UNIVERSIDADE DO SAGRADO CORAÇÃO

RODOLFO OLIVIERI SIVIERI

RECUPERAÇÃO DE HASHES CRIPTOGRAFADAS COM ALGORITMO MD5 APLICANDO A TÉCNICA DE ATAQUE HÍBRIDO: BRUTE-FORCE E DICTIONARY ATTACK

BAURU
2017

RODOLFO OLIVIERI SIVIERI

RECUPERAÇÃO DE HASHES CRIPTOGRAFADAS COM ALGORITMO MD5 APLICANDO A TÉCNICA DE ATAQUE HÍBRIDO: BRUTE-FORCE E DICTIONARY ATTACK

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

BAURU
2017

Sivieri, Rodolfo Olivieri

S6247r

Recuperação de hashes criptografadas com algoritmo md5 aplicando a técnica de ataque híbrido: brute-force e dictionary attack / Rodolfo Olivieri Sivieri. -- 2017.

53f. : il.

Orientador: Prof. M.e Henrique Pachioni Martins.

Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) - Universidade do Sagrado Coração - Bauru - SP

1. Segurança da informação. 2. Vulnerabilidades. 3. Criptografia. 4. MD5. 5. Criptoanálise. I. Martins, Henrique Pachioni. II. Título.

RODOLFO OLIVIERI SIVIERI

RECUPERAÇÃO DE HASHES CRIPTOGRAFADAS COM ALGORITMO MD5 APLICANDO A TÉCNICA DE ATAQUE HÍBRIDO: BRUTE-FORCE E DICTIONARY ATTACK

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

Bauru, 21 de novembro de 2017.

Banca examinadora:

_____
Prof. Me. Henrique Pachioni Martins
Universidade do Sagrado Coração

_____
Prof. Me. Renan Caldeira Menechelli
Universidade do Sagrado Coração

_____
Prof. Dr. Elvio Gilberto da Silva
Universidade do Sagrado Coração

AGRADECIMENTOS

Primeiramente, agradeço imensamente a minha mãe, Cynthia Lacotissa Olivieri, e a toda minha família por sempre me apoiar e me darem as melhores condições possíveis para que eu pudesse ter chegado a este momento, por toda educação e ensinamentos me dado ao longo desta vida.

Agradeço também, a todos os meus professores ao longo da vida, certamente eu não chegaria aqui sem nenhum conhecimento transmitido por eles. Agradeço muito também, a todos os professores durante esses 4 anos, que foram fundamentais para minha formação em bacharel em ciência da computação em especial, meu orientador Prof. Me. Henrique Pachioni Martins.

Gostaria também de agradecer especialmente a minha melhor amiga, Nicoly pela companhia e amizade durante todos esses anos. Sem ela certamente não teria chegado até o final dessa jornada.

Não poderia esquecer de agradecer também a todos meus amigos que fiz durante a graduação, por sempre estarem ao meu lado. É com toda certeza que agradeço a todos por proporcionarem tantos momentos de alegria e diversão durante a vida acadêmica.

RESUMO

A criptografia é um conjunto de técnicas para camuflar informações legíveis de pessoas que não podem ter acesso a elas, essas informações são camufladas através de uma chave gerada pelo algoritmo de encriptação (RSA, MD5, AES entre outras). Essas informações são chamadas de Cifras, as cifras por sua vez são indecifráveis até que você tenha o mesmo algoritmo de encriptação. Há dois tipos de chaves criptográficas que podem ser geradas a partir do algoritmo usado, chaves simétricas e chaves assimétricas. Na segurança digital é recomendado sempre utilizar as técnicas de criptografia, evitando assim que sejam perdidas as informações valiosas ou acabem parando em mãos indevidas. O objetivo deste trabalho é demonstrar por meio de comparações de tempo e tamanho a eficiência e segurança do algoritmo MD5. O MD5 tem como função de distribuição criptográfica de 128bits criado pela empresa RSA Data Security Inc., utilizado principalmente em softwares que utilizam o protocolo P2P (*Peer-to-Peer*) na verificação da proteção de arquivos e dados de login. Utilizando o software Hashcat para fazer a validação dessas *hashes*, aplicando as técnicas de criptoanálise Ataque por Força-Bruta e Ataque por Dicionário. Para ambas as técnicas, serão utilizados diferentes dicionários de palavras da língua inglesa e portuguesa que auxiliaram na quebra dessas *hashes*. Mesmo o Hashcat não oferecendo uma alta quantidade hashes recuperadas, pode-se dizer que o software é útil na maioria dos casos, caso seja utilizado com máscaras incrementas de diferentes tamanhos, o software passa a ter resultados excepcionalmente melhores, pois, as chances de quebrar as hashes passam a aumentar.

Palavras Chave: Segurança da Informação. Vulnerabilidades. Criptografia. MD5. Criptoanálise.

ABSTRACT

Encryption is a set of techniques to camouflage readable information from people who cannot access it, this information is camouflaged through a key generated by the encryption algorithm (RSA, MD5, AES among others). This information is called Cifras, the numbers in turn are indecipherable until you have the same encryption algorithm. There are two types of cryptographic keys that can be generated from the algorithm used, symmetric keys and asymmetric keys. In digital security it is always recommended to use the encryption techniques, thus avoiding the loss of valuable information or end up in the wrong hands. The objective of this work is to demonstrate, through comparisons of time and size, the efficiency and safety of the MD5 algorithm. The MD5 has a 128-bit cryptographic distribution function created by the company RSA Data Security Inc., used mainly in software that uses the P2P (Peer-to-Peer) protocol to verify the protection of files and login data. Using Hashcat software to validate these hashes, applying cryptanalysis techniques, Brute Force Attack and Dictionary Attack. For both techniques, different dictionaries of English and Portuguese words will be used to aid in the breaking of these hashes. Even though the Hashcat does not offer a high amount of recovered hashes, it can be said that the software is useful in most cases, if it is used with masks increments of different sizes, the software will have exceptionally better results because, the chances of breaking the hashes begin to increase.

Keywords: Information Security. Vulnerabilities. Cryptography. MD5. Cryptoanalysis.

LISTA DE ABREVIATURAS E SIGLAS

AES - Advanced Encryption Standard

CPP - Código de Processo Penal

DAS - Digital Signature Algorithm

DES - Data Encryption Standard

DOS - Denial of Service

GCHQ - Government Communications Headquarters

IP - Internet Protocol

MD4 - Message-Digest algorithm 4

MD5 - Message-Digest algorithm 5

MITM - Man in The Middle

NIST - National Institute of Standards and Technology

NSA - National Security Agency

RC4 - Alleged RC4

RSA - Ronald Rivest, Adi Shamir e Leonard Adleman

TI - Tecnologia da Informação

XOR - Disjunção Exlusiva

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A criptografia é algo fundamental nos dias de hoje, qualquer coisa que fazemos, seja dentro da internet ou fora, a criptografia está envolvida, é importante todo usuário e programador saber como funciona um algoritmo de encriptação, não em níveis avançados com todas as equações matemáticas, mas pelo menos o básico da criptografia, desde os primórdios que o homem tem sentido a necessidade de guardar os segredos, sejam segredos familiares, segredos sentimentais, segredos pessoais, segredos religiosos, ou segredos militares e governamentais.

Tão forte quanto à necessidade nata de espécie humana de guardar o segredo sobre determinados assuntos é a vontade dos mesmos humanos de desvendar esses segredos. Seja por dinheiro, poder, vingança, curiosidade, arrogância, ou qualquer outro sentimento essa tem sido uma batalha que, ao longo dos anos vem sendo travada entre aqueles que querem guardar segredos e os que querem desvendar esses segredos. (FRANÇA, 2015).

A cifração de mensagens foi, aos poucos, se tornando um processo cada vez mais sofisticado, passando pelas maquinas Enigma, usadas pelo exército alemão na Segunda Guerra Mundial, até aos nossos dias com as transações eletrônicas na Internet (MACHIAVELO, 2016).

Na atual sociedade da informação, em que cada vez mais as pessoas se comunicam através da internet, um meio de interação muito exposto, a importância da criptografia é enorme, só através da cifração de mensagens dessas comunicações é que podemos garantir a confidencialidade da informação que queremos transmitir. (LOPES; QUARESMA, 2016).

Atualmente, a criptoanálise é utilizada a fim de se consultar as tentativas de quebra de segurança de outros tipos de algoritmos e de protocolos criptográficos. Em geral, a quebra ou então, a publicação de uma criptografia é um ato ilegal, previsto em lei.

Entretanto, a criptoanálise exclui geralmente os ataques que não alvejam primeiramente fraquezas na criptografia real. Os métodos, tais como o suborno, coerção física, furto de dados, *keylogger*, embora esse tipo de ataque advém

pela necessidade da segurança computacional, estão gradativamente tornando-se menos eficazes em relação a criptoanálise tradicional.

Mediante aos fatos ocorridos nos últimos anos de ataques DoS, invasões, fraudes entre outras, justifica-se uma pesquisa de um trabalho na área de criptografia, a fim de demonstrar a importância da tal para poder amenizar esses tipos de acontecimentos que vem aumentando a cada ano.

2 OBJETIVOS

Nos próximos tópicos foram apresentados o objetivo geral e os objetivos específicos.

2.1 OBJETIVO GERAL

Utilizar o software Hashcat para validar as técnicas de ataque por força bruta e dicionário, assim como, testar a segurança e o tempo de decifração do algoritmo MD5.

2.2 OBJETIVOS ESPECÍFICOS

a) Efetuar uma análise bibliográfica sobre os temas abordados nessa pesquisa;
b) Mostrar na prática as técnicas de ataque por força bruta e dicionário da criptoanálise;
c) Utilizar o software Hashcat para os testes de segurança entre as hashs geradas;
d) Realizar uma análise comparativa entre as técnicas de criptoanálise: Ataque por Força Bruta e Ataque por Dicionário.

3 REFERENCIAL TEÓRICO

Nessa seção serão discutidos os referenciais teóricos sobre os assuntos de segurança, perícia forense, criptografia, tipos de criptografias, criptoanálise e técnicas de criptoanálise. A seguir, serão apresentadas uma breve explicação da importância na utilização da segurança no meio digital.

3.1 SEGURANÇA DIGITAL

A dependência cada vez maior da tecnologia de informação (TI) torna o software seguro um elemento chave para a continuidade dos serviços de nossa sociedade atual. Nos últimos anos, instituições públicas e privadas aumentaram seus investimentos em segurança da informação, mas a quantidade de ataques vem crescendo mais rapidamente do que a nossa capacidade de poder os enfrentar, colocando em risco a propriedade intelectual, a relação de confiança de clientes e a operação de serviços e negócios apoiados pelos serviços de TI (BATISTA, 2016).

Garantir a segurança da informação em ambiente digital constitui, cada vez mais, uma preocupação à escala mundial, seja por parte de organismos públicos, privados, universidades, empresas e até pelos cidadãos, de forma individual ou coletiva. Na realidade, os riscos e ameaças não conhecem fronteiras de natureza geográfica, linguística, política ou qualquer outro tipo de barreiras.

O que se verifica é que da mesma forma que aumenta a quantidade de informação em formato digital disponível, nomeadamente na Internet, também se verifica um aumento contínuo das ameaças e dos ataques à segurança da informação digital, levando também a um crescimento das estratégias de promoção da segurança e redução do risco. (PEREIRA, 2005).

O uso cada vez amplo e disseminado de sistemas informatizados para a realização das mais diversas atividades, com a integração destes sistemas e de suas bases de dados por meio de redes, é um fato determinante da sociedade da informação. Contudo, este universo de conteúdos e continentes digitais está sujeito a várias formas de ameaças, físicas ou virtuais, que comprometem

seriamente a segurança das pessoas e das informações a elas atinentes, bem como das transações que envolvem o complexo usuário-sistema-informação.

A tecnologia da informação é capaz de apresentar parte da solução a este problema, não sendo, contudo, capaz de resolvê-lo integralmente, e até mesmo contribuindo, em alguns casos, para agravá-lo. Nos ambientes organizacionais, a prática voltada à preservação da segurança é orientada pelas chamadas políticas de segurança da informação, que devem abranger de forma adequada as mais variadas áreas do contexto organizacional, perpassando os recursos computacionais e de infraestrutura e logística, além dos recursos humanos. (MARCIANO, 2016).

Embora a segurança digital seja de extrema importância, tal objetivo só pode ser atingido com a utilização da criptografia.

3.2 CRIPTOGRAFIA

Segundo Schneier (1996), o processo de transformar a mensagem de texto puro em uma mensagem com texto de conteúdo oculto é chamado de Encriptar ou Cifrar. A mensagem encriptada é denominada *ciphertext* ou criptografada. O processo de tornar a mensagem encriptada em texto puro novamente é denominado decriptografar ou decifrar. (LARROSA, 2016).

O surgimento da criptografia (do Grego: kryptos, oculto + graph, r. de graphein, escrever) deve ter sido quase que simultâneo com o da escrita. Os Espartanos, em 400 a.C., desenvolveram um sistema muito curioso, num bastão enrolava-se uma tira de couro, após isso escrevia-se a mensagem em uma tira de couro, o ato de desenrolar a tira do bastão cifrava a mensagem, a qual só podia ser decifrada tornando a enrolar a tira num bastão de diâmetro semelhante.

Em contraponto com este método puramente mecânico a cifra de Júlio César implicava um algoritmo de cifração. Um sistema criptográfico é então um conjunto de técnicas que nos permitem tornar incompreensível uma dada mensagem, de modo que só o verdadeiro destinatário da mesma a consiga decifrar, obtendo dessa forma o texto original. (LOPES; QUARESMA, 2016).

