CLOUDERA: UMA ABORDAGEM PARA ANÁLISE DE LOGS DO PROXY SQUID-CACHE USANDO HADOOP, FLUME, MAPREDUCE E IMPALA

CLOUDERA: AN APPROACH FOR PROXY SQUID-CACHE LOGS ANALYSIS USING HADOOP, FLUME, MAPREDUCE AND IMPALA

Volnei Cervi PUTTINI

vcputtini@gmail.com, Pós-Graduação Lato Sensu em BI em Bigdata, UniAnchieta

Juliano SCHIMIGUEL

Professor do Centro Universitário Padre Anchieta, Jundiaí/SP, schimiguel@gmail.com

Resumo

Nosso objetivo é demonstrar uma abordagem no contexto do Big Data a qual permita a coleta, armazenagem persistente, tratamento e geração de resultados para análises dos dados mediante o uso das ferramentas disponíveis na plataforma Cloudera. Os dados utilizados são os registros de log gerados em tempo real pelo servidor proxy Squid-cache como resultado dos acessos à Internet feitos a partir da rede local. Usando o Apache Hadoop para o armazenamento de massa e processamento MapReduce quando necessário, conversão dos dados armazenados no HDFS para tabelas Impala para criação de rotinas de manipulação dos dados usando a linguagem SQL, e o Apache Flume para coleta de transmissão dos dados entre a fonte e o Apache Hadoop.

Palavras-chaves: Big Data; Apache Hadoop; Apache Flume; Cloudera. Log.

Abstract

Our goal is to demonstrate an approach in the context of Big Data that enables collection, persistent storage, processing and results generation for data analysis using the tools available on the Cloudera platform. The data used are real-time logs generated by the Squid-cache proxy server, as a result of Internet access from the local network. Using Apache Hadoop for mass storage and MapReduce processing when necessary, converting HDFS-stored data to Impala tables for creating data manipulation routines using the SQL language and Apache Flume to collect data transmission between source and Apache Hadoop.

Key-words: Big Data; Apache Hadoop; Apache Flume.; Cloudera; Log.

INTRODUÇÃO

Historicamente sempre foi um grande desafio efetuar analises consistentes em dados de log devido a fatores como: possuem formatos de dados complexos e proprietários, são comumente armazenados no mesmo host da instalação do sistema, estão em formato ASCII (texto puro), por vezes binários porém este formato está fora do nosso escopo, são semiestruturados, são gerados em tempo real ou muito próximo disso e seus arquivos podem conter milhões de eventos registrados, a segurança desses dados deve ser considerada sempre como uma prioridade. Dado este cenário analisar manualmente é tarefa impossível, e criar-se programas capazes de automatizar este processo também torna-se complexo pois algumas necessidades devem ser sanadas tais como a interpretação dos dados registrados por diferentes sistemas simultaneamente, o armazenamento deve ser centralizado e persistente a fim de podermos ter acesso a todo o histórico a qualquer momento, além de se criar rotinas padronizadas capazes de manipular os registros e deles extraírem as informações necessárias de modo consistente e principalmente coletar os dados em tempo real e transferi-los a um sistema capaz de armazená-los permitindo processamento centralizado e ágil. A segurança destes dados também é preocupação importante pois se tratam de informações sensíveis podendo ser utilizadas inclusive como prova em ações judiciais.

Analisando tais características constatamos que nossas necessidades podem ser suportadas pelas tecnologias quem vem sendo consolidadas no âmbito da ciência do Big Data. O Big Data se estabeleceu devido a alguns fatores primários os quais remetem as variedades nos formatos dos dados e como eles devem ser armazenados, na velocidade na qual esses dados são gerados e seu imenso volume. Além destas variáveis (TAURION, 2013) complementa com "[...] veracidade dos dados (os dados tem significado ou são sujeira?) e valor para o negócio. Outra questão que começa a ser debatida é a privacidade, tema bastante complexo e controverso.", aqui ele nos proporciona um vislumbre das complexidades em se distinguir dados úteis dos não úteis além de toda a problemática envolvendo as questões sobre privacidade. Quando colocamos o gerenciamento de log sobre está perspectiva, se torna importante que dediquemos nossos esforços na criação de soluções para este fim, conforme demonstrados nos trabalhos correlatos abaixo citados e comentados.

(TEOH et al, 2004) apresentam uma abordagem para inspeção de logs de mediante o desenvolvimento de uma aplicação própria. Apesar de não fazem uso de técnicas de Big Data a aplicação é capaz de análises estatísticas e visualizações para auxiliar na tarefa dos administradores, tal qual é um dos nossos objetivos a serem alcançados.

Conforme proposto por (JAKUB e BRANISOVÁ, 2015, tradução nossa) em "Detecção de anomalias de arquivos de log usando técnicas de mineração de dados", utilizam de técnicas avançadas "para detecção de anomalias em arquivos de log, baseado em técnicas de mineração de dados para criação de regras dinâmicas. Para suportar o processamento paralelo, utilizamos o framework Apache Hadoop, fornecendo armazenamento distribuído e processamento distribuído de dados.", o foco do trabalho é mais amplo quando pretende tratar de vários sistemas diferentes e não somente de um especifico como no nosso caso, porém também fazem uso do Apache Hadoop e

MapReduce, mas não informando sobre os meios utilizados para coleta dos dados. Nós iremos fazer uso do Apache Flume para coletar os dados nos servidores proxy Squid e transmiti-los para o Apache Hadoop aonde eles serão armazenados.

