mesma quantidade de receptores. Somando esta característica de combinação com o aprimoramento de suas especificações, o padrão 802.11n é capaz de fazer transmissões de dados na faixa de 300 MB/s e, teoricamente, pode atingir taxas de até 600 MB/s. No modo de transmissão mais simples, com uma via de transmissão, o protocolo 802.11n pode chegar à casa dos 150 MB/s. (MORENO, 2016)

Em relação à frequência deste padrão, ela pode trabalhar com faixas de 2,4 GHz e 5GHz, o que faz com que ele se torne compatível com os padrões anterior, inclusive com o 802.11a. Cada canal dentro dessas faixas possui por padrão uma largura de 40 MHz.

Suas técnicas de transmissão mantem as das anteriores, o OFDM, porém com algumas determinadas alterações, devido ao uso do esquema MIMO, sendo assim, muitas vezes chamado de MIMO-OFDM. Alguns estudos apontam que sua área de cobertura pode ultrapassar os 400 metros. (ALECRIM, 2013)

2.6. 802.11AC

O padrão 802.11AC é o sucessor do padrão 802.11n, cujas especificações foram desenvolvidas quase que totalmente entre os anos de 2011 e 2013.

O principal foco do desenvolvimento do protocolo 802.11ac é a sua velocidade, que é estimado em até 433 MB/s no modo mais simples de operação. Porém, teoricamente, é

possível fazer a rede superar os 6GB/s em um modo mais avançado, que utiliza-se de múltiplas vias de transmissões, no máximo até oito. O foco é que a indústria priorize equipamentos com uso de até três antenas, fazendo a velocidade máxima ser de aproximadamente até 1,3 GB/s. (MORENO, 2016)

O protocolo 802.11ac trabalha na frequência de 5 GHz, sendo que, dentro desta faixa, cada canal pode ter, por padrão, largura de 80 MHz (160 MHz como opcional).

O 802.11ac também possui técnicas mais avançadas de modulação, mais precisamente, trabalha com o esquema MU-MUMO *(Multi-User MIMO*), de forma que este permite a transmissão e receptação de sinal de vários terminais, como se estes estivessem trabalhando de maneira colaborativa, todos na mesma frequência.

Se destaca também neste padrão, o uso de um método de transmissão chamado TxBF (*Beamforming)*, que já existia no padrão 802.11n, porém era opcional, ele trata-se de uma tecnologia que permite ao aparelho transmissor, avaliar a comunicação com um dispositivo cliente para otimizar a transmissão em sua direção.

3. MODELOS DE CRIPTOGRAFIA

Atualmente encontra-se três tipos de Criptografia em redes wireless, WEP (*Wired Equivalent Privacy*), WPA (*Wi-fi Protected Acess*), WPA2 (*Wi-fi Protected Acess, Version Two*).

3.1. WEP

Com o foco em proteger os dados da rede, o modelo de criptografia WEP acrescenta criptografia aos pacotes que circulam na rede wireless. Desta forma, caso um pacote seja capturado, eles estarão criptografados, tornando sua leitura ilegível. Como em redes wireless o foco não é utilizar-se os cabos a criptografia faria o papel da segurança, somente os dispositivos que tiverem acesso a chave WEP criptográfica podem conectar-se à rede e utilizar-se dela. (MORENO, 2016)

Como o WEP atua na camada de enlace entre as estações e o ponto de acesso. Sua confidencialidade impede que pessoas não autorizadas tenham acesso à informação, e a implementação desta é opcional. (ROCHA, 2006)

Quando está ativada, cada estação tem uma chave secreta compartilhada com o ponto de acesso, e não há uma forma padrão de distribuição dessas senhas, sendo feita manualmente em cada estação. Sua integridade garante que o receptor obtenha os dados corretos, ou seja, que não haja alterações nos frames enviados pelo transmissor, nem dados indesejados incluídos na transmissão ou removidos no meio do caminho e sua autenticidade identifica quem está executando uma determinada ação, podendo assim fazer um controle de acesso aos recursos disponíveis. (ROCHA, 2006)

Apesar de o WEP ser bastante utilizado para tornar a comunicação de uma rede sem fio mais segura, muitas falhas são apontadas. Uma das vulnerabilidades desse protocolo está associada à reutilização do vetor de inicialização (IV).

Como seu algoritmo de garantia de integridade é linear, possibilita que modificações sejam feitas no pacote sem que sejam detectadas. Uma das grandes fraquezas do WEP é

a falta de gerenciamento de chaves, pois o padrão WEP não especifica como deve ser a distribuição das chaves. (ANDRADE, et al., 2008)

3.2. WPA

Por conta do sistema de criptografia WEP ser extremamente frágil, em 2002, o grupo da Wify Alliance lançou o WPA. (MORENO, 2016)

O WPA atua em duas áreas. A primeira é a qual substitui o WEP, criptografando os dados e garantindo a privacidade do tráfego, e a segunda, autentica o usuário, utilizando para isso padrões 802.1x e EAP (*Extensible Authentication Protocol*). (GIMENES,2005)

A autenticação do WPA pode ser realizada por dois protocolos, EAP utilizando um servidor Radius (*Remote Authentication Dial-In User Server*) para autenticação, podendo ainda necessitar de uma infraestrutura de chaves públicas (ICP), caso se utilize certificados digitais para autenticar usuários.Ou PSK (*Pré-shared Key*), utilizada em pequenas redes domesticas ou escritórios utilizando uma chave previamente compartilhada, sendo responsável pelo reconhecimento do aparelho. (GIMENES, 2005)

O TKIP substitui o WEP com um novo método de criptografia que é mais forte que o algoritmo WEP e ainda pode ser utilizado usando o poder computacional presente no hardware de equipamento wireless. (ROCHA, 2006)

O TKIP também dá o suporte de verificação da configuração de segurança depois de determinar a chave de criptografia e a alteração de sincronização da chave de criptografia. (Rocha,2006)

Os produtos que implementaram o WPA têm suporte para WEP, TKIP e autenticação de usuários 802.1X. (AMARAL, MAESTRELLI, 2005)

O WPA define o uso do AES (Advanced Encription Standart), como uma substituição opcional para criptografia WEP. Pelo fato de não ser possível o suporte AES através de atualização de firmware em equipamentos sem fio existentes, este suporte para adaptadores de redes sem fio e nos pontos de acesso não é necessário.

O WPA veio com o intuito de corrigir os problemas de criptografia da WEP, sem o usuário necessitar trocar de hardware, o TKIP (*Temporal Key Integrity Protocol*) substitui o WEP

com um novo algoritmo de criptografia que é mais forte que o algoritmo WEP e ainda pode ser executado usando as facilidades de cálculo presente no hardware existente do equipamento wireless. (ROCHA, 2006)

O WPA suporta chaves de 40 a 104 bits com vetor inicialização de 24 bits, e a combinação de 104 bits da chave com os 24 bits do vetor de inicialização gera uma chave de 128 bits.