Desenvolvida por Arthur Scherbius em 1918, a Enigma levantou um grande interesse por parte da marinha de guerra alemã em 1926, quando passou a ser usado como seu principal meio de comunicação e ficaram conhecidas

como Funkschlüssel C. Em 1928, o exército elaborou sua própria versão, a Enigma G, e passou a ser usado por todo o exército alemão, tendo suas chaves trocadas mensalmente. (CASTELLÓ; VAZ, 2016)

A máquina era elétrico-mecânica e funcionava com rotores (primeiramente com 3 e depois com até 8 rotores). Ao pressionar uma tecla, o rotor mais da esquerda avançava uma posição, o que ocasionava a rotação dos outros rotores da direita. Esse movimento contínuo dos rotores ocasionava em diferentes combinações na encriptação. (CASTELLÓ; VAZ, 2016)

A codificação de uma mensagem criptografada pela Enigma era considerada impossível na época (já que para tal, seria necessária uma alta força bruta computacional). A título de curiosidade, os aliados só conseguiram decifrar os códigos da Enigma graças ao roubo de uma dessas máquinas, e que com graças à engenharia reversa, foram construídas máquinas capazes de ler e codificar os códigos alemães, os Colossus. A Enigma acabou por gerar diversos descendentes, tais como a Typex, a SIGABA e a M-134-C, que apesar de serem semelhantes à Enigma em seus princípios básicos, eram muito mais seguras. (CASTELLÓ; VAZ, 2016)

A criptografia passou a ser usada em larga escala por todas as ações, principalmente em épocas de guerra, tal como durante a Guerra Fria, onde Estados Unidos e União Soviética usaram esses métodos a fim de esconder do inimigo suas ações e movimentações, criptografando-as e impedindo que outros que não possuíssem a chave pudessem ler, forçando-os a usar diversos métodos para quebrar os códigos de criptografia. (CASTELLÓ; VAZ, 2016)

Depois disso surgiram diversos tipos de criptografia, tais como a pôr chave simétrica, por chave assimétrica e por hash. Pode-se ainda citar a criptografia quântica, que ainda está em fase de testes (CASTELLÓ; VAZ, 2016).

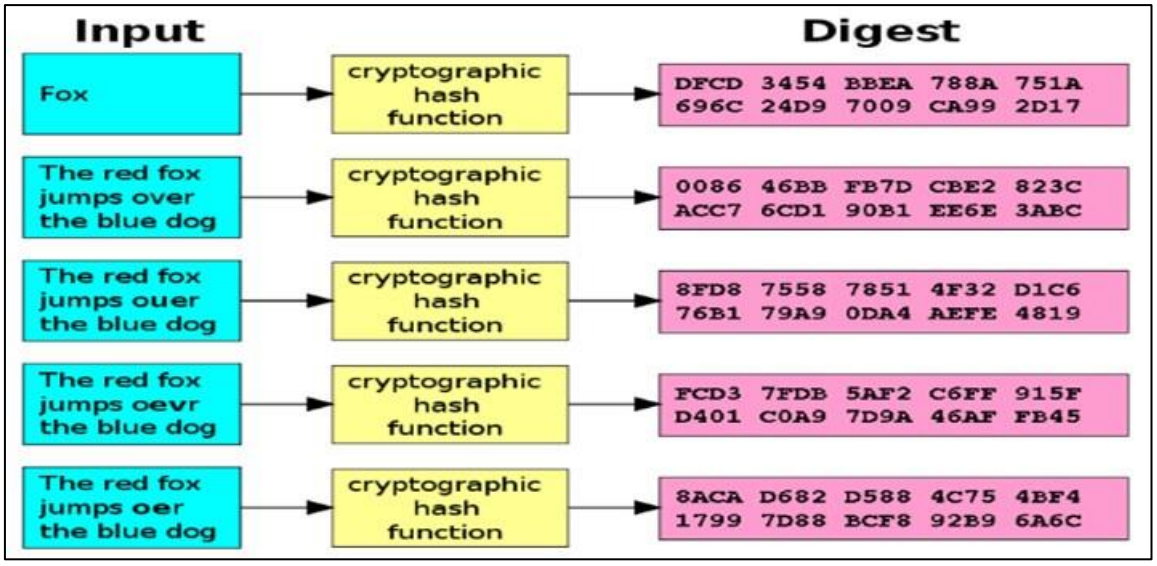
Para podermos ter um começo de entendimento sobre a criptografia, podemos ter como base o algoritmo MD5. Este algoritmo por mais antigo e simples que seja, tem uma relevância grande nos dias atuais, varios sistemas ainda utilizam o MD5 para encriptar seus dados.

3.2.1 MD5

O algoritmo MD5 é uma extensão do algoritmo MD4, sendo parcialmente mais lento que o MD4, porém é conservador no seu design. O MD5 foi projetado após considerar que seu antecessor pudesse ser adotado com a finalidade de uso ativo. O MD4 foi desenvolvido a fim de ser excepcionalmente mais veloz, no entanto, tornou-se incerto os ataques criptoanáliticos. Por sua vez, o MD5 utiliza pouca velocidade, mas com maior segurança, incorpora-se algumas sugestões feitas por vários usuários e contém otimizações adicionais (RIVEST, 1992).

Como pode ser visto na Figura 1, é demonstrado o funcionamento do algoritmo MD5.

Figura 1 - Exemplo do algoritmo MD5



Fonte: BROOKE (2016).

O algoritmo MD5 com a mensagem de entrada de um tamanho variável para uma saída que contém um comprimento fixo de 128bits. Para isso, a mensagem é dividida em blocos de 512bits, equivalente a dezesseis palavras de 32bits, dessa forma o texto é preenchido de um modo que seu comprimento total seja divisível em 512bits.

O preenchimento da mensagem é dado por um único bit que em primeiro momento é anexado ao final da mensagem. Logo em seguida é adicionado

tantos zeros que serão necessários para levar o comprimento da mensagem até 64bits, dessa forma, representando a mensagem original.

Principalmente, o algoritmo MD5 opera em um estado de 128bits, divido em quatro palavras de 32bits, que são retratadas como A, B, C e D. Estas palavras têm sua inicialização para suas determinadas constantes que são fixas. Em seguida, o algoritmo utiliza cada bloco de mensagem para alterar o seu estado.

O processamento do bloco de mensagens consiste em quatro estágios parecidos, que tem o nome de rodadas. Cada rodada é composta por dezesseis operações baseadas em uma função não-linear denominada F, adição modular e rotação a esquerda.

A criptografia possui também outro algoritmo bem conhecido e vastamente utilizado nas aplicações, o RSA.

3.2.2 RSA

Criado em 1977 por Ron Rivest, Adi Shamir e Len Adleman, é um dos algoritmos de chave assimétrica mais utilizados. Seu funcionamento consiste na multiplicação de 2 números primos muito grandes para a geração de um terceiro número. Para quebrar essa criptografia, seria necessário a fatoração desse número para encontrar os 2 números primos que o geraram, porém, para isso é necessário um poder muito alto de processamento, o que acaba inviabilizando a tarefa. A chave privada são os dois números primos e a pública é o terceiro número (CASTELLÓ; VAZ, 2016).

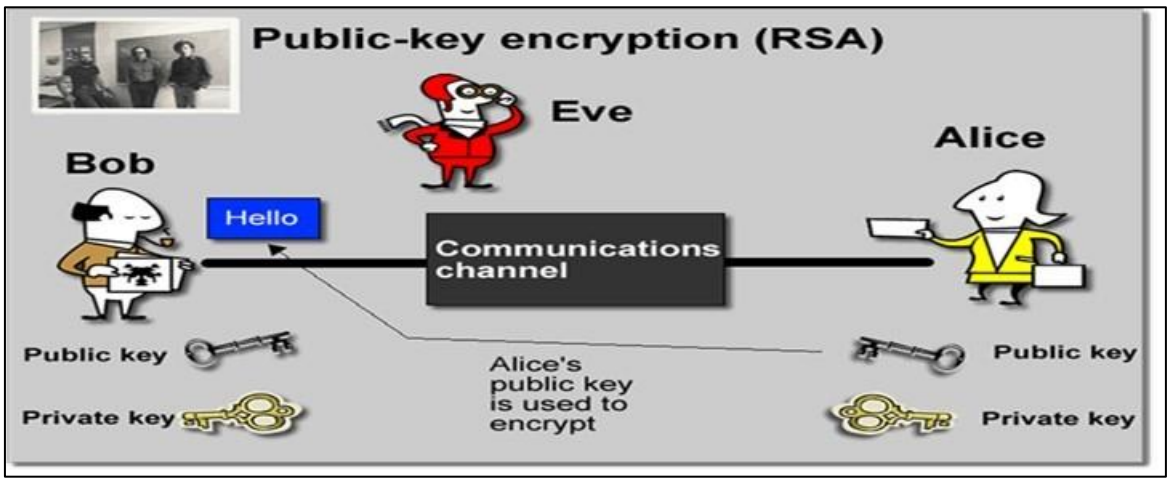
Dentre dos métodos criptográficos de chave pública, o RSA é o mais conhecido e, atualmente, o mais usado, sobre todo em aplicações comerciais. Este sistema de criptografia é muito útil para o comercio eletrônico via Internet, assim como para navegar nela, através do Netscape (SILVA, 2006).

O RSA, por ser um método de chave pública, permite que qualquer usuário codifique mensagens, mas como a chave de decodificação é secreta, só o destinatário legítimo poderá decodificá-la. A impossibilidade de quebrar a chave de decodificação é possível pela não existência de algoritmos eficientes para a fatoração de inteiros em fatores primos, sobre todo, se o número de algarismos é 100 ou maior. O tempo de codificação de uma mensagem é

praticamente desprezível, mas o tempo de decodificação pode tornar o processo inviável. (SILVA, 2016).

Como pode ser visto na Figura 2, é demonstrado o exemplo do funcionamento do algoritmo de encriptação RSA.

Figura 2 - Exemplo do algoritmo RSA



Fonte: BILLATNAPIER (2016).

O algoritmo RSA é composto por um par de chaves, sendo elas, uma chave pública que é conhecida por todos e uma chave privada, que sempre é mantida em sigilo. Toda e qualquer mensagem que for cifrada utilizando uma chave pública só conseguira ser decifrada caso utilize a chave privada equivalente.

Para a geração das chaves públicas e privadas, no algoritmo RSA, pode-se ser feito o seguinte cálculo para determina-las.

a) É escolhido de forma aleatória dois números primos P e Q, da ordem de no mínimo 10^{100}.
b) Então, calcula-se $n = p.q$
c) Calcula a função Totiente de Euler em n: $\varphi(n) = (p-1)(q-1)$
d) Escolhe-se um inteiro e tal que $1 < e < \varphi(n)$, de forma que e e $\varphi(n)$ sejam primos entre si mesmo.
e) Por último, calcula-se d de forma que $de = 1\ (mod\ \varphi(n))$, ou seja, d seja o inverso multiplicativo de e em $(mod\ \varphi(n))$.

No fim, teremos uma chave pública, composta de um par (n, e) e uma chave privada, composta por a tripla (p, q, d).

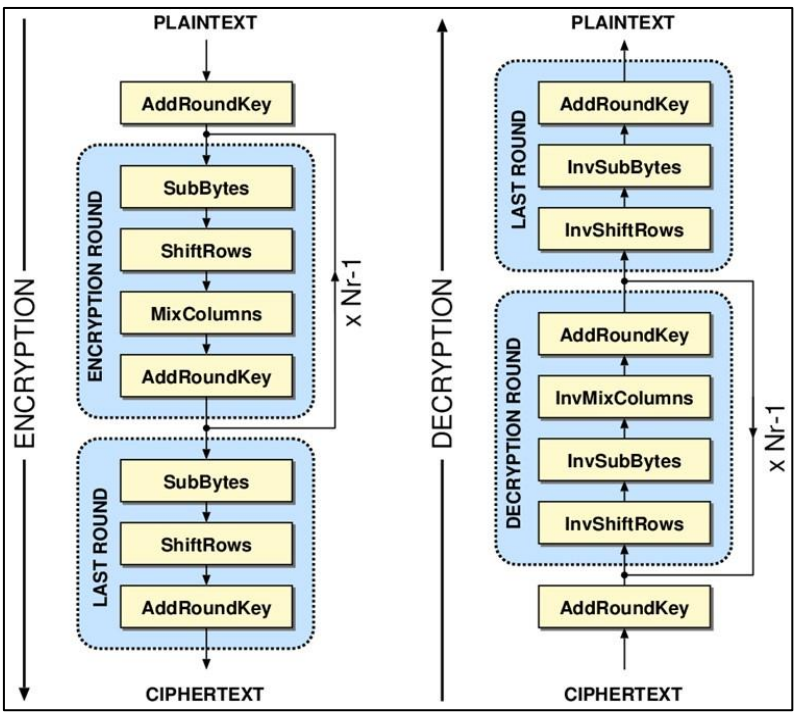
Contudo, vale notar também outro algoritmo importante para a criptografia atual, o AES. Sua utilização é mais ampla que o RSA e, por muitos, é considerado mais seguro.

3.2.3 AES

O Advanced Encryption Standard (AES) é uma cifra de bloco, anunciado pelo National Institute of Standards and Technology (NIST) em 2003, fruto de concurso para escolha de um novo algoritmo de chave simétrica para proteger informações do governo federal, sendo adotado como padrão pelo governo dos Estados Unidos, é um dos algoritmos mais populares, desde 2006, usado para criptografia de chave simétrica, sendo considerado como o padrão substituto do DES. O AES tem um tamanho de bloco fixo em 128 bits e uma chave com tamanho de 128, 192 ou 256 bits, ele é rápido tanto em software quanto em hardware, é relativamente fácil de executar e requer pouca memória. (OLIVEIRA, 2012).