Ao unir as técnicas de análise e processamento massivo de dados proporcionada pela ciência do Big Data aplicada ao campo do Gerenciamento de Log, poderemos desenvolver soluções capazes de garantir tanto as necessidades dos administradores de sistemas bem como dos requisitos de segurança da informação.

SEGURANÇA DA INFORMAÇÃO E O GERENCIAMENTO DOS LOGS

A Segurança da Informação conceitua as bases para podermos melhor entender o por que que devemos dar as informações a importância necessária.

A informação é um ativo que, como qualquer outro ativo importante, é essencial para os negócios de uma organização e consequentemente necessita ser adequadamente protegida (ABNT NBR ISO/EIC 27002, 2005).

Ainda há as seguintes definições que devem ser citadas:

1. Preservação da confidencialidade, integridade e disponibilidade da informação; adicionalmente, outras propriedades, tais como autenticidade, responsabilidade, não repúdio e confiabilidade, podem também estar envolvidas (ABNT NBR ISO/EIC 27002, 2005).
2. Podemos definir a Segurança da Informação como uma área do conhecimento dedicada à proteção de ativos da informação contra acessos não autorizados, alterações indevidas ou sua indisponibilidade. (SêMOLA,2003,p.43).

Conforme (ABNT NBR ISO/EIC 27002, 2005), um ativo pode ser "qualquer coisa que tenha valor para organização". Desse modo dentro do contexto deste trabalho o registro dos acessos à Internet é um ativo a ser monitorado, controlado e mantido, possivelmente indefinidamente, a fim que as organizações possam através dessas informações gerir estas atividades que tem como potencial impactar negativamente nas atividades internas das empresas, causando prejuízos das mais diversas ordens.

Conforme explicado por Lyra (2008), ”Quando falamos em segurança da informação, estamos nos referindo a tomar ações para garantir a confidencialidade, integridade, disponibilidade e demais aspectos da segurança das informações dentro das necessidades do cliente.”

Dadas estas características das informações obtidas mediante as análises dos registros de log, devem ser geridas sob a mesma ótica, desse modo torna-se indispensável que o gerenciamento dos logs tenham estabelecidas políticas próprias.

Segundo (Kent e Souppaya, 2006, tradução nossa), ”Um log é a gravação dos eventos que ocorrem dentro dos sistemas e das redes das organizações. Os logs são compostos pelas entradas de logs; cada entrada contém informações relacionadas a um

evento específico que ocorreu dentro de um sistema ou da rede. Muitos logs contém registros relacionados a segurança dos computadores.[...]", definem perfeitamente o que é o log e a sua importância no desenvolvimento das políticas de segurança as quais impactam diretamente na qualidade, segurança e confiabilidade dos ativos das empresas, daí a necessidade em se auditar os logs constantemente.

Porém auditar logs é uma tarefa considerada tediosa e muito trabalhosa, mas de modo algum deve ser negligencia pelos administradores de sistemas. Em trabalho correlato, (SIMON et al, 2008, tradução nossa) demonstram que "auditoria procura identificar e evitar ações suspeitas e fraudulentas por parte do usuário, coletando dados sobre suas atividades no banco de dados.", os autores nos fazem compreender a importância da auditoria quando falam que "As informações coletadas são analisadas a fim de descobrir problemas de segurança e sua origem. A necessidade de identificar quais foram as ações e quais os padrões suspeitos são importantes requisitos para a segurança do sistema.", os autores focam entretanto seu trabalho na auditoria visando a segurança dos dados armazenados nos Sistemas Gerenciadores de Banco de Dados (SGBD) propondo a utilização de Sistemas Gerenciadores de Stream de Dados (SGSD) para enviar os dados dos registros de log para o sistema de destino a fim de serem processados e analisados. É interessante notar que neste artigo não são usados termos comuns ou ferramentas do Big Data porém são claras as semelhanças quando descrevem que "[...] o processamento de fluxos de dados (*streams*) é realizado através de máquinas de processamento de *streams*.[...]" a qual é usada para transmitir dos dados coletados para o destino final, e também que "A integração desses dois métodos, auditoria e processamento de *streams*, pode garantir que os registros de log sejam processados em tempo real [...]" corroborando nossa ideia sobre a importância em se criar soluções destinadas à este modelo de processamento.

Assim o registro dos acessos à Internet é um ativo a ser monitorado, controlado e mantido, possivelmente indefinidamente, a fim que as organizações possam através dessas informações gerir estas atividades que tem como potencial impactar negativamente em suas atividades, causando prejuízos das mais diversas ordens. Tais conceitos convergem sempre para a tríade denominada de CID, ou seja, Confidencialidade, Integridade e Disponibilidade, sendo este paradigma os pilares fundamentais da Segurança de Informação.

TECNOLOGIAS ENVOLVIDAS

PROXY SQUID: FONTE DOS DADOS

O servidor Squid atua como *proxy,* sendo um intermediário entre a rede local e a Internet, provendo diversas funcionalidades as quais nos permitem controlar os acessos, otimizar a utilização da banda e registrar em tempo real todas as requisições efetuadas pelos usuários da rede em um arquivo de log.

PLAFORMA OPERACIONAL CLOUDERA

O Cloudera é uma plataforma de software open-source baseada em GNU-Linux, o qual é mantido pela empresa que leva o mesmo nome, conforme pode ser lido no site oficial, ”Cloudera Inc., é uma empresa de software sediada nos EUA que fornece uma plataforma de software para engenharia de dados, armazenamento de dados, aprendizado de máquina e análise que é executada na nuvem ou no local. (CLOUDERA, 2018)."