3.3. WPA2

O WPA2 possui dois componentes de criptografia e autenticação os quais são cruciais para a segurança de uma rede wireless. Ele utiliza dos modelos de criptografia AES e TKIP, porem o método TKIP está presenta para modelos antigos que não possuem compatibilidade com o AES. (ARANA, 2006)

O método de autenticação do WPA2 é o mesmo do WPA, utilizando o PSK e o método de autenticação utilizado por empresas EAP. (ARANA, 2006)

O padrão WPA2 teve obteve melhorias na segurança em redes sem fio, que aperfeiçoaram as técnicas de segurança correspondentes à proteção dos dados e aos acessos e autenticações dos usuários dos dois padrões anteriores com uma mudança principal, o método de criptografia, com o AES (*Advanced Encryption Standard*), que também substitui o WEP por um algoritmo de criptografia bem mais forte. Como o TKIP do WPA, o AES permite a descoberta de uma chave de criptografia de difusão ponto a ponto inicial exclusiva para cada autenticação, bem como a alteração sincronizada da chave de criptografia de difusão ponto a ponto para cada quadro. (GIMENES, 2005)

O protocolo WPA2 utiliza chaves de 128, 192 e 256 bits, assim sendo uma ferramenta poderosa de criptografia com a utilização dessa nova ferramenta precisou de novos hardwares, capaz de realizar o processo de criptografia.

Apesar do WPA2 possuir um padrão de criptografia muito mais otimizado ele não será capaz de trabalhar com algumas placas de rede mais antigas. A Wi-Fi Alliance refere-se a sua implementação de todo o 802.11i como WPA2. (ALVES, SOUSA, 2016)

4. KALI LINUX

Kali Linux[5] é uma distribuição Linux baseada em Debian destinada em testes avançados de penetração e auditoria de segurança. Dentro do sistema operacional do Kali Linux estão incluídas centenas de ferramentas para realizar uma série de tarefas de segurança da informação, incluindo testes de penetração, pesquisa de segurança, forense informática e engenharia reversa, cada ferramenta é adaptada para qualquer tipo de serviço, desde auditoria até penetração.

O Kali Linux foi lançado em 2013 pela Offensive Security, ela é uma provedora de treinamento de segurança e de penetração em nível de classe mundial. (https://www.kali.org/offensive-security-introduces-kali-linux/)

O Kali foi desenvolvido como uma reconstrução completa do BackTrack Linux, mantendo completamente o desenvolvimento de padrões Debian. (https://www.kali.org/offensive-security-introduces-kali-linux/)

O Kali Linux possui benefícios e desvantagens como qualquer outra distribuição Linux. O usuário que irá determinar a escolha entre *pen-tester*, simples usuário, ou desenvolvedor.

O Kali é uma distribuição focada em profissionais da área de penetração ou especialistas em segurança e considerando a forma com que ele foi desenvolvido, ele não é recomendado a pessoas que não estão familiarizadas com Linux ou aqueles que estão à procura de propósitos gerais, como, jogos, web design ou desenvolvimento.

A Distribuição do Kali é completamente *open source* e é completamente acessível aos usuários. Toda a codificação dele é facilmente visível para qualquer usuário e a arvore de desenvolvimento também possibilita aos usuários a checagem da codificação em cada etapa.

Todas as ferramentas que este sistema operacional possuí foram avaliados em relação à adequação e eficácia antes de serem incluídos. Entre eles estão incluídos o Metasploit para testes de penetração de rede, Nmap para varredura de portas e vulnerabilidades, Wireshark para monitoramento de tráfego de rede e o Aircrack-ng para testar a segurança de redes sem fio. Estas são algumas das principais ferramentas do Kali Linux. Como seu antecessor, o Kali Linux é completamente gratuito e sempre será. A Offensive Security

[5] https://www.kali.org/offensive-security-introduces-kali-linux/

está emprenhada em apoiar a comunidade de código aberto como desenvolvimento contínuo do Kali Linux.

4.1. AIRCRACK-NG

Aircrack-ng[6] foi iniciado no final de fevereiro de 2006. Aircrack-ng é um conjunto completo de ferramentas para avaliações de segurança em redes wireless, ele é uma suíte de ferramentas para auditoria em redes wireless, ele é composto por diversas ferramentas, como o Airdump-ng, Aireplay-ng, Packetforge-ng, Aircracrack-ng, Eassid-ng, Airbase-ng entre outras. (MORENO, 2016)

Todas suas ferramentas são utilizadas em linhas de comando, e seu principal uso é em mbientes linux, porém ele também pode ser executado em diferentes sistemas operacionais, como, Windows, OS X, FreeBSD, OpenBSD, NetBSD, Solaris e eComStation2.

O Aircrack-NG foca em diferentes áreas de segurança em redes wireless, entre elas estão o monitoramento, que faz a captura de pacotes e exportação de dados para arquivos texto para processamento futuro por ferramentas de terceiros. O Ataque, repetição de ataques, de autenticação, pontos de acesso falsos e outros. Os Testes, checando redes wireless e suas compatibilidades com drivers, captura e injeção de dados; E a o *Cracking* que faz a captura de senhas de todos os métodos de criptografia, WEP, WPA, WPA2.

4.1.1. Airmon-ng

O Airmon-ng é uma das ferramentas da suíte aircrack-ng, ele é utilizado para checar e finalizar processos que podem atrapalhar a execução do programa, ele também é utilizado para transformar interfaces wireless em interfaces de monitoramento, de forma que essas interfaces capturem pacotes de APs. (MORENO, 2016)

[6] https://www.aircrack-ng.org/doku.php

Uma placa wireless possui duas maneiras de trabalhar, modo *managed* ou modo *monitor*, uma mesma interface não pode realizar o mesmo trabalho ao mesmo tempo, ou ela se conecta a uma rede e utiliza-se da mesma, ou, realiza testes em modo de monitoramento. (MORENO, 2016)

4.1.2. Airodump-ng

O Airodump-ng é uma ferramenta utilizada para a captura de pacotes de uma rede wireless. Com os pacotes capturados, podemos visualizar diversas informações das redes, como, o endereço MAC do AP, qual método de criptografia utilizado pelo AP, canal de operação, se as redes estão sendo utilizadas ou não, numero de pessoas conectadas as redes, velocidade máxima de transmissão de dados, entre outros. (MORENO, 2016)

4.1.3. Aireplay-ng

O Aireplay-ng é uma ferramenta de ataque, que contem múltiplos vetores de ataques que podem ser combinadas com outras ferramentas, como o Packetforge-ng e o Aircrack-ng.

Os principais métodos de ataque utilizados pelo Aireplay-ng são os, Testes de injeção, Death, Fake Auth, Modo Interativo, ARP Replay, Chop Chop, Fragmentação. (MORENO, 2016)

4.1.3.1. Testes de Injeção

Os testes de injeção não são realmente um ataque, ele apenas verifica se a interface wireless suporta a injeção de pacotes.