Como pode ser observado na Figura 3, é demonstrado exemplo de funcionamento do algoritmo da criptografia AES.

Figura 3 - Exemplo do algoritmo AES



Fonte: Ebermann (2016).

Para o funcionamento do AES são necessários alguns dados como a S-BOX (tabela de substituição estática), o estado (que é o bloco de dados de entrada sendo modificado pelas transformações do algoritmo), a chave, e a chave de expansão (Announcing The Federal, p. 197).

O AES é separado por dois módulos, sendo um módulo para realização da cifragem e um para a decifragem.

Segundo Trevisa, Sacchi e Sanabria (2013) comentam as transformações necessárias:

a) SubBytes: Transformação que substitui os bytes do estado por bytes da S-box;
b) ShiftRows: rotaciona ciclicamente as linhas do estado, para a segunda em 1 casa, para a terceira em 2 casas e para a quarta, 3 casas;
c) MixColumns: esta operação transforma os dados das colunas do estado multiplicando por um polinômio irredutível fixado;
d) AddRoundKey: “mistura” as colunas com uma das chaves geradas na rotina de expansão.

No módulo de decifragem existem transformações equivalentes contrárias, como a *SubBytes* inversa (que utiliza uma S-Box inversa), a *ShiftRows* inversa, a *MixColumns* inversa e a *AddRoundKey* inversa. Cada uma dessas faz a operação inversa de sua representante no módulo de cifragem (TREVISA; SACCHI; SANABRIA, 2013).

O AES é considerado um dos algoritmos mais seguros e eficazes quando se é tratado sobre segurança digital.

3.3 CRIPTOANÁLISE

A criptoanálise é a ciência de “quebrar” os métodos criptográficos, para que se possa decifrar e ler as mensagens anteriormente criptografadas (TKOTZ, 2005). Dessa forma, a criptoanálise estuda todos os procedimentos que serão necessários para tentar afetar as técnicas criptográficas, sendo assim, obter o

acesso a mensagem previamente encriptada sem o conhecimento da chave utilizada (JASPER, 2009).

A criptoanálise se divide na Criptoanálise Clássica, utilizada nas cifras consideradas clássicas, as cifras de substituição e transposição (ALMEIDA, 2002). Normalmente, as cifras clássicas são examinadas com a técnica de Análise de Frequência, esta técnica é a responsável por analisar o texto cifrado em busca de padrões que possam indicar particularidades com a mensagem original.

A criptoanálise moderna tem um objetivo diferente, que é a descoberta da chave criptográfica utilizada por um determinado algoritmo criptográfico, utilizando a menor quantidade possível de recursos como processamento, memória e número de mensagens claras/cifradas (JUNIOR, 2008). Atualmente, podemos encarar que a criptoanálise moderna é necessária, visto a situação atual das cifras e das que estão por vir, pois estas serão executadas em máquinas muito similares, onde se tornará viável a possibilidade de novos métodos para a criptoanálise.

3.3.1 CRIPTOANÁLISE LINEAR

A técnica de criptoanálise linear foi apresentada em 1993 pelo criptoanalista Mitsuru Matsui e no ano seguinte foi formalizada por Eli Biham 1994, os autores focaram seus estudos na criptoanálise DES. Como fora proposto por Matsui, o propósito da criptoanálise linear é obter uma expressão linear aproximada de um algoritmo criptográfico (DAVI, 2010).

A técnica utilizada na criptoanálise linear consiste em estudar e analisar as relações estatísticas entre os bits correspondentes das mensagens, cifras e da chave criptográfica utilizada. Para alcançar esse objetivo de obter uma expressão linear, Matsui desenvolveu um modelo linear relacionando entradas e saídas das S box do DES. Abaixo, pode-se conferir o passo-a-passo do modelo linear utilizado por Matsui (DAVI, 2010).

Segundo Davi (2010) pode-se ter algumas regras sobre a criptoanálise linear:

a) escolhe-se um subconjunto dos bits da entrada da S-Box analisada, dentre 2^6 valores possíveis e calcula-se a paridade (operação de "ou exclusivo") dos mesmos;
b) escolhe-se um subconjunto dos bits da saída da S box analisada, dentre 2^4 valores possíveis e calcula-se a paridade (operação de "ou exclusivo") dos mesmos;
c) a análise acima é repetida até que todos os subconjuntos de entrada e saída sejam verificados;
d) os valores acima são tabelados de tal forma que as linhas da tabela contenham os subconjuntos possíveis de entrada, enquanto as colunas contêm os subconjuntos possíveis de saída;
e) as entradas desta tabela representaram o número de vezes que para um dado subconjunto de bits de entrada, a paridade do mesmo é igual a paridade do subconjunto de bits de saída correspondente;
f) o ataque por criptoanálise linear será tão mais eficaz quanto o valor da entrada se afastar do valor 32 (metade do número de entradas de uma S box);

Matsui e Biham (1994) adotaram a convenção de subtração de 32 unidades para cada entrada da tabela, fazendo assim que uma entrada que possua valor zero corresponda a entrada 32, como dito anteriormente na descrição da técnica. Matsui, então, através dos seus estudos concluiu que quanto mais afastado do valor zero, maior serão as chances do ataque se tornar bem-sucedido (DAVI, 2010).

Semelhantemente ao estudo da criptoanálise diferencial, as entradas da tabela estão associadas a uma probabilidade, sendo assim, se concluído após uma análise que um dado conjunto analisado tem probabilidade ½ o ataque não funcionará corretamente.

3.3.2 CRIPTOANÁLISE DIFERENCIAL

A criptoanálise diferencial foi de fato apresentada ao público em 1990. Essa técnica da criptoanálise já era conhecida pelos criadores do algoritmo DES, mas a mesma foi mantida em segredo até a década de 90 quando Eli Bahim e Adir Shamir apresentaram essa técnica publicamente

Biham e Shamir (1974) utilizaram um método de ataque que se baseia por mensagens escolhidas, aos quais se toma os pares e a diferença entre os criptogramas é analisada. Como é sugerido por Davi (2010), é necessário definir o termo “diferencial”.

Entende-se por "diferença" a operação de "ou exclusivo" entre dois n-gramas. O uso da palavra "diferença" com a significação enunciada aplica-se, em especial, ao estudo de cifradores de bloco como o DES e similares. (DAVI, 2010)

Os pesquisadores Biham e Shamir (1974) definiram um parâmetro básico para a criptoanálise diferencial, sendo esse a CARACTERÍSTICA. A definição de característica pode ser vista a seguir, entretanto, essa definição é classificada por Biham como “informal”.

Característica: Associado a qualquer par de cifras estão: o valor do "XOR" dos textos em claro correspondentes às cifras; o "XOR" entre as cifras; o "XOR" entre os valores das entradas de cada passo do algoritmo (em duas execuções do mesmo); o "XOR" dos valores das saídas (em duas execuções distintas do algoritmo) de cada passo do algoritmo. Estes valores de "XOR" formam uma *característica de "n iterações"*. A característica possui associada a ela uma probabilidade, a qual é a probabilidade de um par, selecionado ao acaso cujo resultado da operação "XOR" é conhecido, ter especificado na característica os valores "XOR" resultantes dos passos intermediários do algoritmo e das cifras. (DAVI, 2010)

No quadro 1, pode-se ver uma tabela comparativa em um ataque DES utilizando mensagens escolhidas, mensagens conhecidas e mensagens analisadas.

Quadro 1 - Tabela comparativa de um ataque DES, utilizando criptoanálise diferencial

Número de iterações	Mensagens escolhidas	Mensagens conhecidas	Mensagens analisadas	Complexidade da análise
8	2^{14}	2^{38}	4	2^{21}
9	2^{24}	2^{44}	2	2^{32}
10	2^{24}	2^{43}	2^{14}	2^{15}
11	2^{31}	2^{47}	2	2^{32}
12	2^{31}	2^{47}	2^{21}	2^{21}
13	2^{39}	2^{52}	2	2^{32}
14	2^{39}	2^{51}	2^{29}	2^{29}
15	2^{47}	2^{56}	2^{7}	2^{37}
16	2^{47}	2^{55}	2^{36}	2^{37}

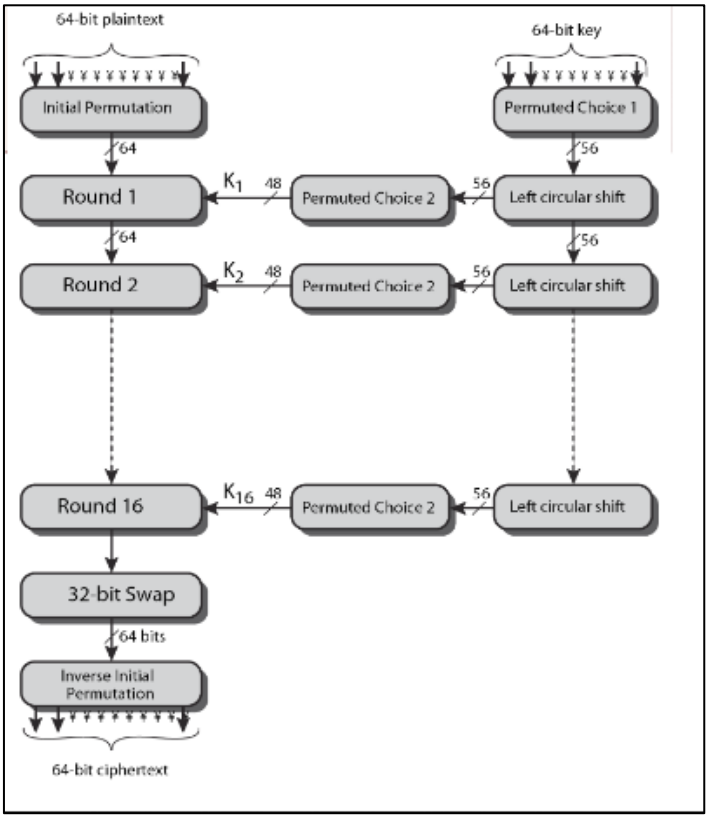
Fonte: Davi (2010)

Como visto no quadro 1, o melhor ataque contra o DES, de dezesseis iterações, requer exatamente 2^{47} iterações de mensagens escolhidas, entretanto, é possível converter este ataque para o de mensagens conhecidas, porém, o mesmo irá utilizar 2^{55} iterações.

A resistência do cifrador de blocos pode ser melhorada aumentando o número de iterações realizadas, utilizando o exemplo da tabela do DES acima: caso uma variante do DES seja utilizada, com dezessete ou dezoito iterações, o ataque terá o mesmo peso que o de um ataque por força bruta. Entretanto, se é utilizado dezenove iterações ou mais, o ataque por criptoanálise diferencial acaba se tornando impraticável, pois, o número de operações necessárias acaba se tornando maior do que o número de mensagens possíveis 2^{64}.

Utilizando a técnica de criptoanálise diferencial, pode-se realizar ataques contra algoritmos tais como DES, que possuem uma S-Box constante. No caso do DES, tal ataque possui complexidade equivalente independentemente do modo de operação do algoritmo (ECB, CBC, CFB ou OFB).

Figura 4 - Diagrama de funcionamento do DES



Fonte: STALLINGS, W. (s.d)

Como visto na figura 4, existem dezesseis etapas idênticas no funcionamento do algoritmo DES, estas etapas são denominadas rounds. Além dos rounds, existem duas permutações no DES, denominadas “IP” e “IIP”, estas permutações são inversas uma das outras. A permutação IP “desfaz” a ação realizada pela IIP, e vice-versa.

Antes da realização dos rounds, o bloco é dividido em duas metades de 32bits e tem um processamento alternado, este cruzamento é denominado como “esquema de Feistel”. Neste cruzamento, Feistel garante que o processo de criptografia e descriptografia sejam similares, a diferença é que na descriptografia as subchaves são aplicadas de formas diferentes.

Como qualquer outro cifrador de bloco, o DES sozinho não é considerado um meio seguro de criptografar os dados, é recomendado que ele seja utilizado em um modo de operação.

Em geral, o funcionamento do DES se dá na seguinte maneira:

a) Uma substituição fixa, chamada de permutação inicial, 64bits para cada substituição;
b) Uma transformação, que depende de uma chave de 48bits, reservando a outra metade;
c) Uma troca de 32bits entre cada metade;
d) Ocorre a repetição dos primeiro e segundo passo por dezesseis vezes;
e) Realiza a inversão da permutação inicial.