Assim Cloudera tornou-se uma plataforma operacional de grande valia para aqueles que desejam iniciar estudos em Big Data ou aqueles que possuem pretensões mais profissionais. O Cloudera permite com sua distribuição denominada Quickstart, composta por um ambiente totalmente integrado baseado na distribuição Linux CentOS no modelo de VM, permite a instalação simples e rápida para testes e estudos, podendo também ser ampliada para ambientes mais complexos caso necessário. Porém dadas as necessidades de hardware poderoso e armazenamento de enormes capacidades, para o uso em escala realmente profissional, talvez seja mais conveniente por optar por instalações do Cloudera em sistemas de terceiros com o dimensionamento adequado para grandes escalas.

Apache Hadoop

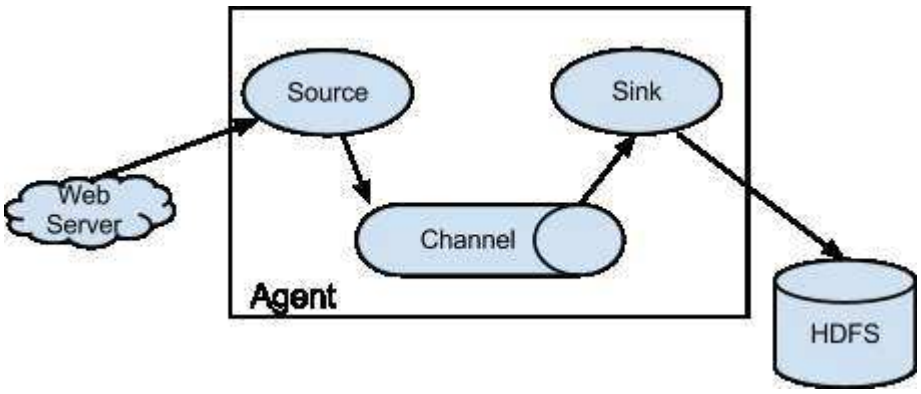
O Hadoop é uma plataforma open-source destinada a armazenamento massivo de dados de qualquer tipo, além de permitir o processamento desses dados em clusters. Suas principais características são: o sistema de arquivos HDFS - Hadoop File System que permite o armazenamento distribuído por milhares de sistemas interligados via rede, além da sua arquitetura MapReduce destinada ao processamento paralelo das aplicações, abstraindo toda a complexidade em se mapear, organizar e retornar os dados armazenados no cluster.

Apache Flume

Conforme descrito no site oficial do projeto "O Flume é um serviço distribuído, confiável e disponível para coletar, agregar e mover com eficiência grandes quantidades de dados de log. Tem uma arquitetura simples e flexível baseada em fluxos de dados de fluxo contínuo. Ele é robusto e tolerante a falhas, com mecanismos de confiabilidade ajustáveis e muitos mecanismos de failover e recuperação. Ele usa um modelo de dados extensível simples que permite a aplicação analítica on-line.".Em suma o Flume é um coletor universal de dados, que por meio de suas configurações é capaz de ler os arquivos de log em tempo de execução e transferir estes dados pela rede até o nó central aonde ele deverá ser armazenado.

O Flume é capaz mediante sua estrutura de agentes de capturar em lotes ou tempo real os dados oriundos de registros de logs de sistemas dos mais variados tipos.

Figura 1: Modelo de processamento de *data stream*.



Fonte: https://flume.apache.org/

A Figura 1 mostra como é estruturado o software denominado agente, que é o responsável por ler, processar e transmitir os dados entre a origem e o seu destino.

- Sources: Recuperam os dados e enviam para os canais.
- Channels: Os canais mantém as filas de dados e os transmitem entre fontes e coletores, atuando também na sincronização do fluxo.
- Sinks: Os sinks (coletores) processam os dados obtidos dos canais e os entregam aum destino, no nosso caso o HDFS.
-

Para que o agente possa operar ele deve possuir ao menos um grupo completo dos componentes. O agente é executado em sua própria instancia Java VM.

Apache Impala

O Impala é um sistema capaz de efetuar consultas via instruções SQL aos dados armazenados no HDFS ou no Apache HBase. Sua arquitetura não utiliza os mecanismos do MapReduce, pois acessa diretamente os dados por meio de um mecanismo de consulta distribuído próprio, resultando em enorme ganho de desempenho, sendo inclusive superior ao Hive.

As tabelas são os principais depósitos dos dados do Impala. Apesar de seus esquemas de linha e colunas serem parecidos aos dos SGDB tradicionais, possui recursos diferenciados, tais como particionamento normalmente associados as sistemas de data warehouse de ponta.

A estrutura lógica das tabelas se baseia nas definições das suas colunas, partições e demais características. Já a estrutura física é baseada na estrutura de diretórios do HDFS, sendo que cada tabela tem seus dados armazenados em arquivos dentro desta estrutura. (IMPALA, 2018).

HUE

Hue é uma interface visual entre as quantidades enormes de dados dos data warehouses e as ferramentas de BI/Machine Learning. Ele é destinado ao desenvolvedor de aplicativos de dados permitindo que ele possa iniciar seus projetos rapidamente.

Em resumo o Hue é um IDE, que permite tanto aos programadores bem como aos cientistas/analistas acessarem seus dados de forma simples, como por exemplo digitar no editor de textos instruções SQL para serem então executadas pelo Apache Impala, por exemplo. (HUE, 2018)

IMPLEMENTAÇÃO

Implementar localmente uma infraestrutura de hardware para suportar processamento massivo é algo que normalmente pode não ser viável para as empresas, pois como é sabido o Hadoop necessita a partir de dezenas de nós para que suas qualidades possam ser utilizadas plenamente. Não obstante, isso também é verdade quanto a capacidade de armazenamento dos dados, que como já sabemos é gigantesca. Sendo assim o ambiente implementado é suficiente apenas para efetuar os testes demonstrados nesse trabalho.