4.1.3.2. Deauth

O ataque de Deauth explora a fraqueza de redes wireless na forma de não existir nenhum mecanismo de segurança que informe que pacotes Deauth devem ser enviados somente da estação até o AP, pedindo desautenticação. (MORENO, 2016)

Em um ataque de Deauth com o Aireplay-ng é possível forjar pacotes Deauth em nome do AP e envia-los diretamente a maquina do cliente, fazendo com que sua conexão seja negada. (MORENO, 2016)

Neste tipo de ataque é possível desautenticar o cliente apenas uma vez ou mante-lo desautenticado por tempo indeterminado, até que o atacante decida o momento de parar. (MORENO, 2016)

Existem diversas finalidades para o ataque de Deauth, como, manter o usuário em um estado de DOS (Denial of Service), negação de serviço, apenas para mantes o usuário sem acesso a rede, além disso pode utilizar-se em conjunto com o DOS a *Evil Twin/honeyspot,* uma rede falsa com o mesmo nome, fazendo com que o usuário se conecte nesta rede para que ele consiga pegar os dados da verdadeira rede. Outra forma de utilizar o Deauth é para ataques na criptografia, capturando o *keystream* ou o *four-way handshake e gerar a ARP Request.* (MORENO, 2016)

4.1.3.3. Fake Auth

A falsa associação consiste em associar-se à rede pulando o estagio de autenticação. Atente-se que estar associado a rede não necessariamente compreendera o trafego dela, pois ele estará todo criptografado, portanto, não é possível obter um endereço DHCP.

Este tipo de ataque é utilizado em APs com o tipo de criptografia WEP, quando não há nenhum cliente conectado à rede, ou em ataque de negação de serviço por *Association Flood*, várias estações falsas associadas a um AP, fazendo com que em pouco tempo a tabela ARP do AP fique superlotada, ocasionando a perda de sinal ou a reinicialização do dispositivo. (MORENO, 2016)

4.1.3.4. Modo Interativo

Neste ataque é possível selecionar o tipo de pacote que ira interagir com a rede. O conceito deste ataque é injetar tráfego de dados na rede, aumentando de maneira rápida o campo Data para posteriormente quebrar a chave WEP. Para injetar dados na rede, é necessário que o tipo do pacote seja apenas para aumentar outros pacotes, com o vetor de inicialização duplicado. O principal tipo de pacote que aumenta o índice do vetor é o ARP Replay. (MORENO, 2016)

O Ataque de Modo interativo vai injetar o pacote desejado na rede em questão, e o AP vai responder com pacotes ARP Replay, aumentando o trafego na rede e consequentemente os pacotes com índice de inicialização. Existem duas maneiras de se fazer isso, a primeira, é utilizando o método de *Fake Authentication* e utilizar o modo interativo, o segundo é utilizar o modo interativo com o endereço MAC de algum cliente conectado. (MORENO, 2016)

4.1.3.5. ARP Replay

O ataque de ARP Replay é bem semelhante ao modo interativo, porem o este ataque já captura pacotes ARP Request e retransmite ARP Replay, sendo mais efetivo.

Uma forma de utilizar o ARP Replay é utilizar do endereço MAC de algum cliente já conectado, de forma que as requisições aumentem. Caso as requisições não alterem com o tempo, pode-se resolver isso de algumas formas, como, enviar o pacote ICMP para um ip válido que não esteja em uso, utilizar de pacotes Deauth, para forçar o cliente a reassociar-se a rede, gerando pacotes ARP Request/Replay ou conectar-se a um cliente legitimo da rede e fazer solicitações ARP Request, povoando a tabela MAC. (MORENO, 2016)

4.1.3.6. Chop Chop

O ataque *Chop Chop* permite descriptografar um pacote WEP sem o conhecimento da senha, pois com esse ataque são gerados dois arquivos, .cap e .xor, o arquivo .xor

contém o *keystream*, com o PRGA (Algoritmo Pseudoaleatório Utilizado na Cifragem da Senha). (MORENO, 2016)

A vantagem desse modo de ataque é que não é necessário nenhum usuário conectado à rede, sendo assim, facilitando a obtenção da *keystream*.

Esse modo de ataque pode ser utilizado de duas formas, a primeira é utilizando o *Fake Authentication* e depois o *Chop Chop*, outra forma é utilizando diretamente o *Chop Chop* em um endereço MAC de algum cliente conectado à rede. (MORENO, 2016)

4.1.3.7. Ataque de Fragmentação

O Ataque de Fragmentação é bem similar ao ataque *Chop Chop*, pois ele também captura o PRGA.

Possui os dois mesmos métodos de se utilizar, com o *Fake Authentication* e o ataque de fragmentação, e utilizando o endereço MAC de um cliente já conectado à rede.

Normalmente quando um AP é suscetível a o ataque de *Chop Chop*, ele não será ao Ataque de Fragmentação e vice-versa.

4.1.4. Packetforge-ng

A ferramenta Packetforge-ng é utilizada para a criação de pacotes. Seu uso mais comum é parar criar pacotes customizados de ARP Replay e utiliza-lo em conjunto com o ataque interativo para geração de pacotes com o vetor de inicialização duplicado. (MORENO, 2016)

4.1.5. Wordlist

Wordlist é um dicionário contendo os possíveis caracteres da criptografia. Existem diversas *wordlists* que são disponibilizados para download na internet, porém, podem

conter trojans embutido na *wordlist*. O mais indicado e seguro é criar a *wordlist* utilizando softwares que auxiliam na criação. (COSTA, Bruno Nogueira Lima et al, 2016)

O Kali Linux possui uma aplicação que já vem integrada ao sistema operacional chamada *crunch*. Essa ferramenta pode ser utilizada para montar sua própria *wordlist*. (COSTA, Bruno Nogueira Lima et al, 2016)

O interessante dessa aplicação e das próprias *wordlists* é a utilização de regras para a sua criação pois se utilizássemos todas letras, números, e caracteres especiais estaríamos criando uma *wordlist mixalpha-numeric-all-space* geraria uma *wordlist* de aproximadamente 56 Terabytes, sendo necessário cancelar a operação por falta de meios. (COSTA, Bruno Nogueira Lima et al, 2016)

4.2. PYTHON

Python[7] é uma linguagem de programação criada por Guido van Rossum em 1991. Os principais objetivos da linguagem são a produtividade e legibilidade. Python é uma linguagem que foi criada para produzir código bom e fácil de manter de maneira rápida. As principais características que ressaltam esses aspectos é o seu baixo uso de caracteres especiais, o uso de identação para marcar bloco, seu coletor de lixo para gerenciar automaticamente o uso da memória e quase nenhum uso de palavras-chave voltadas para a compilação.