3.4 FORMAS DE ATAQUE NA CRIPTOANÁLISE

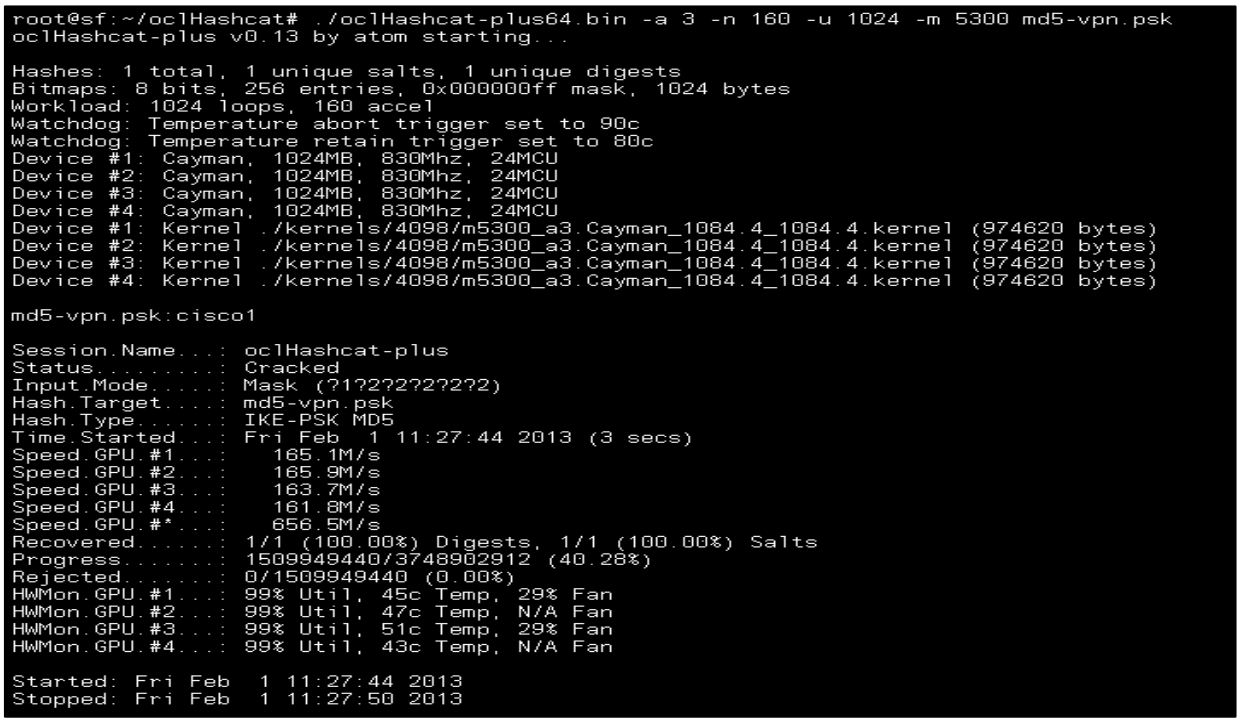
A criptoanálise possui diferentes formas de ataques, sendo alguns deles mais populares que outros. Nesse tópico citaremos alguns que são mais utilizados entre os criptoanalistas.

3.4.1 FORÇA BRUTA (BRUTE FORCE)

Segundo o artigo do site Learn Cryptography (2017), pode-se definir o ataque Brute-Force como: “Qualquer tipo de ataque que envolva a tentativa de cada possível combinação de caracteres ou dados, a fim de encontrar a chave e descriptografar uma mensagem criptografada”.

Sendo assim, os ataques do tipo Força Bruta, geralmente são utilizados como último recurso em um ataque de criptoanálise, visto que esse ataque envolve uma quantidade enorme de tentativas para obter a mensagem encriptada, conforme ilustra a figura 5.

Figura 5 - Software Hashcat realizando um ataque de Força Bruta em uma hash MD5



Fonte: Hashcat (2013)

Quando um criptoanalista está realizando um ataque do tipo força bruta, ele já tem em mente que este tipo de ataque nem sempre será eficiente, tendo em mente que para isso ocorrer temos uma senha segura, porém se o atacante é capaz de adivinhar todas as senhas possíveis, uma delas estará correta e então ele terá acesso ao sistema.

Um método bem comum para impedir um ataque de força bruta ao sistema seria o de utilização de Captchas. Os Captchas, geralmente, são verificações textuais ou por imagens que um usuário deve satisfazer antes do sistema enviar as informações de login para o servidor, conforme apresentado nas Figuras 6 e 7.

Figura 6 - Captcha utilizado para previnir um ataque de Força Bruta em uma página de login.

Fonte: Google. (s.d)

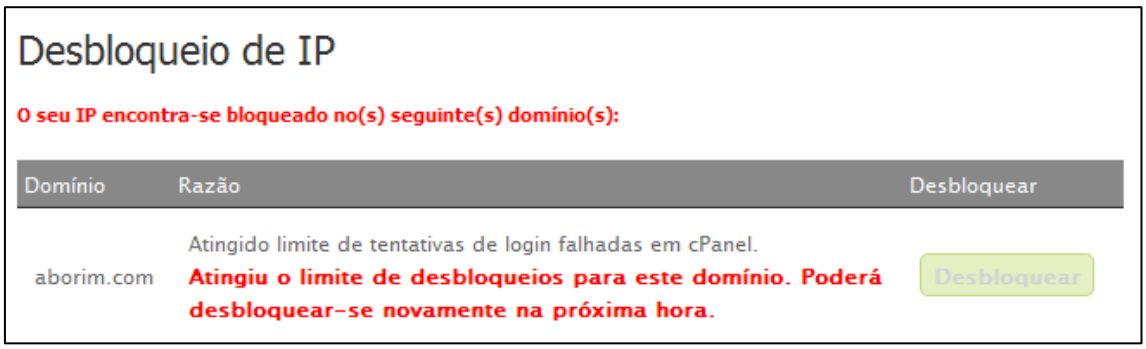
Figura 7 - Captcha textual para prevenção de um ataque Força Bruta.

Fonte: Captcha.net (s.d)

Ao utilizar um sistema de Captcha, o administrador do sistema estará reduzindo drasticamente as chances de um ataque por força bruta, pois assim, o Captcha estaria impedindo que scripts automatizados fiquem enviando requisições ao servidor realizando o ataque.

Outro método bem comum e que vale ser citado é o de tempo limite de tentativa de login. Esse método, por mais simples que seja, é bem eficaz contra os scripts de força bruta. Após uma quantia definida de tentativas de login inválidas (como por exemplo; digitar a senha errada), o servidor automaticamente recusa a requisição de acesso desse endereço de IP por um período de tempo definido, conforme apresentado na Figura 8.

Figura 8 - IP bloqueado após uma tentativa de ataque por Força Bruta



Fonte: Webtuga (s.d)

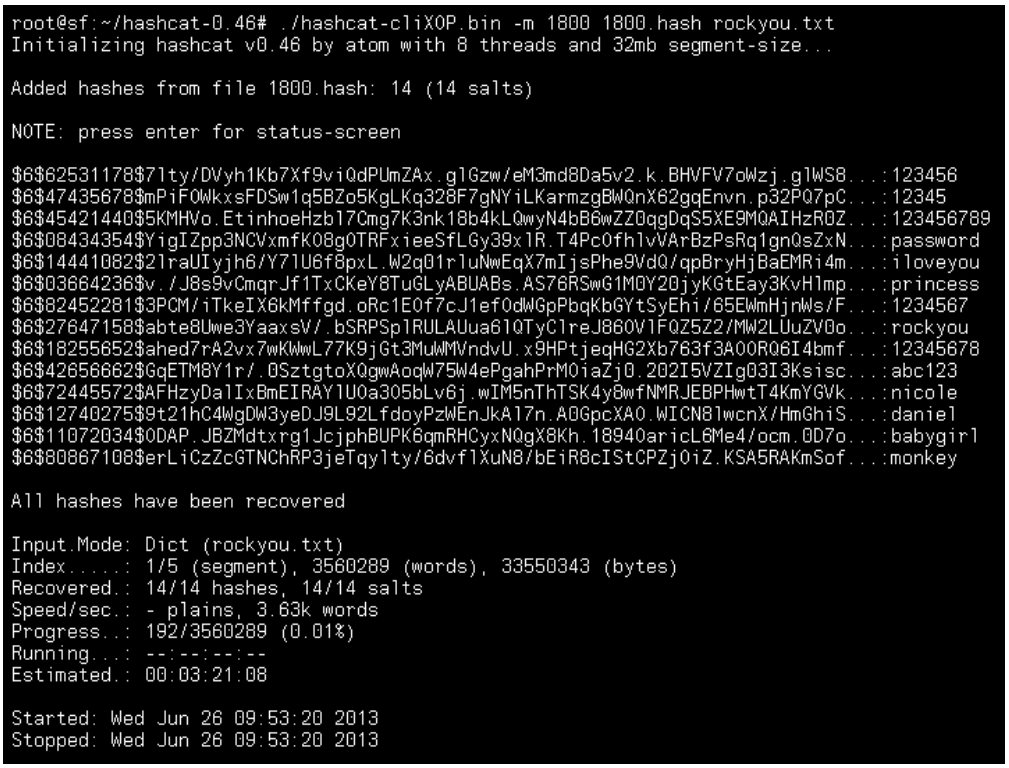
Como visto na figura 8, após realizar um número significante de tentativas de login no domínio especificado, o IP do atacante foi bloqueado, sendo assim, impedindo a tentativa de ataque por força bruta.

3.4.2 ATAQUE POR DICIONÁRIO (DICTIONARY ATTACK)

Tendo como definição o site Learn Cryptography (2017), um ataque por dicionário é um ataque que tenta adivinhar a chave de um texto cifrado, tentanto várias senhas comuns e possíveis, que são susceptíveis a serem usadas pelos indivíduos. Mas comumente esse ataque utiliza um dicionário de palavras inglesas, pode ser empregado qualquer outro dicionário de outra língua, frases e senhas prontas para adivinhar a chave de encriptação.

Novamente, segundo o Learn Cryptography (2017), um ataque por dicionário é mais eficiente que um ataque por força bruta, pois o ataque por dicionário não precisa tentar cada combinação existente de letras e número, ele apenas tenta as palavras, senhas e frases que estão armazenadas geralmente em um arquivo de texto. Porém, há uma desvantagem quando se utiliza o ataque por dicionário, se a chave que o atacante está tentando quebrar não estiver contida dentro do dicionário, ele nunca irá encontra-la com êxito, de acordo com a Figura 9.

Figura 9 - Software Hashcat executando um ataque de dicionário em várias hashes indefinidas



Fonte: Hashcat (2013)

Pode-se observar na figura 9, que o software Hashcat realizou um ataque de força bruta nas hashes criptografas com senhas aleatórias de alguns usuários, neste caso em específico, foi utilizado o dicionário de palavras *rockyou.txt* para auxiliar na quebra destas hashes.

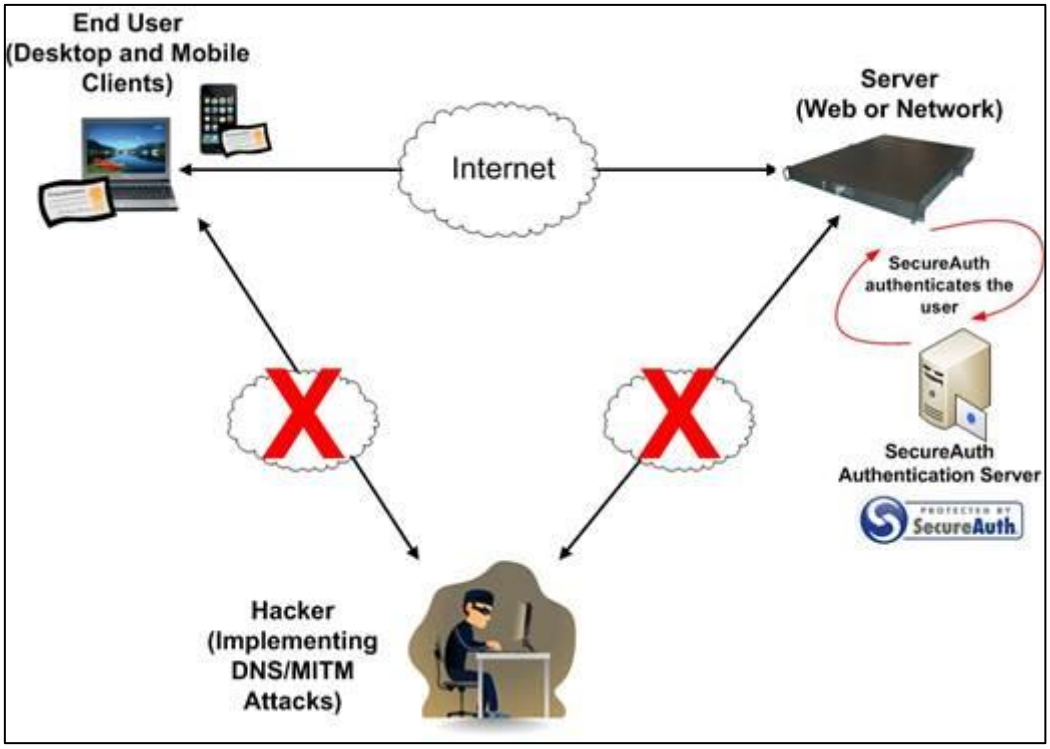
3.4.3 ATAQUE HOMEM NO MEIO (MAN IN THE MIDDLE)

O Ataque Homem no Meio (Man in The Middle), tem uma premissa bem simples, para esse ataque poder surtir efeito, o atacante se posiciona entre as duas partes que estão tentando se comunicar, após isso, todos os dados que o emissor está transmitindo são interceptados pelo atacante no meio da comunicação.

Ao que diz respeito à os clientes da conexão, nada de diferente é percebido por ambos, no entanto, o atacante tem controle pleno sob as

informações que estão sendo transmitidas pelos clientes, podendo até mesmo manipular as informações transmitidas, de acordo com a Figura 10.