· Hardware Físico

– CPU: Intel Xeon E3-1240

– RAM: 32GB Totais

· Appliance Virtual17

– VM: Cloudera 5.13.0 QuickStart

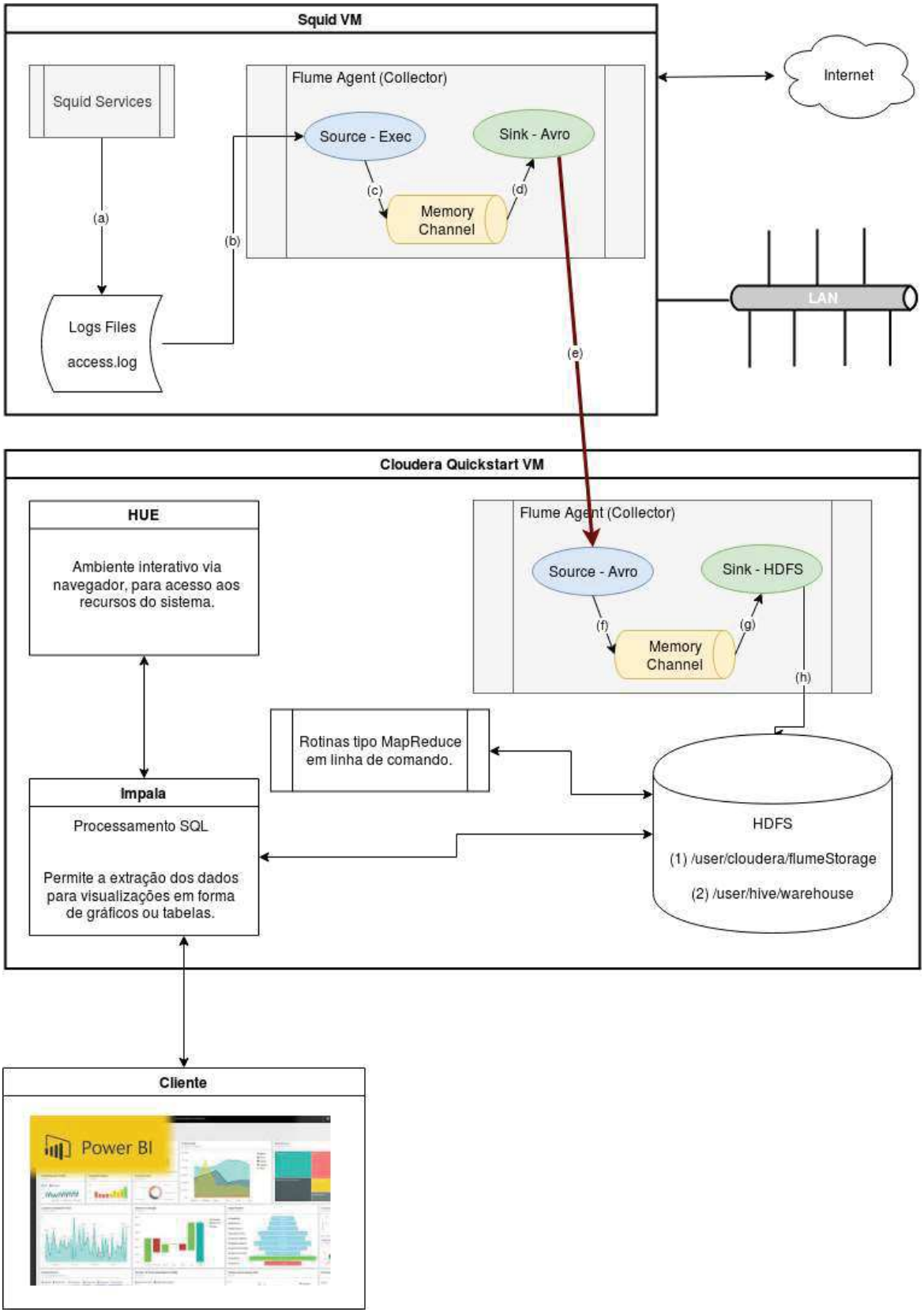
– Tamanho do VHD18: 55GB

– CPU Virtual: 04

– RAM Virtual: 10GB

– Hadoop Cluster Nodes: 01

Figura 2 - Fluxo dos dados pelos sistemas até ser armazenado para processamento.



Descrição do fluxo dos dados

(a) O proxy Squid intercepta o tráfego de Internet e grava no arquivo de log, por padrão access.log, todos os registros.

(b) O agente Flume usa como fonte para ingestão os dados gravados no arquivo access.log por meio do source tipo exec, que irá executar o comando tail -F access.log e enviar a saída para o canal (c).

(c) O canal, armazena os dados e os envia ao coletor (d).

(d) O coletor (sink) irá formatar os dados para o protocolo escolhido, no caso Avro, e mediante o uso de uma porta de comunicação enviará os datagramas para o source do host de destino (e).

(e) Este agente está sendo executado dentro do ambiente Cloudera. Seu source foi configurado com o tipo Avro e atribuídos o endereço IP deste host e a porta de comunicação.

(f) O canal de memória foi configurado do mesmo modo que o usado no item (c), pois é necessário que ambos tenham capacidades semelhantes.