Python tem uma biblioteca padrão imensa, que contém classes, métodos e funções para realizar essencialmente qualquer tarefa, desde acesso a bancos de dados a interfaces gráficas com o usuário, ela também possui muitas ferramentas para lidar com dados científicos.

Essa característica da linguagem é comumente chamada baterias inclusas, significando que tudo que você precisa para executar um programa está presente na instalação básica.

[7] https://docs.python.org/3/license.html

4.2.1. Biblioteca Python – Requests

As *Requests*[8] permitem que você envie solicitações, *grass-fed* HTTP / 1.1, sem a necessidade de mão-de-obra manual. Não é necessário adicionar manualmente as *strings* de consulta aos seus URLs ou codificar os seus dados POST. O *keep-alive* e o pool de conexões HTTP são 100% automáticos, graças ao urllib3.

4.2.2. Biblioteca Python - urllib2

O módulo urllib2 define funções e classes que ajudam na abertura de URLs (principalmente HTTP) em um mundo complexo - autenticação básica e digest, redirecionamentos, cookies e muito mais. (https://docs.python.org/2/library/urllib2.html)

4.2.3. Biblioteca Python - base64[9]

Este módulo fornece codificação e decodificação de dados conforme especificado no RFC 3548. Este padrão define os algoritmos Base16, Base32 e Base64 para codificar e decodificar strings binárias arbitrárias em strings de texto que podem ser enviadas com segurança por email, usando as partes de URLs ou incluídas como parte de uma solicitação HTTP POST. O algoritmo de codificação não é o mesmo que o programa uuencode.

Existem duas interfaces fornecidas por este módulo. A interface moderna suporta a codificação e decodificação de objetos *string* ambos usando os alfabetos base-64 definidos no RFC 3548 (normal e protegido por URL e sistema de arquivos). A interface legada fornece codificação e decodificação de e para objetos semelhantes a arquivos, bem como strings, mas usando apenas o alfabeto padrão Base64.

[8] http://docs.python-requests.org/en/master/
[9] https://docs.python.org/2/library/base64.html

4.2.4. Biblioteca Python – hashlib[10]

Este módulo implementa uma interface comum para diferentes algoritmos de secure hash e message digest. Incluem-se os algoritmos de secure hash, FIPS SHA1, SHA224, SHA256, SHA384 e SHA512 (definidos no FIPS 180-2), bem como o algoritmo MD5 da RSA (definido na Internet RFC 1321).

Os termos secure hash e message digest são intercambiáveis. Algoritmos mais antigos eram chamados message digest. O termo moderno é secure hash.

[10] https://docs.python.org/2/library/hashlib.html

5. METODOLOGIA

A metodologia do presente trabalho consiste na utilização das ferramentas e linguagens apresentadas em um AP, para consolidarem-se suas vulnerabilidades e demonstrar como podemos torná-los menos vulneráveis ou se possível invulneráveis.

5.1. UTILIZAÇÃO DO CRUNCH

O crunch, como foi descrito anteriormente, foi utilizado como necessidade para criar *wordlists* para utilização em parceria com o Aircrack-NG.

Para verificar se o crunch está instalado e sua versão, no terminal utiliza-se apenas seu nome “crunch”, como a Figura 1 ilustra:

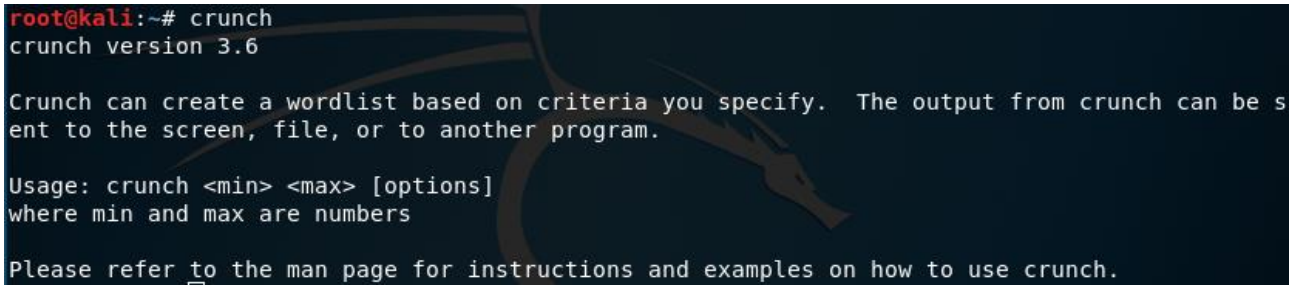

Figura 1: Verificação do crunch
Fonte: Próprio Autor

Para utilizarmos o crunch, será utilizado o comando “crunch 8 8 0123456789 wordlist.txt”, onde,

- “8” é o número de caracteres da palavra inicial da wordlist; Ex: 00000001
- “8” é o numero de caracteres da palavra final da wordlist; Ex: 99999999
- “0123456789” é o alfabeto a ser utilizado pela wordlist, com o alfabeto provido à ele gerará todas as possíveis combinações com o tamanho mínimo “<min>” e máximo “<max>” de caracteres possíveis.

- “Wordlist.txt” é o nome do arquivo de saída, poderia utilizar qualquer nome para esse parâmetro.

A Figura 2 ilustra o código sendo executado pelo Kali Linux.

Figura 2: Utilização do crunch na criação de wordlists
Fonte: Próprio Autor

O crunch também pode ser utilizado com regras, caso uma parte da senha seja conhecida é possível criar um filtro na criação das *wordlists*, utilizando um parâmetro a mais, “crunch 8 8 0123456789 –t 0@@@@@@@@ wordlist.txt”, onde

- “-t” 0@@@@@@@@ é uma regra a qual gerará todos as palavras da lista iniciando com 0.

A Figura 3 ilustra a utilização do comando.

Figura 3: Utilização do crunch na criação de wordlists com regras
Fonte: Próprio Autor

5.2. UTILIZAÇÃO DO AIRCRACK-NG

A primeira ferramenta utilizada foi o Aircrack-ng, com ele foi verificado quais processos estão ativos no momento que podem atrapalhar os comandos que serão utilizados futuramente. Para visualizar os processos utiliza-se o comando “airmon-ng check”, como é ilustrado na Figura 4.