Figura 10 - Processo de execução de um ataque Man In The Middle



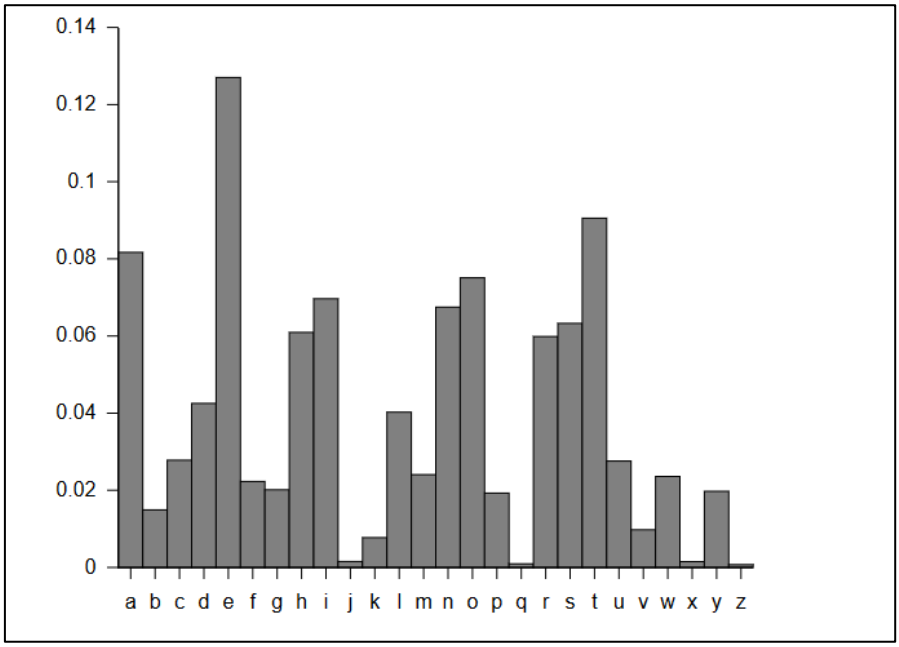
Fonte: Lancenewman (2014)

Como visto na figura 10, o atacante se posiciona no meio, entre a comunicação do usuário com o servidor. Desta forma, o atacante pode manipular as informações que estão sendo recebidas pelo servidor, tanto quanto as informações que estão sendo enviadas para o servidor.

3.4.4 ATAQUE ANÁLISE DE FREQUÊNCIA (FREQUENCY ANALYSIS)

O método de ataque da Análise de Frequência, consiste em estudar um grupo seleto de letras contidas em um texto cifrado, numa tentativa de revelar parcialmente a mensagem cifrada. Utilizando como referência a língua inglesa, temos certos grupos e letras que aparecem em uma frequência incomum, conforme a Figura 11.

Figura 11 - Dados de um ataque por Frequência de Análise.



Fonte: Learn Cryptography (2017)

Entende-se que as frequências utilizadas na língua inglesa para a comunicação, ao utilizarmos um método de criptografia para desmascarar essas frequências e não obtivermos sucesso, é possível determinar estatisticamente partes do texto cifrado sozinho.

3.5 PERÍCIA FORENSE COMPUTACIONAL

Nessa seção serão apresentadas as definições sobre a perícia forense computacional.

3.5.1 DEFINIÇÃO DE PERÍCIA FORENSE COMPUTACIONAL

“A inovação tecnológica traz uma série de benefícios para as pessoas e a comunidade em geral. Todavia, com as vantagens, surge também a possibilidade de realização de novas práticas ilegais e criminosas” (ALMEIDA, 2011, p. 10).

Como citado previamente por Almeida (2011), a inovação que a tecnologia proporciona é benéfica para todas as pessoas que desfrutam da

mesma, porém por conta dessa mudança surgem também, novas práticas transgressoras no meio digital.

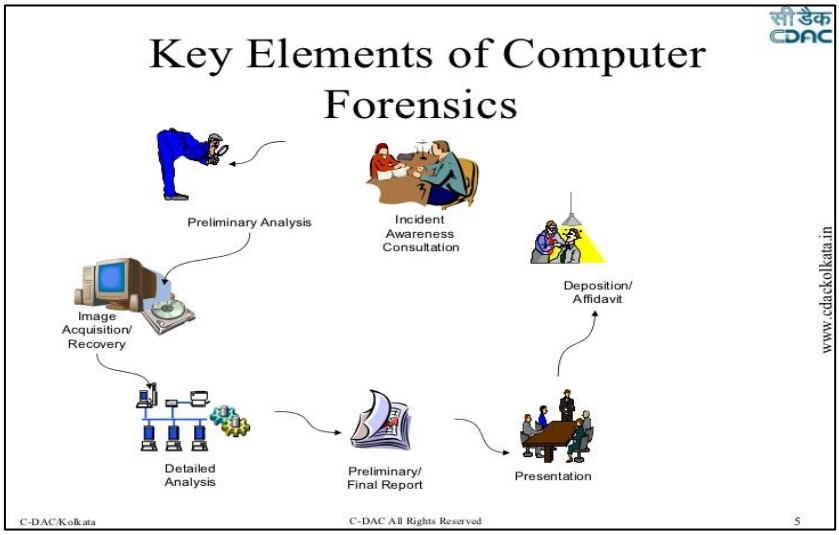
Conforme é determinado no Código de Processo Penal Brasileiro (CPP) no artigo 158: “Quando a infração deixar vestígios, será indispensável o exame de corpo de delito, direto ou indireto, não podendo supri-lo a confissão do acusado.” Traz essa definição do CPP para a área de computação, podemos entender que estes vestígios são digitais, uma vez que todas informações armazenadas em computadores são compostas por uma ordem lógica de bits.

Agora se olharmos diretamente no artigo 159, o CPP estabelece que “O exame de corpo de delito e outras perícias serão realizados por perito oficial, portador de diploma de curso superior.” Pode-se então, reconhecer que Forense Computacional é a prática pertencente aos exames realizados por especialistas e profissionais legalmente habilitados.

3.5.2 ETAPAS DE UM PROCEDIMENTO DE FORENSE COMPUTACIONAL

Ainda citando Almeia (2011), após finalizar os cumprimentos de um mandato de busca e apreensão que tenha como resultado a coleta de equipamentos computacionais, conforme fora descrito no capítulo anterior, é necessário que encaminhe o material para um laboratório de informática capacitado, a fim de, realizar os exames necessários, de acordo com a Figura 12.

Figura 12 - Processo de uma Perícia Forense Computacional



Fonte: C-DAC/Kolkata (s.d)

Almeida (2011) também descreveu alguns dos passos necessários que um perito deve seguir, a fim de, realizar sua investigação.

a) Preservação: Diferentemente do que ocorre durante a busca e apreensão de equipamentos, onde há um foco maior na integridade física deles, esta fase tem o objetivo de garantir que as informações armazenadas no material questionado jamais sejam alteradas durante toda a investigação e processo.
b) Coleta de dados: Esta etapa baseia-se na execução de um conjunto de procedimentos para recuperar e catalogar todos os dados contidos na mídia investigada, estando eles ocultos ou explícitos. A partir de sua conclusão, será possível realizar buscas rápidas por palavras-chaves no conteúdo dos dispositivos armazenados.
c) Análise: A análise dos dados consiste no exame das informações extraídas do material questionado durante a fase anterior, a fim de identificar evidências digitais que tenham relação com o delito investigado.
d) Formalização: É a etapa final dos exames forenses e é formada pela elaboração do laudo pelo perito, apontando o resultado e apresentando as evidências digitais encontradas nos materiais examinados.

4 TRABALHOS CORRELATOS

No campo da criptoanálise e da criptografia, podem ser encontrados diversos trabalhos científicos, teses e artigos. Após feito um estudo e uma comparação dentre todos os trabalhos encontrados, relacionados a este tema, os artigos que serão citados a seguir, possuem a maior relevância técnica e científica para o desenvolvimento deste trabalho.

Em relação a área da criptoanálise, Candia e Pasin (2005), trazem em seu artigo uma demonstração da Criptoanálise distribuída de alto desempenho comparando o número de iterações das implementações referenciadas (DES e MD5) utilizando três processadores diferentes para as comparações e duas linguagens de programação diferentes.

Além disto, ainda na área da criptoanálise, Rinaldi (2012), traz em seu trabalho uma análise do algoritmo AES e sua criptoanálise diferencial. Em seu

trabalho, Rinaldi, demonstra as noções básicas de como o algoritmo AES funciona e as técnicas utilizadas para realizar a criptoanálise diferencial no algoritmo AES.

No contexto da segurança digital, Oliveira (2012), traz em seu artigo, a importância da criptografia para o mundo atual, visto que, temos uma grande gama de informações sendo transmitidas através da internet, tais como, dados pessoais, comerciais, bancários, entre outros… Oliveira, também comenta sobre a importância do profissional de Tecnologia da Informação, saber como proteger e garantir a privacidade das transações em meios eletrônicos. Para Oliveira, é fundamental que todo profissional saiba como utilizar estes algoritmos, técnicas, protocolos e como estes cuidam dos dados para serem mantidos em segurança.

Ainda na área de segurança digital, Marciano e Marques (2006), citam em seu artigo, “O enfoque social da segurança da informação”, que o universo de conteúdo que nos encontramos atualmente está sujeito a diversas ameaças que podem comprometer seriamente o “usuário-sistema-informação”. Contudo, a tecnologia da informação não consegue resolver todos os problemas por conta própria, ela apenas apresenta parte da solução desejável, ainda assim, temos políticas de segurança que por sua vez, deve-se adequar ao equilíbrio das particularidades humanas e técnicas da segurança da informação.

Através da pesquisa realizada, é possível encontrar diversos artigos e trabalhos que estejam relacionados a área da criptoanálise e criptografia. Com isso, pode-se evidenciar que: não só a criptografia precisa de uma evolução constante para manter nossos dados seguros, mas também, a criptoanálise necessita desta mesma evolução constante, dessa forma, conseguimos criar meios alternativos para experenciar os algoritmos e criar uma proteção mais efetiva de dados. Sendo assim, este trabalho tem o intuito e finalidade de demonstrar duas técnicas de ataques na criptoanálise.

5 METODOLOGIA

O trabalho desenvolvido, em primeiro momento, teve por finalidade ser uma pesquisa teórica, com a realização de estudos sobre o tema de segurança da informação, criptografia, criptoanálise e forense computacional, enfatizando o uso das técnicas de ataques na criptoanálise.

Para o desenvolvimento deste trabalho, foi dividido em três partes distintas:

a) Foi realizada uma pesquisa bibliográfica envolvendo todo o aspecto teórico da segurança da informação, criptografia, criptoanálise e forense computacional, tendo o foco principal nas técnicas de ataque na criptoanálise.
b) Aplicação de um ataque real de criptoanálise, utilizando o software Hashcat para quebrar a segurança do algoritmo MD5, para exemplificar a teoria.
c) Apresentação dos resultados obtidos, complementando o estudo a aplicação prática proposta.

Na primeira fase deste trabalho, foi realizada uma pesquisa sobre os temas relacionados e um levantamento de informações a respeito de segurança da informação, criptografia, criptoanálise e forense computacional. Também, foram analisados os procedimentos necessários, técnicas e conceitos teóricos para a realização de dois ataques bem comuns na criptoanálise, ataque por força bruta e ataque por dicionário. Em segundo momento, foram analisados os procedimentos necessários que são realizados por um perito forense computacional.

Na segunda fase, foi utilizado um software para validar as técnicas de ataque da criptoanálise, esse software tem por sua finalidade a recuperação de hashes criptografas em diferentes algoritmos, tais como: MD5, AES, RSA, SHA, entre outras. Para atingir tal objetivo, foi utilizado um computador com processador Intel Core i5, com 8GB de memória RAM, placa de vídeo AMD Radeon HD 6500 com 2GB de memória RAM e sistema operacional ArchLinux 64bits.

O software Hashcat foi escolhido para realizar a validação destas hashes criptografadas, pois, ele é o software mais avançado em recuperação de hashes de vários algoritmos diferentes. O Hashcat conta os seguintes tipos de ataque

a) Straight – Este ataque é mais popularmente conhecido como ataque por dicionário, ele utiliza uma lista de palavras para tentar “adivinhar” a hash criptografada.
b) Combination – Esta técnica utiliza dois dicionários para realizar o ataque, para cada dicionário ele junta as palavras contidas em ambos para criar uma nova palavra.
c) Brute-force – Esta técnica, como já foi discutido no trabalho, tenta todas as combinações possíveis para “adivinhar” a palavra correspondente a hash
d) Hybrid – Este ataque combina o Brute-force com o dictionary attack, além disso, ele utiliza uma máscara que auxilia na geração de palavras para “adivinhar” a hash.

O modo de ataque escolhido para fazer a validação destas hashes foi o ataque híbrido, ou ataque combinatório, fornecido pelo software Hashcat. Este tipo de ataque junta as técnicas de ataque por força bruta e ataque por dicionário. Enquanto esta técnica utiliza o dicionário de palavras do ataque por dicionário, ela tenta “concatenar” o ataque por força bruta no meio da lista de palavras, por isso é chamado de ataque híbrido.