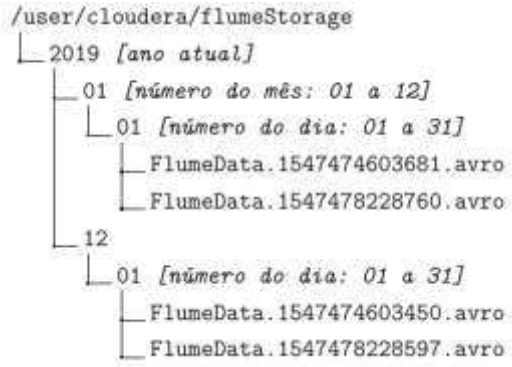
(g) o coletor (sink) foi configurado como tipo HDFS, pois irá armazenar os dados recebidos do canal diretamente no sistema Hadoop HDFS..

Áreas para armazenamento - HDFS

Armazenamento do dados coletados:

Esta área é destinada à gravação dos dados coletados o servidor proxy Squid. Os dados recebidos são gravados conforme a figura abaixo:

Figura 3. Estrutura em árvore dos diretórios



Esta organização tem o objetivo de tornar o acesso aos dados o mais simples e rápido possível. No topo da árvore está o diretório que recebe como nome o ano atual, no nível abaixo temos o diretório que recebe o nome do número do mês corrente e abaixo dele os dias do mês. Dentro do diretório dia do mês são gravados os dados coletados em arquivos que representam 01(uma) hora de logs, sendo assim teremos a cada dia 24(vinte e quatro) arquivos.

Podemos então notar o quanto esta estrutura simplifica a localização dos dados e como os arquivos gerados são menores sua leitura é muito rápida. Para simplificar os processos de identificação dos arquivos seu nome é assim composto:

- Prefixo que identifica o coletor: FlumeData.
- Data de criação: No formato Unix timestamp, a qual é o número de segundos que se passaram desde 00:00:00 de quinta-feira, 1 de janeiro de 1970 até a data da criação do aquivo.
- Sufixo: Identifica o Source, neste caso o Avro.

O conteúdo desses arquivos são as linhas de log separadas por espaços em branco.

Exemplo:

1546553031.980 30120 192.168.1.20 TCP MISS/200 668 GET https://tr2.terra.com/broadcast/sub/

ch=pt-BR.BreakingNewsm=529s=M/ch=pt-BR.NewStoriesm=251495s= M0.0460431667643667051546552990823? - ORIGINAL DST/208.84.244.40 application/json

Armazenamento definitivo dos dados:

Os dados coletados serão movidos para outro local, o qual será o banco de dados que irá permitir tanto a persistência quanto a possibilidade de acesso para execução das análises necessárias.

Figura 4. Local padrão de armazenagem no ambiente Cloudera

```
/user/hive/warehouse
└─ squidlogs.db [Nome do banco de dados]
   └─ accesslog [Nome do tabela]
      └─ logdate='yyyy-mm-dd' [Partições nomeadas com datas]
         └─ 474571a55b9088db-edee7a0800000000_1636462698_data.0.parq
```

A parte mais relevante na estrutura acima são as partições, o Impala permite criar uma tabela particionada por um atributo definido pelo usuário, no nosso caso o atributo logdate, o qual nada mais é que a data dos logs registrados. Desse modo são criadas pastas e dentro delas são gravados apenas os registros referentes à aquela data. Isto permite uma busca mais otimizada pelos algoritmos do Impala.

Tratamento dos dados

Estrutura do Log

Os registros do log são gerados uma linha por vez e representam uma ação que ocorreu, por exemplo, a tentativa de acesso a um Web site que foi negada pelo proxy Squid. O formato do registro que estamos utilizando é o padrão do proxy Squid que é assim composto:

- Não possui cabeçalho informando os nomes dos campos ou quaisquer outras das suas características, ou seja não possui esquema.
- Os campos são separados por espaços em branco, ASCII (32_{10}).
- O registro possui 10(dez) campos distintos os quais são identificados apenas pela sua posição na linha.

Tabela 1. Descrição dos campos registrados pelo Squid no arquivo access.log.

Posição	Nome Atributo	Função
0	unixts	Data e hora da ocorrência da ação. Fomato Unix timestamp.
1	timems	Tempo gasto nesta requisição, dado em milissegundos.
2	host	IP ou nome do host que fez a requisição ao cache.
3	action	A ação tomada pelo cache. Exemplo: TCP_HIT ou TCP_MISS.
4	lenreg	Tamanha em bytes do objeto requisitado.
5	method	O método utilizado, podendo ser: GET, PUT, POST, etc.
6	url	A URL (objeto) solicitada ao cache.
7	cliname	Pode ou não conter o nome usuário se houver autenticação.
8	direct	Informa se o proxy fez um requisição direta. Geralmente um tunel SSL.
9	mime	Informa o tipo do arquivo baixado: HTML, TXT, IMAGEM, etc.).

Nota: A coluna Nome Atributo, é apenas ilustrativa.

Tabelas Impala

- ·accesslog_tmp: Os dados são primeiro armazenados nesta tabela e depois inseridos na tabela principal. Tem por objetivo manter os dados em seu estado original até serem copiados para a tabela de final.
- accesslog: Contém os dados dos logs para armazenamento definitivo e processamento.

Ambas possuem esquemas semelhantes, conforme abaixo apresentado:

Os campos em destaque na Tabela 2, são utilizados para auxiliar a criação das consultas, facilitando a manipulação das seleções por datas e horas, diminuindo a necessidade de utilização de formulas muito complexas. É importante salientar que o campo logdate apensar de poder ser utilizado nas seleções, é um campo especial com a função primária de separar a tabela pela partição data de criação dos logs.