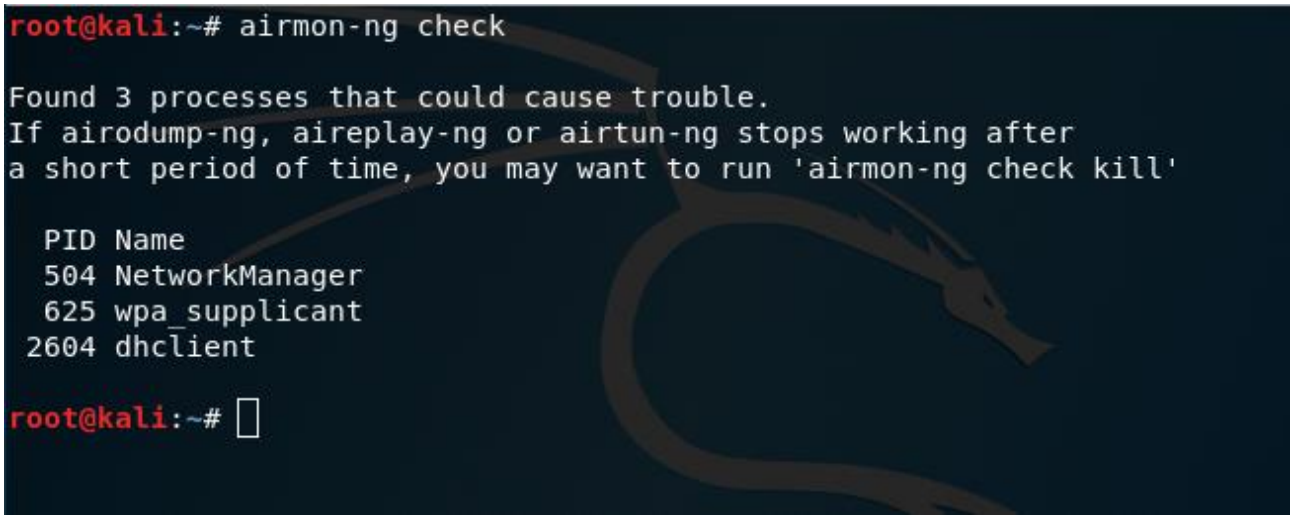

Figura 4: Checagem de processos ativos que podem intervir na utilização do aircrack-ng

Fonte: Próprio Autor

Após verificar quais processos estão ativos utiliza-se o comando “airmon-ng check kill” para matar todos os presentes processos, como pode ser visualizado na Figura 5.

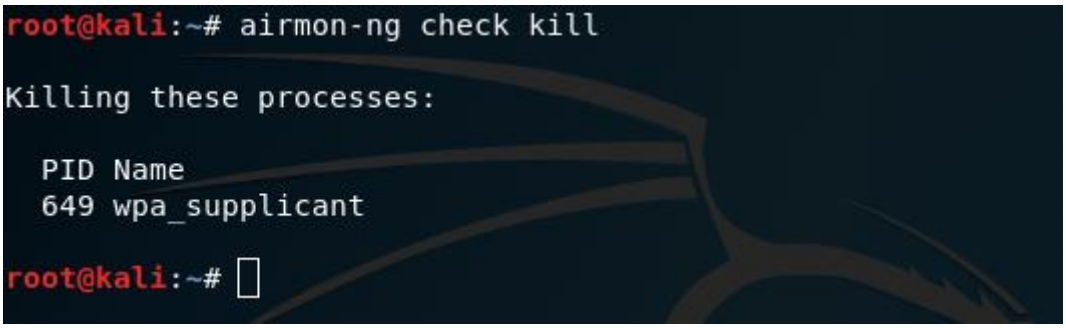

Figura 5: Finalização dos processos que poderiam atrapalhar a execução do aircrack-ng

Fonte: Próprio Autor

Após os processos finalizados, verifica-se quais são as interfaces de redes ativas, para visualizá-las, o comando utilizado é o “iwconfig”, com esse procedimento executado, pode-se visualizar as interfaces de rede, as quais

- “wlan1mon” interface de rede em modo de monitoramento
- “eth0” interface de rede cabeada
- “lo” interface de *loopback*
- *“wlan0” interface de rede wireless em modo de gerenciamento*

Como é ilustrado pela Figura 6.

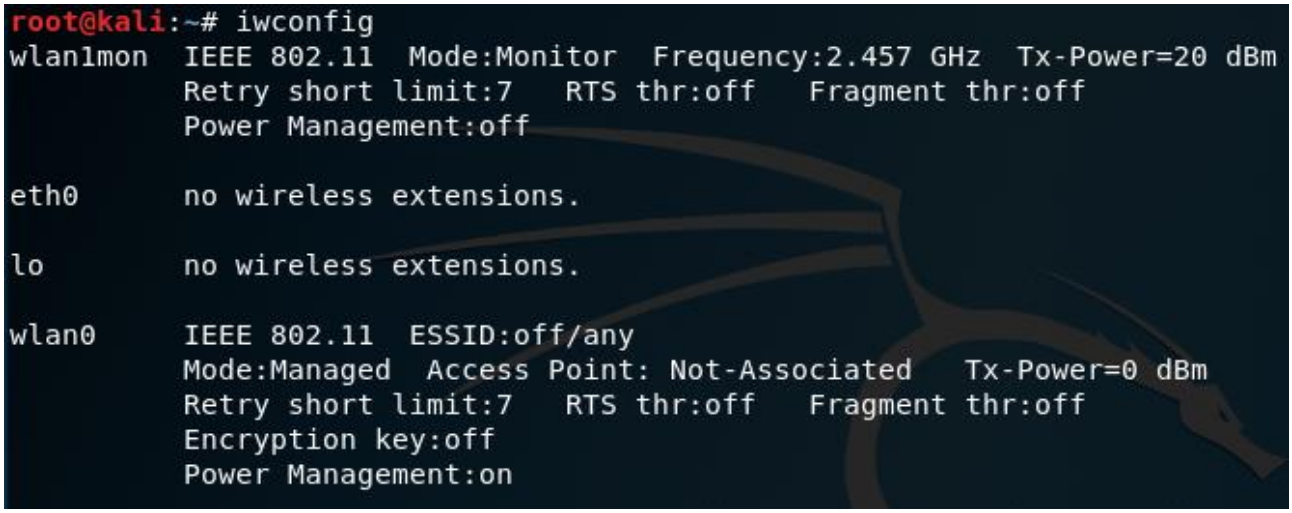

Figura 6: Verificação das interfaces pelo terminal do Kali Linux

Fonte: Próprio Autor

Com os nomes das interfaces de rede em mãos, podemos utilizar um comando do Kali que mostrará todas redes wireless que estiverem ao alcance, obter seu MAC e qual canal ela está operando, para efetuar a busca utiliza-se o comando “iwlist (interface de rede em modo de gerenciamento) scanning”, como pode ser visualizado a Figura 7.