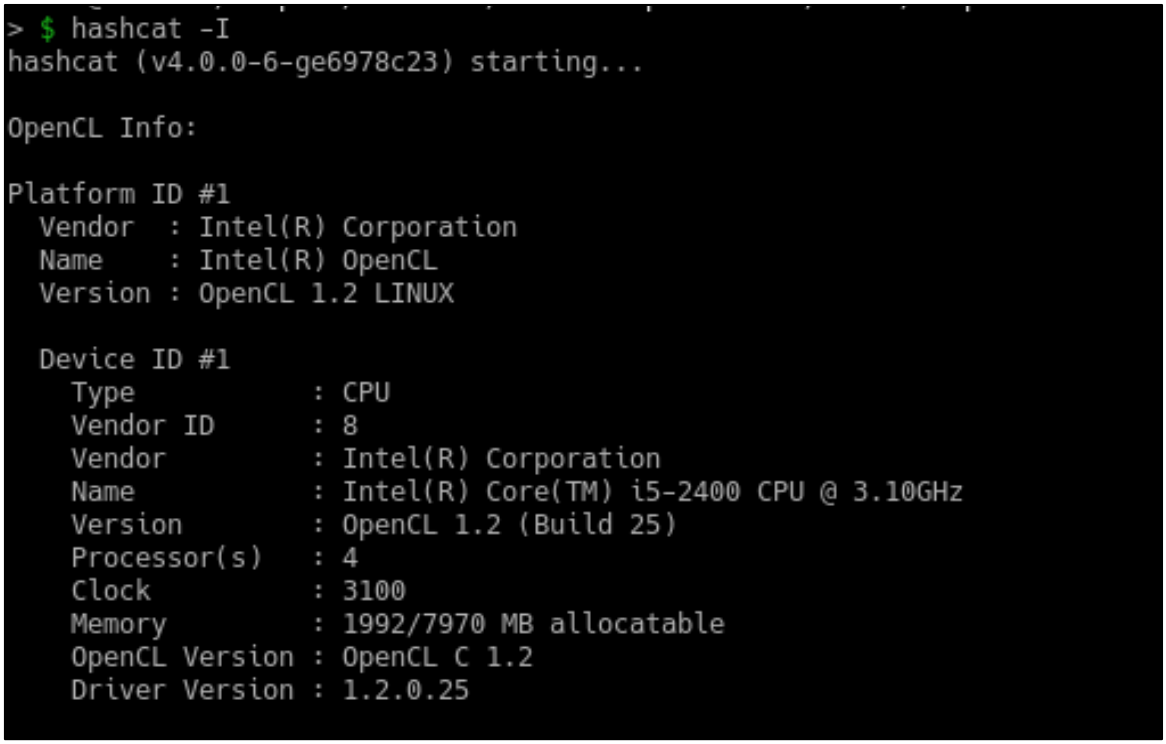
Para realizar o ataque híbrido no Hashcat, é necessário especificar o seguinte comando: “hashcat -t 32 -a 7 hashes.hash "?a?a?a?a" hashcat.dict“

a) “hashcat” – Este comando siginifca que estamos invocando o programa do Hashcat
b) “-t 32” – Está opção é utilizada para dizer ao Hashcat quando parar de aceitar novas correntes de markov, no caso o limite é 32.
c) “-a 7” – Esta opção indica para o Hashcat que ele deve utilizar o modelo de ataque para MD5
d) “hashes.hash” - Aqui definimos para o Hashcat qual é o arquivo que contém as hashes para serem recuperadas
e) “?a?a?a?a” – Este comando define uma máscara para o Hashcat usar durante o ataque híbrido, neste caso o “?a” significa uma

junção de caracteres, maiúsculos e minúsculos e números. O Hashcat irá adicionar esses caracteres extras juntamente com as palavras definidas no dicionário.

f) “hashcat.dict” – Por último, é definido o dicionário de palavras que é utilizado na hora do ataque, neste exemplo foi utilizado o dicionário de palavras padrão do Hashcat, porém, qualquer arquivo de texto pode ser utilizado como dicionário.

Figura 13 - Informações Hashcat



Fonte: Elaborada pelo autor

Na figura 13, pode-se notar o software Hashcat imprimindo algumas informações básicas sobre o computador utilizado, como por exemplo, memória ram, versão do driver, clock, processador utilizado, quantidade de processadores.

6 RESULTADOS

Embora o software não tenha conseguido recuperar todas as hashes fornecidas, o mesmo teve uma média satisfatória de recuperação em grupos individuais. Conforme o resultado apresentado na tabela 2, o Hashcat conseguiu mostrar um desempenho relativamente satisfatório se comparado com os parametros utilizados para tal recuperação.

Como pode ser visto na tabela 2, nenhuma das hashes conseguiram ser recuperadas completamente. Isso ocorre por conta da máscara que foi utilizada no ataque híbrido, portanto, apenas uma quantidade significativa foi recuperada durante os testes.

Tabela 2 - Tabela comparativa dos resultados das hashes recuperadas

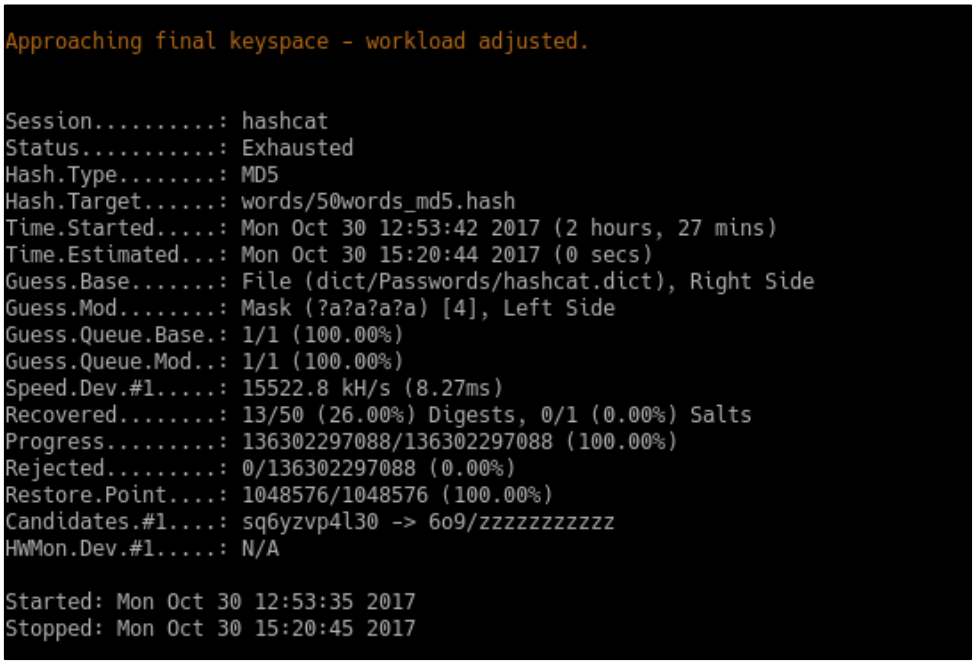
N° de hashes	Recuperação Total	Recuperação Parcial	Nenhuma Recuperação	N° hashes recuperadas	Link Resultado
50				13	Link
100				26	Link
150				35	Link
200				47	Link
250				64	Link
300				80	Link
350				111	Link
400				98	Link
450				136	Link
500				133	Link
				Total Recuperado	743/2,750

Fonte: Elaborada pelo autor.

Com esta técnica de ataque fornecida pelo Hashcat, é possível especificar um incrementador para esta máscara, sendo assim, conforme a ataque terminar, o Hashcat irá reiniciar o ataque incrementando uma máscara a mais no ataque, os limites podem ser especificados nos parâmetros de execução do Hashcat.

Nas figuras 13 à 23, pode ser conferido mais detalhadamente as informações relacionadas a recuperação das hashes de cada arquivo que foi utilizado no Hashcat.

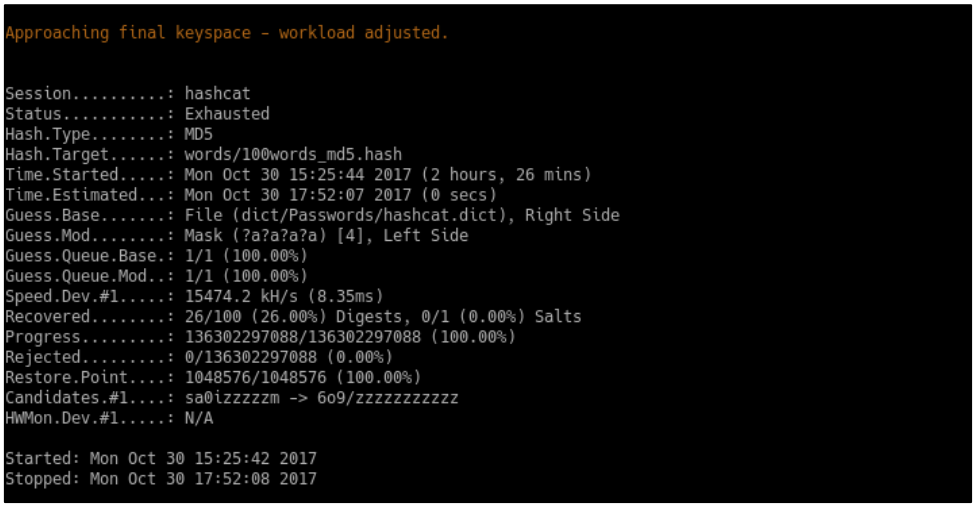
Figura 14 - Resultado do ataque em lista de 50 palavras



Fonte: Elaborada pelo autor.

Como visto na figura 15, o Hashcat conseguiu recuperar cerca de 26% das hashes propostas no arquivo que continham 50 hashes criptografadas, isso é devido ao fato de ter sido utilizada uma máscara de quatro caracteres.

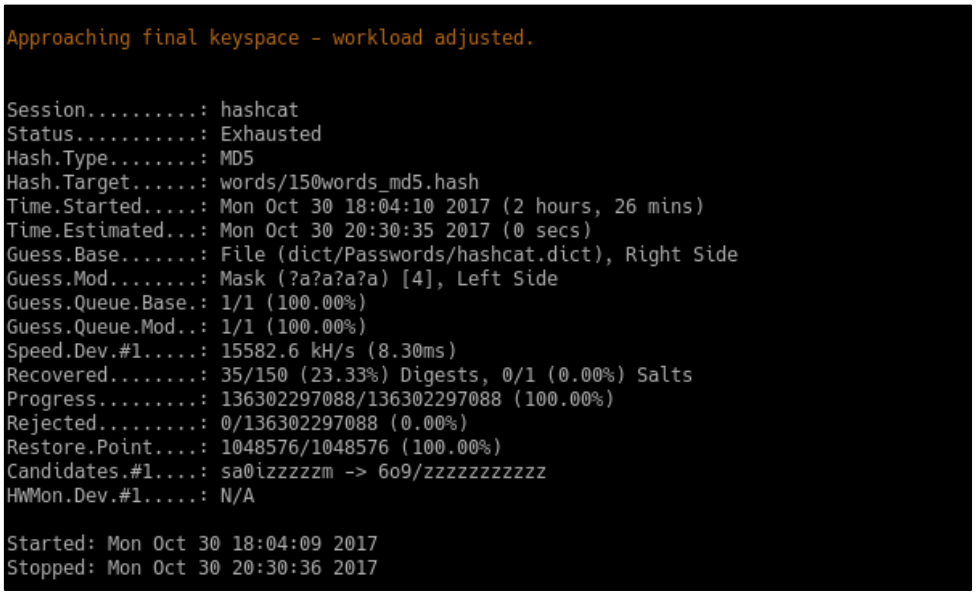
Figura 15 - Resultado do ataque em uma lista de 100 palavras



Fonte: Elaborado pelo autor.

Novamente, como pode-se ver na figura 16, o Hashcat não conseguiu recuperar todas as senhas referentes aos arquivos de hashes, apenas 26% das hashes foram recuperadas de um total de 100 hashes.

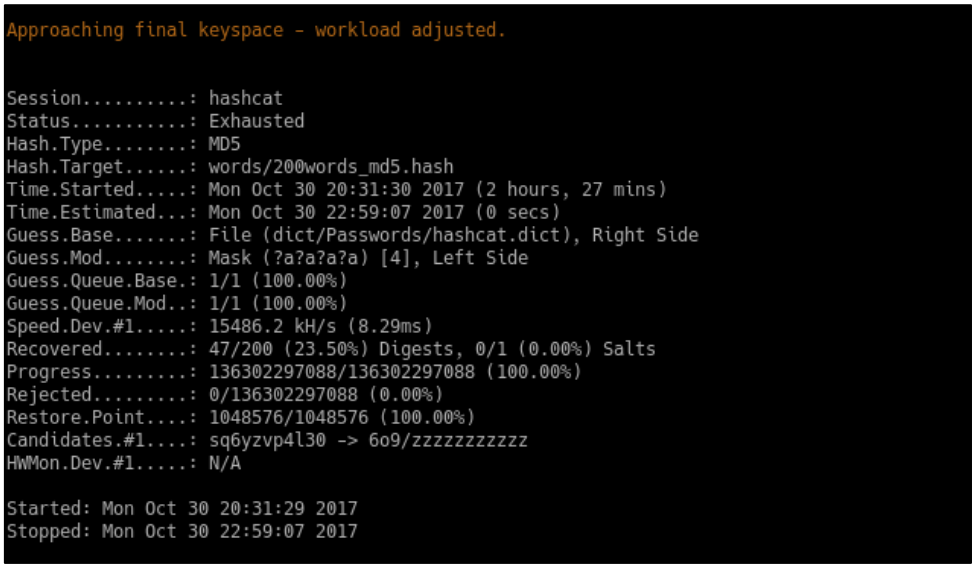
Figura 16 - Resultado do ataque em uma lista de 150 palavras



Fonte: Elaborado pelo autor.

Na figura 17, o hashchat teve um desempenho um pouco menos satisfatório, apenas 23.33% de hashes recuperadas de um total de 150.