Tabela 2. Esquema da tabela Impala.

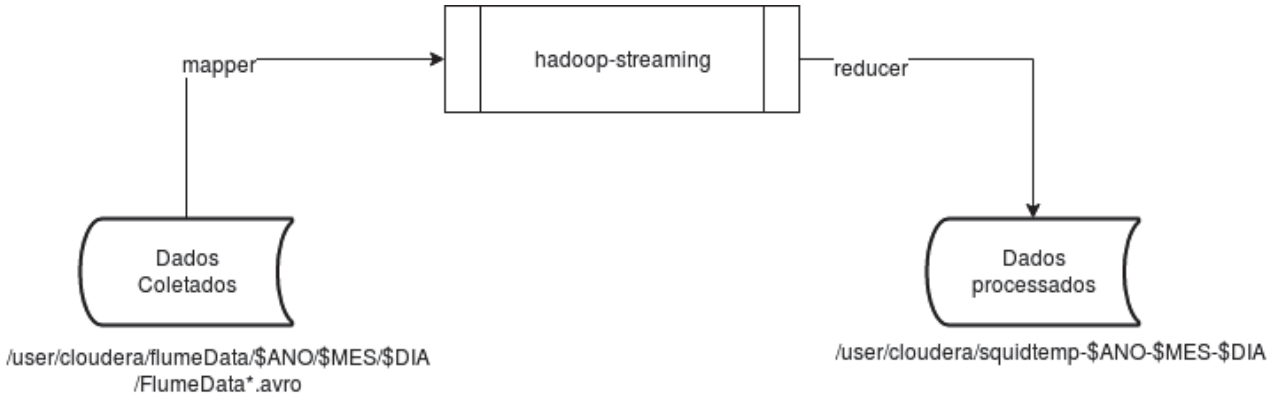
Campo	**Tipo**	**Notas**
Unixts	double	Data no formato UNIX timestamp.
timems	string	Duração da requisição em milissegundos.
Host	string	IP do Host.
Action	string	Ação tomada pelo proxy.
Lenreg	int	Tamanho em bytes da requisição.
method	string	Métodos utiizados.
url	string	A URL solicitada.
cliname	string	Contém o nome do cliente. Padrão ’-’.
Direct	string	Requisição direta pelo proxy ou via outro.
mime	string	Tipo de arquivo baixado.
sdatetime	string	Data e Hora no formato AAAA-MM-DD HH:MM:SS.
stime	string	Apenas a hora no formato HH:MM:SS.

iday	int	Dia.
imonth	int	Mês.
iyear	int	Ano.
logdate	string	Campo especial que identifica a partição: AAAA-MM-DD.

Parte 1: Pré-processamento MapReduce

Quando os dados são coletados pelo Flume eles são armazenados exatamente como são lidos, ou seja, o formato da linha de log, como já explicado, tem seus campos separados por espaços em branco. Para nossos fins é mais conveniente que os campos estejam usando como separador a vírgula. Para tanto foram criadas duas rotinas em linguagem Python que serão executadas pelo programa cliente hadoop no modo linha de comando.

Figura 3: Ilustração do fluxo do processamento do script.



Parte 2: Carga dos dados para o Impala

Dois procedimentos são necessários nessa fase sendo eles:

1. Carga dos dados do HDFS para a tabela temporária: A carga dos dados armazenados no HDFS para a tabela temporária é importante para que tenhamos uma cópia dos dados puros no caso da necessidade de reprocessá-los sem necessidade de reprocessamento.
2. Carga dos dados da tabela temporária para a tabela de produção com o processamento dos campos adicionais, preenchendo-os com os dados auxiliares para consultas por data e hora.

Listagem 1: O procedimento automatizado completo é feito pelo Bash script

```
#!/bin/bash
# Este script ira sempre processar os dados do DIA ANTERIOR sendo
# nenecessario que todos os arquivos de log estejam completamente
# gerados, completando o dia.

MAPPER="/home/cloudera/python/squid_map.py"
REDUCER="/home/cloudera/python/squid_reduce.py"
# O comando 'date' ira retorna a data do dia *anterior*
ONTEM=`date -d "1 days ago" +"%Y/%m/%d"`
ANO=${ONTEM:0:4}
MES=${ONTEM:5:2}
DIA=${ONTEM:8:2}
echo ">> Referente: $ANO-$MES-$DIA"
# Executa os script MapReduce
hadoop jar
/usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.6.0-mr1-cdh
5.13.0.jar \
  -D mapreduce.job.name="CvntLogDaily" \
  -files ${MAPPER},${REDUCER} -mapper ${MAPPER} -reducer ${REDUCER} \
  -input /user/cloudera/flumeStorage/${ANO}/${MES}/${DIA}/FlumeData*.avro \
  -output /user/cloudera/squidtemp-${ANO}-${MES}-${DIA}
# Ajusta o modo de acesso para o diretorio para leitura
hadoop fs -chmod 0777 /user/cloudera/squidtemp-${ANO}-${MES}-${DIA}
# A particao da tabela deve ser criada *antes* da carga dos dados caso ela ja nao exista
impala-shell -u cloudera -d squidlogs --var=ANO=$ANO --var=MES=$MES
--var=DIA=$DIA -q 'alter table accesslog_tmp add partition
(logdate="${var:ANO}-${var:MES}-${var:DIA}"); load data inpath
"/user/cloudera/squidtemp-${var:ANO}-${var:MES}-${var:DIA}/part-00000"
```

```
overwrite into table squidlogs.accesslog_tmp

partition(logdate="${var:ANO}-${var:MES}-${var:DIA}");'

# Remove o diretorio apos os dados serem lidos para a tabela

hadoop fs -rm -r -f /user/cloudera/squidtemp-${ANO}-${MES}-${DIA}

# Insere os dados do log processado na particao correta da tabela temporaria.

# Alem de inserir os dados, popula os campos adicionais com a data da

# transacao do log, nos seguintes formatos: aaaa-mm-dd hh:mm:ss,

# dia, mes e ano em campos separados

impala-shell -u cloudera -d squidlogs --var=ANO=$ANO --var=MES=$MES

--var=DIA=$DIA -q 'insert overwrite table squidlogs.accesslog partition

(logdate) select unixts, timems, host, action, lenreg, method, url, cliname, direct, mime,
from_unixtime(cast(unixts as bigint), "yyyy-MM-dd HH:mm:ss") as sdatetime,
from_unixtime(cast(unixts as bigint), "HH:mm:ss") as stime,
cast(from_unixtime(cast(unixts as bigint), "dd") as int) as iday,
cast(from_unixtime(cast(unixts as bigint), "MM") as int) as imonth,
cast(from_unixtime(cast(unixts as bigint), "yyyy") as int) as iyear, logdate from
squidlogs.accesslog_tmp where logdate="${var:ANO}-${var:MES}-${var:DIA}";'

# eof
```