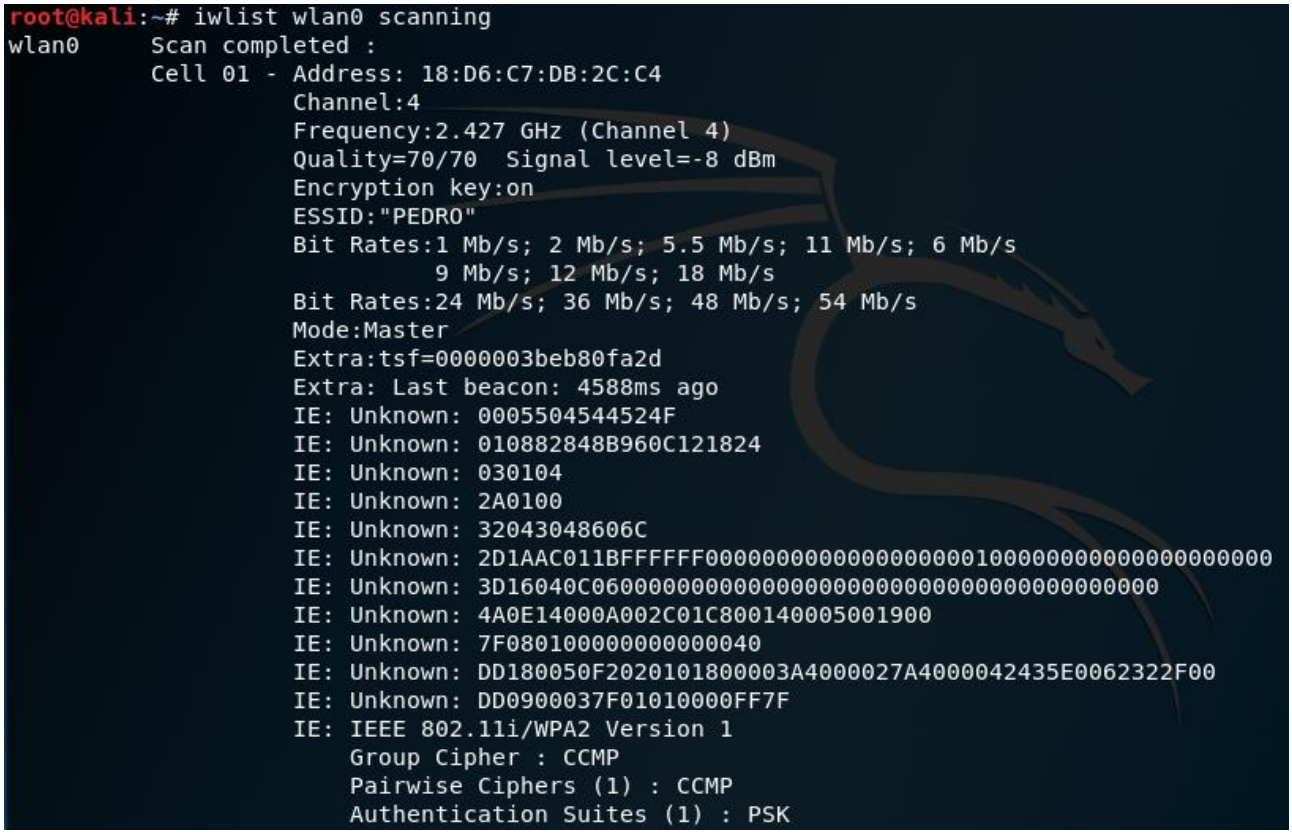

Figura 7: Utilizando as interfaces de rede para escanear por redes próximas

Fonte: Próprio Autor

Com isso, deve-se iniciar/*startar* a interface wireless em modo de monitoramento, com o comando “airmon-ng start (interface de rede wireless)”, como pode ser visualizado na Figura 8.

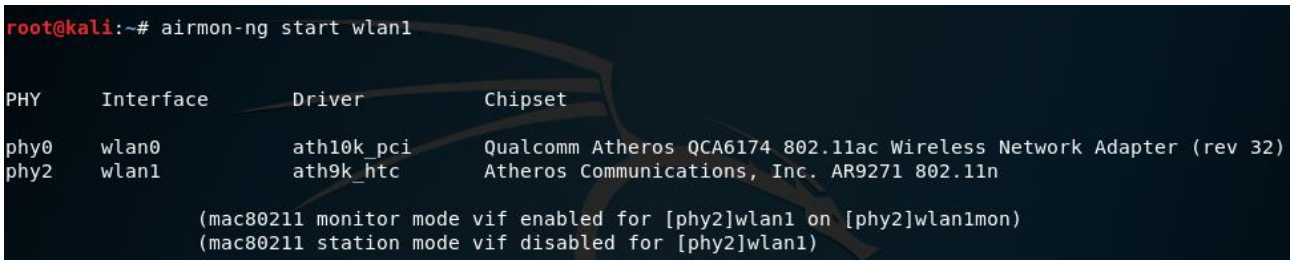

Figura 8: Utilizando o aircrack-ng para iniciar a interface de rede em modo de monitoramento

Fonte: Próprio Autor

Com a interface em modo de monitoramento utiliza-se o airodump-ng, para buscar clientes que estão conectados à rede que buscamos com o “iwlist scanning”, como demonstra a Figura 9.

Figura 9: Utilizando o airodump-ng para fazer a varredura na rede

Fonte: Próprio Autor

O parâmetro “wlan1mon” é a interface de monitoramento, o parâmetro “–bssid” se refere ao endereço MAC da rede, o parâmetro “–channel” é referente ao canal em que a rede esta operando e o parâmetro --write serão os arquivos que armazenarão os pacotes que forem capturados com os futuros ataques executados pelo aircrack-ng, o nome ‘’senhawifi’’ será o nome dos arquivos criados, esse parâmetro pode receber qualquer nome.

Assim que o comando for executado, será aberto o terminal com a rede em operação, com os usuários conectados, como demonstra a Figura 10.

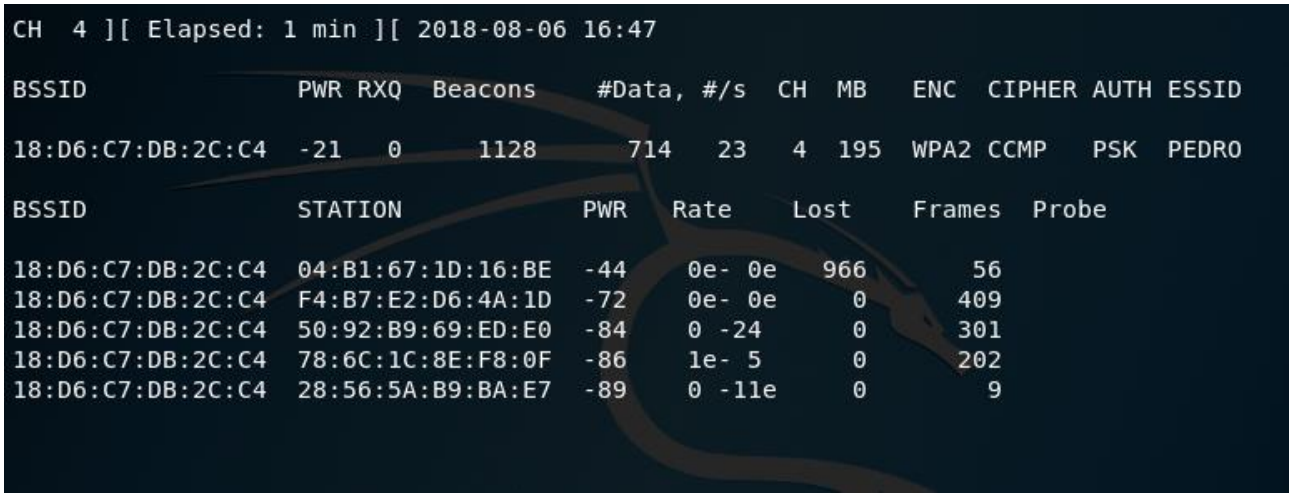

Figura 10: Listagem dos usuários conectados em uma rede wireless

Fonte: Próprio Autor

A partir do momento que é possível visualizar o MAC de um cliente da rede, pode-se utilizar o ataque de Deauth com o Aireplay-ng, o comando a ser utilizado é o “aireplay-ng -0 100 –a 18:D6:DB:2C:C4 –c 04:B1:67:1D:16:BE wlan1mon”, onde