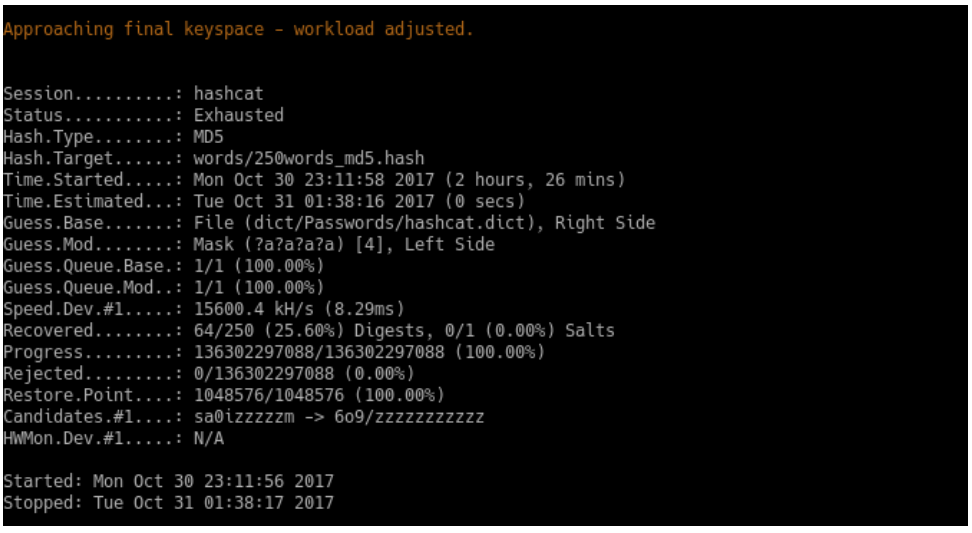
Figura 17 - Resultado do ataque em uma lista de 200 palavras



Fonte: Elaborado pelo autor.

Novamente, na figura 18, o resultado obtido não foi satisfatório o suficiente, apenas 23.50% das hashes recuperadas de 200.

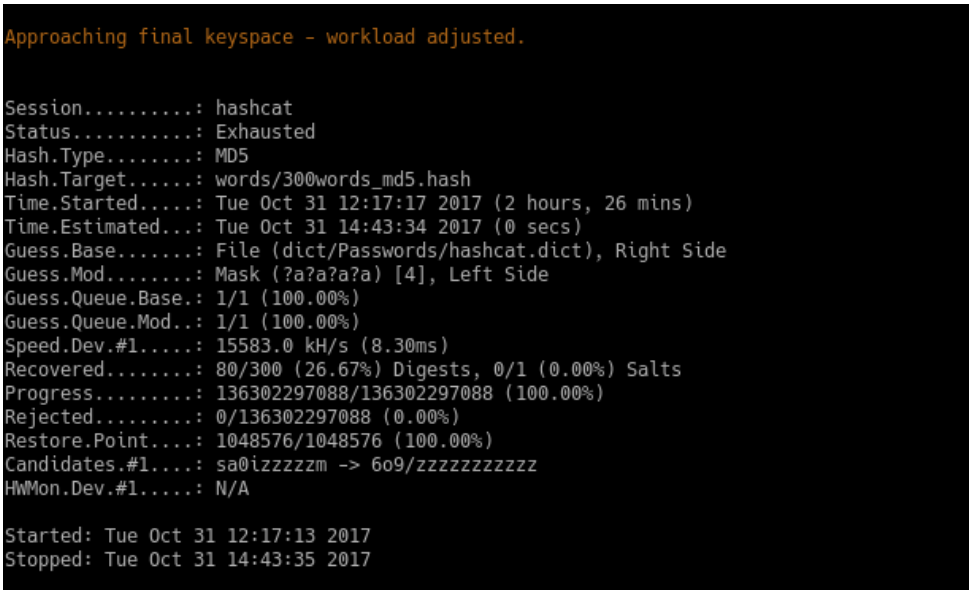
Figura 18 - Resultado do ataque em uma lista de 250 palavras



Fonte: Elaborado pelo autor.

Como visto na figura 19, o Hashcat teve um aproveitamento de 25.60% de hashes recuperadas de um total de 250, uma margem bem baixa de recuperação.

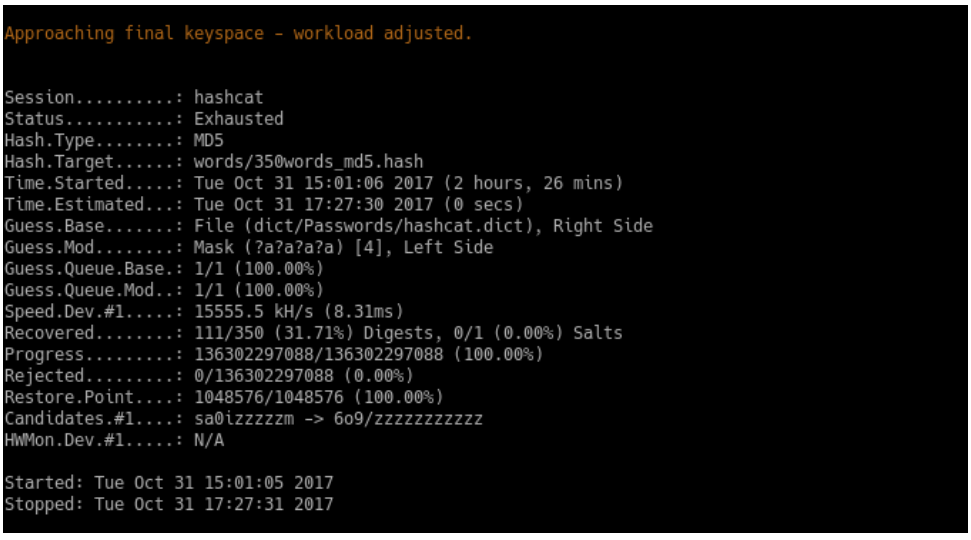
Figura 19 - Resultado do ataque em uma lista de 300 palavras



Fonte: Elaborado pelo autor.

Como visto na figura 20, o Hashcat conseguiu recuperar 26.67% de um total de 300 hashes cifradas.

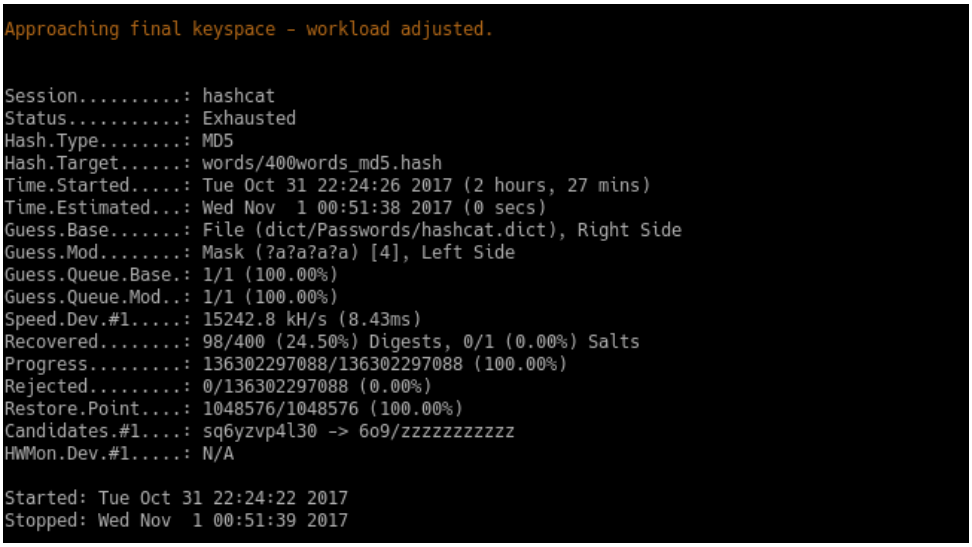
Figura 20 - Resultado do ataque em uma lista de 350 palavras



Fonte: Elaborada pelo autor.

Como pode-se ver na figura 21, o Hashcat teve uma recuperação de 31.71% de recuperação de hashes, de um total de 350 palavras. Nesse teste, o Hashcat teve um desempenho um pouco mais significativo.

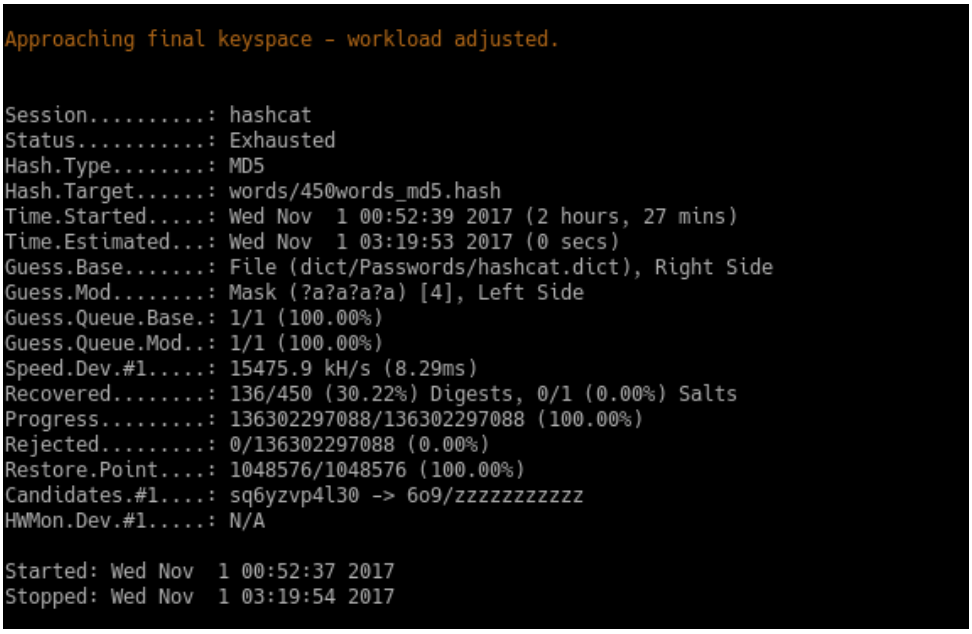
Figura 21 - Resultado do ataque em uma lista de 400 palavras



Fonte: Elaborada pelo autor.

Já na figura 22, o Hashcat teve uma margem de recuperação de 24.50% de recuperação das hashes, de um total de 400 hashes encriptadas.

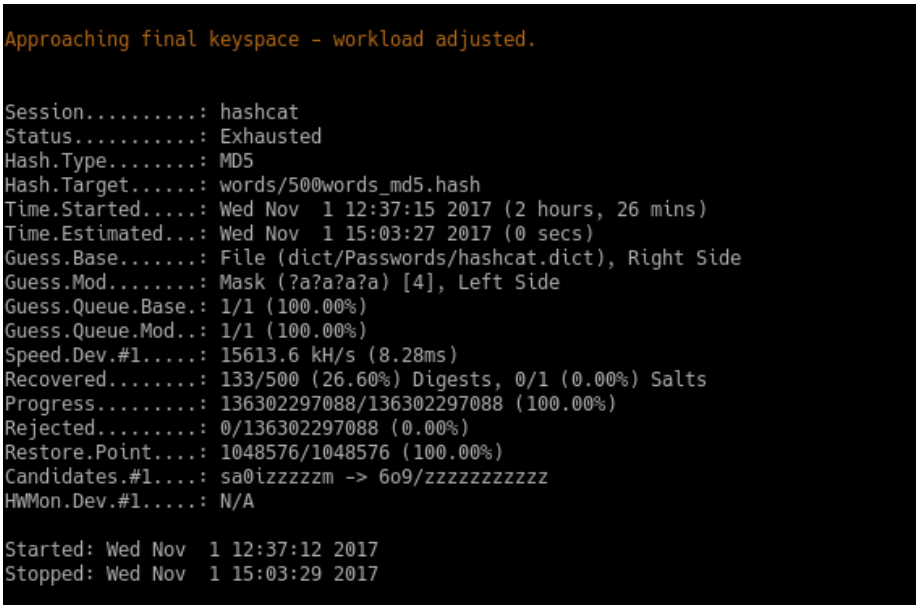
Figura 22 - Resultado do ataque em uma lista de 450 palavras



Fonte: Elaborado pelo autor.

Como visto na figura 23, o Hashcat teve um total de 30.22% de recuperação das hashes, de um total de 450 palavras.

Figura 23 - Resultado do ataque em uma lista de 500 palavras



Fonte: Elaborada pelo autor.

Como visto na figura 24, o Hashcat teve um total de 26.60% de recuperação de hashes de um total de 500 hashes cifradas.

Como pode-se ver nas imagens, os resultados de conclusão do Hashcat podem fornecer algumas informações extras, como por exemplo: Quantidade de hashes recuperadas, tempo de execução, velocidade do ataque, uma estimativa de tempo, as palavras que o software estava utilizando para “adivinhar” qual a hash equivalente, e também, o progresso que foi feito percorrendo o dicionário de palavras.

O desempenho do Hashcat poderia ter sido aproveitado de forma mais satisfatória caso ele fosse utilizado como cluster, desta forma, o Hashcat distribuiria o processamento das hashes entre os diferentes clusters, conectados pela internet ou fisicamente, e assim, conseguiria identificar novas palavras que fossem equivalentes as hashes fornecidas.

O número total de hashes utilizadas foram 2.750, sendo o total de recuperação, 743 hashes e com o arquivo maior de 500 hashes, que o mesmo levou aproximadamente 2 horas e 26 minutos para a recuperação de cada arquivo com as hashes. O tempo total de recuperação com as 2.750 hashes foi de aproximadamente 24 horas e 24 minutos.

As hashes foram montadas com a utilização de palavras aleatórioas, juntamente com a geração de 50 a 50 vocábulos do inglês e o dicionário utilizado foi um dicionário da lingua inglesa convencional do software utilizado. Os vocábulos utilizados tornan-se mais familiarizados como uma senha convencionais muitas vezes utilizadas por usuários comuns por serem aleatórias, gerando uma certa segurança.

O software utilizado teve um grande desempenho na recuperação de hashes, tornando um programa rápido, prático e sem grandes dificuldades em seu uso. Sua utilização se mostrou eficiente e eficar em termos de recuperação, porém, se não fornecidos os devidos parametros o software tende a perder o seu valor se tornando lento na recuperação das hashes e aumentando o uso computacional.