DEMONSTRAÇÃO

Esta seção demonstrará algumas maneiras de se extrair informações sobre os eventos dos logs. As possibilidades para extração dos dados são muitas, principalmente se usarmos a linguagem SQL e o ambiente interativo HUE. O banco de dados contém aproximadamente de 10 milhões de linhas de eventos gerados pelos acessos à Internet. Para nossos objetivos esse número é expressivo e suficiente, porém no contexto do Big Data é uma quantidade bem pequena.

Figura 4: Contagem simples mensal dos acessos aos domínios .com.br, agrupados por hosts.

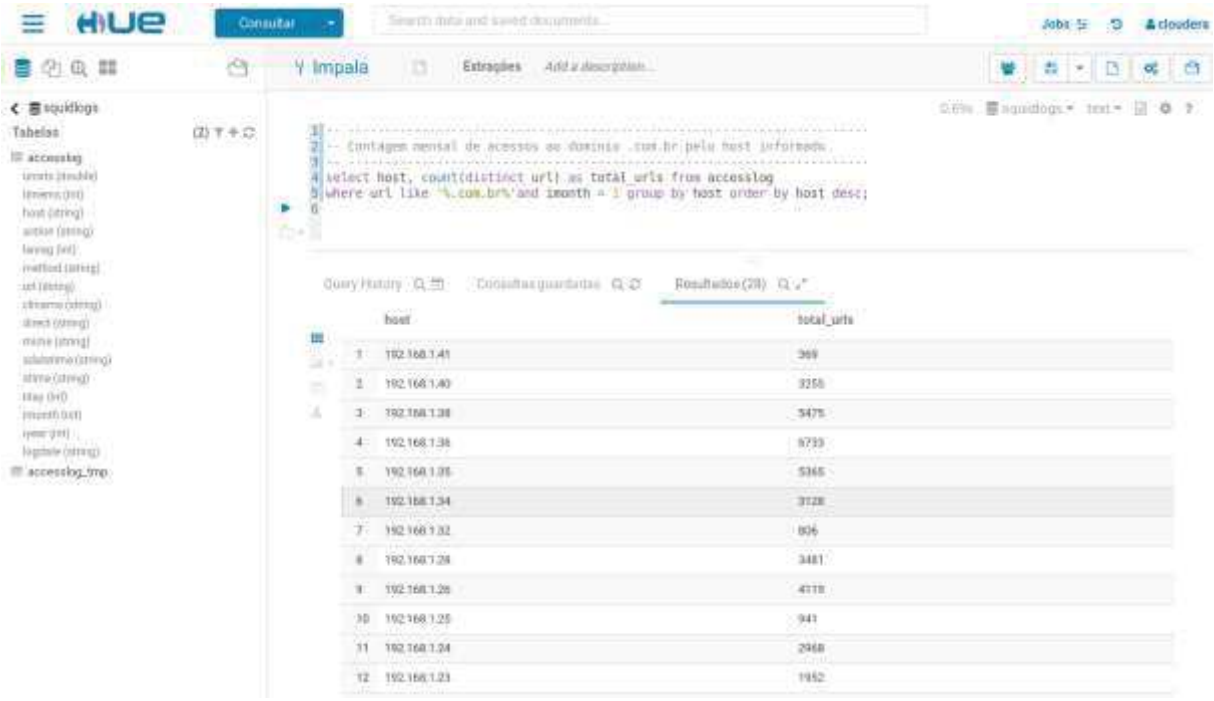


Figura 5: Gráfico representado a contagem mensal.