- “-0” é o parâmetro de modo de ataque, no caso -0 é o pertencente ao Deauth;
- “-a” endereço MAC da rede no caso 18:D6:DB:2C:C4;
- “-c” endereço MAC do cliente, no caso 04:B1:67:1D:16:BE;
- “wlan1mon” é a interface em modo de monitoramento;

Como é ilustrado na Figura 11.

Figura 11: Listagem dos usuários conectados em uma rede wireless

Fonte: Próprio Autor

No momento em que o comando for executado, o ataque começará como a Figura 13 demonstra.

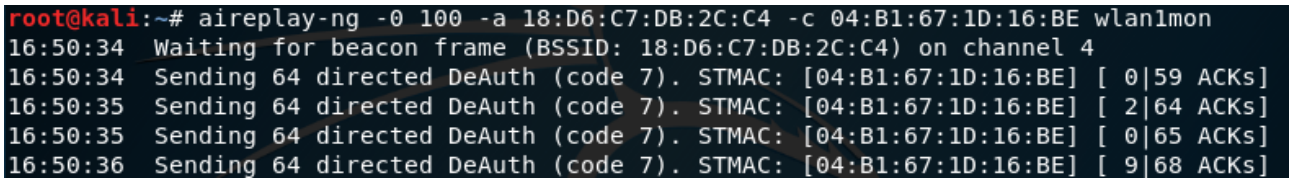

Figura 12: Ataque Deauth sendo executado

Fonte: Próprio Autor

Com a utilização do ataque Deauth, o cliente irá ser forçadamente desconectado da rede, quando o ataque parar, o cliente irá automaticamente tentar se conectar à rede, no momento que isso acontece, o aircrack captura os pacotes que trafegam na parte de trafego, esses pacotes contem a senha criptografa, e esses pacotes serão salvos nos arquivos que criamos antigamente, que os demos de nome “senhawifi”.

Com isso, foi utilizado a suíte do aircrack-ng para fazer o *brute-force*, o qual utilizará dois parâmetros, um contendo o arquivo de captura de pacotes, sua extensão é .cap, e será utilizado uma *Wordlist* que foi gerada pelo crunch, o comando ficará “aircrack-ng senhawifi-01.cap –w wordlist.txt “, como a Figura 14 ilustra.

```
root@kali:~# aircrack-ng senhawifi-01.cap -w wordlist.txt
```

Figura 13: Utilização da suíte aircrack para fazer o *brute-force*

Fonte: Próprio Autor

Então ele começará a executar os testes repetidamente, utilizando toda a *wordlist*, caso a senha esteja contida no presente *wordlist*, ele mostrará que a senha foi encontrada, como apresenta na Figura 14.

Figura 14: Senha encontrada com sucesso

Fonte: Próprio Autor

5.3. UTILIZAÇÃO DO SCRIPT EM PYTHON

O script em python possui algumas variáveis em seu inicio, como ilustra a Figura 16, cada uma dessas variáveis irá executar o script de uma maneira diferente, sendo

- “usuario” o *login* utilizado para acessar o roteador;
- “senha” a senha utilizada para acessar o roteador;
- “ip” o caminho do host para acessar o roteador;
- “utilizar_controle_do_wifi” utilizado para condicionar o uso do modo de operação para desligar e ligar o wifi pelo script, passar o valor “s” para utilizar desse modulo e qualquer outro valor para não utilizar.
- “controle_do_wifi” utilizado para ligar ou desligar a rede wireless de um roteador; o valor “*True*” ativa ou mantem ativada o wireless da rede, enquanto o “*False*” desliga ou mantem desliga o wireless da rede.
- “reiniciar_roteador” utilizado para reiniciar o roteador quando passado o valor “*True”*, caso seja passado o valor *“False*” essa parte do script não sera executada.
- “resetar_roteador” utilizado para resetar o roteador com seu modelo de configuração prévio enviado pela fábrica, caso passado o valor “*True*”, caso o valor passado seja “*False*” essa parte do script não sera executada.

```
usuario = 'admin'
senha = 'admin'
ip = '192.168.0.1'
utilizar_controle_do_wifi = 's' # Para utilizar do controle do wifi, utilizar 's', caso nao queira ativar ou desativar o wifi, utilizar ''
controle_do_wifi = False # True = Ativa Wireless  False = Desativa Wireless
reiniciar_roteador = False # True = Reinicia o roteador  False = Não reinicia
resetar_roteador = False # True = Reseta o roteador  False = Não Reseta o Roteador
```

Figura 15: Variáveis do script em python

Fonte: Próprio Autor

O script pode executar apenas uma operação por vez, por exemplo, ele apenas pode reiniciar o roteador sem desligar a rede sem fio e sem voltar para modelo de fabricação, ou ele apenas pode ligar ou desligar o roteador, ou apenas voltar para o modelo de fabricação.

O script em python pode ser executado pelos terminais, desde que a distribuição utilizada possua o python 2.7 instalado, para fazer uso do script o comando “python nomedoarquivo.py” pode ser usado em qualquer terminal como a Figura 16 ilustra.

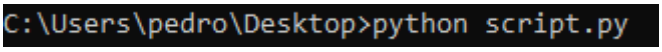

Figura 16: Execução do Script no Sistema Operacional Windows 10

Fonte: Próprio Autor

5.4. CONCLUSÃO

Sabe-se que não há nenhum sistema totalmente seguro, senhas podem ser descobertas ou “quebradas” por ferramentas de testes de vulnerabilidades como, por exemplo, *Brute-Force*, isso dependerá apenas da quantidade de testes e do poder computacional da máquina utilizada. Por conta disso, a utilização de senhas mais complexas sempre deve ser uma prioridade em qualquer tipo de rede wireless.

Bloquear ataques ao roteador é uma contramedida complicada de ser realizada, pois são redes que estão propagando-se pelo ar, porém existem algumas medidas que podem ser realizadas, como, manter o *firmware* dele sempre atualizado, pois as empresas de desenvolvimento estão em constante aprimoramento de seus modos se segurança. Em sua configuração utilizar sempre as criptografias mais recentes, nunca deixar a utilização de senhas padrão, como, *login:*“*admin*”, *password*:“*admin*”.