7 CONCLUSÃO

No presente trabalho, foi possível testar a técnica e validar a eficiência do software Hashcat utilizando a técnica de ataque híbrido, com hashes criptografas em MD5. Porém, vale ressaltar que mesmo que todas as hashes não foram recuperadas com sucesso, o ataque ainda se mostra eficiente se combinarmos mais máscaras para o ataque, ou até mesmo, um dicionário que contenha mais palavras ou palavras mais variadas.

A partir dos testes que foram realizados, pode-se constatar que o algoritmo MD5 tem uma segurança relativamente boa, porém, em apenas poucas horas em que o software estava rodando, tentando quebrar essas hashes, foi possível conseguir no mínimo um terço de todas as hashes fornecidas, sendo assim, por mais amplo que seja sua utilização, não é totalmente recomendado o uso do algoritmo MD5 como forma de segurança em senhas, ainda que sendo utilizado por alguns sistemas atuais, gerando algumas reclamações sobre a segurança de tal agoritmo.

Após a realização dos diferentes testes, com quantidades de palavras diferentes, pode-se chegar a uma conclusão que quanto maior a lista de hashes a serem quebradas, também é necessário que o tamanho do dicionário utilizado seja maior, assim como, o tamanho da máscara seja levemente aumentado,

podendo assim, dar maior liberdade para o software trabalhar tentando adivinhar quais palavras conseguem ser correspondentes as hashes.

Como trabalhos futuros, sugere-se, nessa mesma área de estudo a utilização de mais máscaras e dicionários diferentes para validar a segurança das hashes cifradas com o algoritmo MD5, assim como, a e eficiência do software Hashcat em quebra-las. Este trabalho forneceu um grande conhecimento na área da criptoanálise, segurança digital, criptografia e recuperação de hashes tornando magnificamente satisfatório no campo da segurança digital em buscas de uma evolução na área estudada.

REFERÊNCIAS

AGRAWAL, N; KUMAR, M; RIZVI, M.A. **Transposition Cryptography Algorithm Using Tree Data Structure**. Disponível em: https://www.researchgate.net/profile/Nikhil_Agrawal14/publications. Acesso em: 10 nov. 2017.

ALMEIDA, R. **Perícia Forense Computacional: Estudo das técnicas utilizadas para coleta e análise de vestígios digitais.** Disponível em: http://www.fatecsp.br/dti/tcc/tcc0035.pdf. Acesso em: 10 jun. 2017.

ANAND, D; KHEMCHANDANI, V; SHARMA, K., R. **Identity-Based Cryptography Techniques And Applications (A Review)**. Disponível em: http://www.joconline.com.cn/CN/article/openArticlePDFabs.jsp?id=156477. Acesso em: 10 nov. 2017.

BATISTA, C. F. A. **Métricas de Segurança de Software**. Disponível em: http://www.maxwell.vrac.puc-rio.br/10990/10990_1.PDF. Acesso em: 16 jun. 2017.

BILLATNAPIER. **EXEMPLO DO ALGORITMO RSA**. Disponível em: http://billatnapier.com/design_tips241.htm. Acesso em: 19 de jun de 2017.

BROOKE, T. **Exemplo do Algoritmo MD5.** Disponível em: http://www.makeuseof.com/tag/md5-hash-stuff-means-technology-explained/. Acesso em: 19 de jun de 2017.

CAPTCHA. **EXEMPLO DE CAPTCHA**. Disponível em: http://www.captcha.net/images/recaptcha-example.gif. Acesso em: 28/05/2017

CASTELLÓ, T; VAZ, V. **História da Criptografia**. Disponível em: http://www.gta.ufrj.br/grad/07_1/ass-dig/HistriadaCriptografia.html. Acesso: em 16 jun. 2017.

C-DAC/KOLKATA. **EXEMPLO DE COMO FUNCIONA A FORENSE COMPUTACIONAL.** Disponível em: https://image.slidesharecdn.com/cyberforensicstandardoperatingprocedures-111212225028-phpapp01/95/cyber-forensic-standard-operating-procedures-5-728.jpg?cb=1323730615. Acesso em: 28 de maio de 2017.

CERT. **Site do Cert para verificar as estatísticas sobre as invasões, ataques e etc**. Disponível em: http://www.cert.br/stats/incidentes/2013-jan-dec/tipos-ataque-acumulado.html. Acesso em: 10 nov. 2017.

CHATTERJEE, D; NATH, J; DASGUPTA, S; NATH, A. **A New Symmetric Key Cryptography Algorithm Using Extended Msa Method**: Djsa Symmetric Key Algorithm. Disponível em: http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?reload=true&arnumber=5966411. Acesso em: 10 nov. 2017.

DAVI. **CRIPTOANÁLISE DIFERENCIAL.** Disponível em: http://www.davi.ws/ss/crip_diferencial.htm. Acesso em: 28 de maio de 2017.

EBERMANN P. **EXEMPLO DO ALGORITMO AES**. Disponível em: http://crypto.stackexchange.com/questions/2711/does-the-mixcolumns-step-come-before-or-after-addroundkey-in-aes-decryption. Acesso em: 19 de jun de 2017.

FRANÇA, W. B. A. **Criptografia**. Disponível em: http://www.ucb.br/sites/100/103/TCC/22005/WaldizarBorgesdeAraujoFranco.pdf. Acesso em: 10 nov. 2017.

FRANCESE, J. P. S. **Criptografia Quântica, 2008**. Disponível em: https://www.gta.ufrj.br/grad/08_1/quantica/index.html. Acesso em: 16 jun. 2017.

GABRIEL, 2010. **HISTÓRIA: CRIPTOGRAFIA E CRIPTOANÁLISE.** Disponível em: https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2010_2/gabriel/hist.htm. Acesso em: 28 de maio de 2017.

GOOGLE. **EXEMPLO DE CAPTCHA**. Disponível em: https://www.google.com/recaptcha/intro/images/hero-recaptcha-demo.gif. Acesso em: 28 de maio de 2017.

HASHCAT, 2017. **EXEMPLO DE ATAQUE FORÇA BRUTA.** Disponível em: https://hashcat.net/wiki/lib/exe/fetch.php?media=hashcat-legacy.png. Acesso em: 28 de maio de 2017.

HASHCAT, 2017. **EXEMPLO DE UM ATAQUE POR DICIONÁRIO**. Disponível em: https://hashcat.net/wiki/lib/exe/fetch.php?media=hashcat-legacy.png. Acesso em: 28 de maio de 2017.

HASHCAT. **HASHCAT OCLPLUS**. Disponível em: https://hashcat.net/wiki/lib/exe/fetch.php?media=oclhashcat-plus.png. Acesso em: 28 de maio de 2017.

IBM. **EXEMPLO DE CHAVE SIMÉTRICA**. Disponível em: http://www.ibm.com/support/knowledgecenter/SSFKSJ_9.0.0/com.ibm.mq.sec.doc/q009800_.htm. Acesso em: 19 de jun de 2017.

JAILIN, S; KAYALVIZHI, R; VAIDEHI, V. **Performance Analysis of Hybrid Cryptography for Secured Data Aggregation in Wireless Sensor Networks**. Disponível em: http://ieeexplore.ieee.org/xpl/articleDetails.jsp%3Freload%3Dtrue%26arnumber%3D5972447. Acesso em: 10 nov. 2017.

JASPER, N. **HISTÓRIA, TÉCNICAS E CLASSIFICAÇÃO DE ALGORITMOS ESTEGANOGRÁFICOS.** Disponível em: https://www.researchgate.net/profile/Nichols_Jasper2/publication/263469536_Historia_Tecnica_e_Classificacao_de_Algoritmos_Esteganograficos/links/00b7d53b0324eb9689000000.pdf. Acesso em: 10 set. 2017

KUMAR, C; DUTTA, S; CHAKBORTY, S. **Musical Cryptography Using Genetic Algorithm**. Disponível em: https://scholar.google.co.in/citations?user=8eGGd4YAAAAJ&hl=en. Acesso em: 10 nov. 2017.

KUMAR, N; AGRAWAL, S**. An Efficient and Effective Lossless Symmetric Key Cryptography Algorithm for an Image**. Disponível em: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=7012788&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D7012788. Acesso em: 10 de nov de 2017.

LANCENEWMAN, 2014. **EXEMPLO DE UM ATAQUE MAN IN THE MIDDLE.** Disponível em: http://lancenewman.me/wp-content/uploads/2014/10/mitm.jpg. Acesso em: 28 de maio de 2017.

LARROSA, O. A. G; et. al. **A Influência e Importância da Criptografia na Velocidade de Rede Ethernet**. Disponível em: http://web.unipar.br/~seinpar/2013/artigos/Otavio%20Augusto%20Goncalves%20Larrosa.pdf. Acesso em: 16 de fev de 2017.

LEARN CRYPTOGRAPHY, 2017. **BRUTE FORCE ATTACK.** Disponível em: https://learncryptography.com/attack-vectors/brute-force-attack. Acesso em: 28 de maio de 2017.

LEARN CRYPTOGRAPHY, 2017. **DICTIONARY ATTACK.** Disponível em: https://learncryptography.com/attack-vectors/dictionary-attack. Acesso em: 28 de maio de 2017.

LEARN CRYPTOGRAPHY, 2017. **EXEMPLO DE UM ATAQUE MAN IN THE MIDDLE**. Disponível em: https://learncryptography.com/assets/img/content/ManInTheMiddle.png. Acesso em: 28 de maio de 2017.

LEARN CRYPTOGRAPHY, 2017. **EXEMPLO DE UM ATAQUE POR ANÁLISE DE FREQUÊNCIA.** Disponível em: https://learncryptography.com/assets/img/content/english_letter_frequency_alphabetic.svg. Acesso em: 28 de maio de 2017.

LEARN CRYPTOGRAPHY, 2017. **FREQUENCY ANALYSIS.** Disponível em: https://learncryptography.com/attack-vectors/frequency-analysis. Acesso em: 28 de maio de 2017.

LEARN CRYPTOGRAPHY, 2017. **MAN IN THE MIDDLE.** Disponível em: https://learncryptography.com/attack-vectors/man-in-the-middle-attack. Acesso em: 28 de maio de 2017.

LOPES, E; QUARESMA, P. **Criptografia**. Disponível em: http://www.mat.uc.pt/~pedro/lectivos/CodigosCriptografia1011/artigo-gazeta08.pdf. Acesso em: 19 mar. 2017.

MACHIAVELO, A. **O Que Vem à Rede**. Disponível em: http://gazeta.spm.pt/pesquisa?campopesquisa=Ant%C3%B3nio%20Machiavelo&campo=autr.nome. Acesso em: 19 mar. 2017.

MARCIANO, J. L; LIMA-MARQUES, MAMEDE. **O Enfoque Social da Segurança Da Informação**. Disponível em: http://www.scielo.br/pdf/ci/v35n3/v35n3a09.pdf Acesso em: 16 jun. 2017.

MATTAR, F. N. **Pesquisa de Marketing**. Disponível em: http://www.elsevier.com.br/site/institucional/Minha-pagina-autor.aspx?aid=76424&seg=6. Acesso em: 10 nov. 2015.

OLIVEIRA, R. R**. Criptografia Simétrica E Assimétrica**: Os Principais Algoritmos De Cifragem. Disponível em: http://www.ronielton.eti.br/publicacoes/artigorevistasegurancadigital2012.pdf. Acesso em: 16 jun. 2017.

PEREIRA, P. J. F. **Segurança da informação digital. Cadernos BAD. Associação Portuguesa de Bibliotecários Arquivistas e Documentalistas**. 2005.

RIVEST, R. L. **Descrição do algoritmo de criptografia MD5**. Disponível em: https://tools.ietf.org/html/rfc1321. Acesso em: 16 jun. 2017.

SILVA, E. V. P**. Introdução à Criptografia RSA**. Disponível em: http://www.impa.br/opencms/pt/eventos/downloads/jornadas_2006/trabalhos/jornadas_elen_pereira.pdf. Acesso em: 19 de jun de 2017.

SOORTA, R; MADHURI, K; VUDAYAGIRI, A. **Hardware Random Number Generator for Cryptography**. Disponível em: http://arxiv.org/abs/1510.01234. Acesso em: 10 nov. 2017.

SSL2BUY. **EXEMPLO DE CHAVE ASSIMÉTRICA**. Disponível em: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences/. Acesso em: 19 de jun de 2017.

STALLINGS, W. **EXEMPLO DE ATAQUE DES**. Disponível em: http://conteudo.icmc.usp.br/pessoas/otj/SSC0547/Cap03.pdf. Acesso em: 28 de maio de 2017.

VIGNESH SAKTHI, R; SUDHARSSUN, S; KUMAR JEGADISH, K.J. **Limitations of Quantum & the Versatility of Classical Cryptography**: A Comparative Study. Disponível em: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5383498&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D5383498. Acesso em: 10 nov. 2017.

WEBTUGA. **EXEMPLO DE BLOQUEIO DE IP POR TENTIVA.** Disponível em: https://blog.webtuga.pt/imagens/desbloqueariplimite.gif. Acesso em: 28/05/2017

WXWIDGETS. **Biblioteca usada para criar a interface da aplicação**. Disponível em: http://wxwidgets.org/. Acesso em: 10 nov. 2018.

ZHOU, P; DING, Q. **The Key Exchange Research of Chaotic Encryption Chip Based on Elliptic Curve Cryptography Algorithm**. Disponível em: http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?reload=true&arnumber=5288393&contentType=Conference+Publications. Acesso em: 10 nov. 2017.