6. TRABALHOS FUTUROS

Para trabalhos futuros, fica a melhoria do script em python, podendo utilizá-lo por meio de uma interface gráfica, a implementação de testes de *brute-force* e o aprendizado da captura de pacotes contendo dados trafegando pela rede.

O trabalho foi de extrema importância para o aprendizado da comunidade, uma vez que poderá ser publicado em eventos relacionados a segurança da informação bem como em mídias sociais.

7. REFERÊNCIAS

ALECRIM, Emerson. **O que é Wi-Fi (IEEE 802.11)?**. Disponível em <https://www.infowester.com/wifi.php>. Acesso em: 05 de maio e 2018.

ALVES, Lucas Pereira; SOUZA, Lucas Vieira de. Análises dos Protocolos de Segurança, WEP, WPA e WPA2 em redes sem fio: Um estudo. In: CONGRESSO DE SISTEMAS DE INFORMAÇÃO DO ITPAC, 1, 2016, Araguaína, Brasil. **I Congresso de Sistemas de Informação do ITPAC**, 1, Outubro, 2016, p.9-14.

ANDRADE, Lidiane Parente et al**. ANÁLISE DAS VULNERABILIDADES DE SEGURANÇA EXISTENTES NAS REDES LOCAIS SEM FIO: UM ESTUDO DE CASO DO PROJETO WLACA.** Universidade Federal do Pará, Belém, v. 15, 2008.

AMARAL, Bruno Marques; Maestrelli, Marita. Segurança em Redes Wireless 802.11. In: **CBPF-NT-002/04**, p.38.

ARANA, Paul. **Benefits and vulnerabilities of Wi-Fi protected access 2 (WPA2)**. INFS, v. 612, p. 1-6, 2006.

COSTA, Bruno Nogueira Lima et al. **Pentest para Quebra de Criptografia Wireless**. Caderno de Estudos Tecnológicos, v. 4, n. 1, 2016.

GIMENES, Eder Coral. **SEGURANÇA DE REDES WIRELESS**. 2005. 58. Monografia – Departamento – Mauá, FATEC, SP, Mauá, 2005.

MORENO, Daniel. **Pentest em Redes Sem Fio**, Primeira Edição. São Paulo: Novatec Editora Ltda., 2016.

ROCHA, João Wilson Vieira. **Redes WLAN de Alta Velocidade II: Recomendações Aplicáveis.** Disponível em <http://www.teleco.com.br/tutoriais/tutorialredeswlanII/pagina_3.asp >. Acesso em: 19 junho e 2018.

8. APENDICE

8.1. CÓDIGO EM PYTHON PARA DESLIGAR O WIFI, REINICIAR OU RESETAR O ACCESS POINT

```
# -*- coding: utf-8 -*-

import requests

import urllib2

import base64

import hashlib


# Credenciais de acesso ao roteador #


usuario = 'admin'

senha = 'admin'

ip = '192.168.0.1'

utilizar_controle_do_wifi = 's' # Para utilizar do controle do wifi, utilizar 's', caso nao queira ativar ou desativar o wifi, utilizar 'n'

controle_do_wifi = True # True = Ativa Wireless  False = Desativa Wireless

reiniciar_roteador = False # True = Reinicia o roteador  False = Não reinicia

resetar_roteador = False # True = Reseta o roteador  False = Não Reseta o Roteador


# Preparação do Cookie para o http request #


hash = hashlib.md5(senha)

hash_criptografado = hash.hexdigest()

parametro_do_cookie = usuario + ':' + hash_criptografado
```

```
parametro_do_cookie = base64.b64encode(parametro_do_cookie.encode('utf8'))
# HTTP request para o Login #

meuServidor = 'http://'+ ip +'/userRpm/LoginRpm.htm?Save=Save'
headers = {
'Host': ip,
'User-Agent':'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0',
'Accept':'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
'Accept-Language':'en-US,en;q=0.5',
'Accept-Encoding':'gzip, deflate',
'Referer':'http://'+ ip + '/',
'Cookie':'Authorization=Basic%20'+parametro_do_cookie,
'Connection':'keep-alive',
'Upgrade-Insecure-Requests':'1',
'Pragma':'no-cache',
 }

# HTTP request para habilitar ou desabilitar a interaface wireless #
header_control = {
'Host': ip,
'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0',
'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
'Accept-Language': 'en-US,en;q=0.5',
'Accept-Encoding': 'gzip, deflate',
'Referer': 'http://'+ip+'/BQKSCNDBZFMGDQGB/userRpm/WlanNetworkRpm.htm',
```

```
'Cookie': 'Authorization=Basic%20'+parametro_do_cookie,
'Connection': 'keep-alive',
'Upgrade-Insecure-Requests': '1',
}
# print do cookie para debug #
 #print "Cookie - "+ headers['Cookie'];
# HTTP response and tratamento #
req = urllib2.Request(meuServidor, None, headers)
UrlAberta = urllib2.urlopen(req)
codigo_html = UrlAberta.read()
controle_da_url = codigo_html[67:-27]
controle_da_url = controle_da_url[0:-9]
if controle_do_wifi == True and utilizar_controle_do_wifi == 's':
        controle_da_url = controle_da_url + "WlanNetworkRpm_AP.htm?operMode=0&ssid1=Netw0rk&channel=7&mode=6&chanWidth=2&rate=71&addrType=1&ap=1&broadcast=2&Save=Save"
if controle_do_wifi == False and utilizar_controle_do_wifi == 's':
        controle_da_url = controle_da_url + "WlanNetworkRpm_AP.htm?operMode=0&ssid1=Netw0rk&channel=7&mode=6&chanWidth=2&rate=71&addrType=1&ap=0&broadcast=2&Save=Save"
if utilizar_controle_do_wifi != 's':
        print "Modelo de Modificar o Wireless do Roteador Desabilitado"
""" print do control url para debug """
#print "Control Url 3 - " +control_url;
 ##Reiniciar##
if reiniciar_roteador == True:
        controle_da_url = controle_da_url + "SysRebootRpm.htm?Reboot=Reboot"
```

```
else:
        print "Modelo de Reiniciar Desativado";
 ##Resetar Roteador
if resetar_roteador == True:
        controle_da_url = controle_da_url + "RestoreDefaultCfgRpm.htm?Restorefactory=Restaurar"
else:
        print "Modelo de Resetar o Dispositivo Desativado";
""" ##Factory Reset
print do control_url para debug """
#print "Control URL - "+control_url
if "http://" in controle_da_url:
        req = urllib2.Request(controle_da_url, None, header_control)
        UrlAberta = urllib2.urlopen(req)
        html = UrlAberta.read()
        #print html
else:
print "Algum problema na execução do script”
```