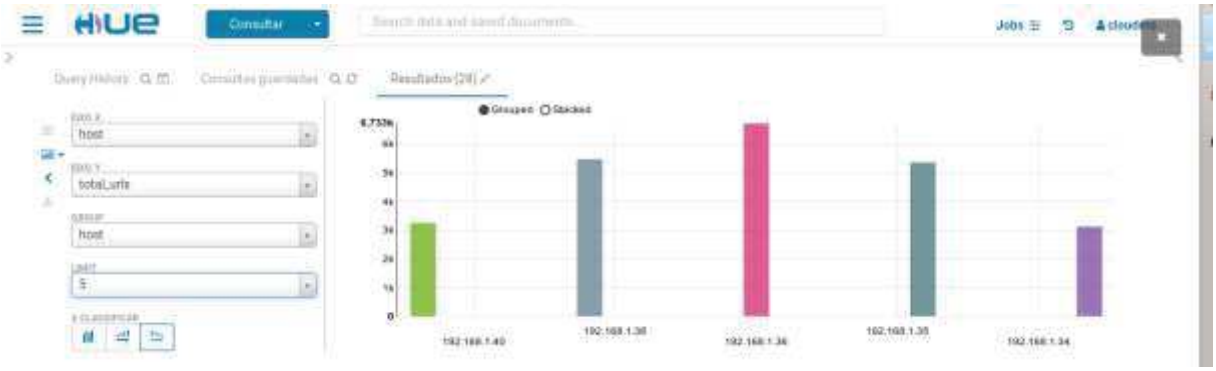
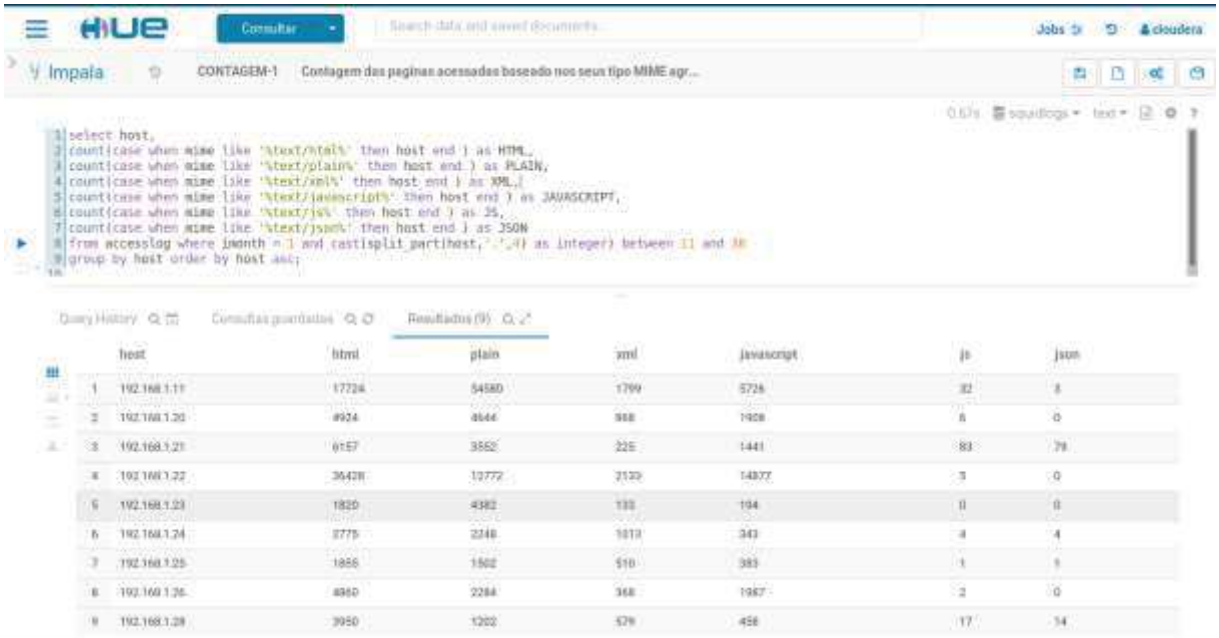


Figura 6: Contagem das páginas acessadas pelo tipo MIME.



CONCLUSÃO E TRABALHOS FUTUROS

Os problemas relacionados ao gerenciamento de logs por nós estudados, podem ter sua complexidade reduzida ao aplicarmos técnicas de Big Data, para o tratamento dos dados. Grande parte da complexidade é atribuída a coleta e centralização dos dados e ao desenvolvimento de algoritmos capazes de efetuar as análises de forma automática e padronizada, já que cada sistema possui particularidades quanto ao log gerado, tanto em formato como nos dados arquivados propriamente ditos.

Ao usarmos as tecnologias sugeridas demonstramos que é possível implementar uma estrutura com baixa complexidade e facilmente reproduzível em ambientes de pequena escala quando necessário.

A utilização do MapReduce mostra-se muito útil quando do tratamento dos dados puros, permitindo sua sanitização a fim de que possamos garantir que apenas dados úteis sejam gravados nas bases de dados para posterior geração de resultados. Apesar da geração das análises não terem não terem sido feitas em tempo real, as mesmas técnicas podem ser adaptadas para ambientes nos quais a detecção e prevenção de intrusões em tempo real sejam prioridade.

Em trabalhos futuros serão abordados os seguintes temas: A implementação de agentes Flume em múltiplos servidores para estudos dos impactos do tráfego massivo de dados na rede e como solucionar os problemas dos gargalos, e o desenvolvimento e otimização de configurações para coleta e transmissão de dados diretamente para o Hadoop HDFS.

REFERÊNCIAS BIBLIOGRÁFICAS

ABNT NBR ISO/EIC 27002. Tecnologia da Informação - Técnicas de segurança - Código de prática para a gestão de segurança da informação. 2005. Primeira Edicao: 31.08.2005 Válida a partir de 30.09.2005. Disponível em: <http://www.fieb.org.br/download/senai/nbr_iso_27002.pdf>. Acesso em: 23 Jan 2019.

CLOUDERA. Cloudera. 2018. Disponível em: <https://www.cloudera.com/>. Acesso

em: 10 Dez 2018.

GARTNER. IT Glossary: What is big data? 2018. Disponível em: <https://www.

gartner.com/it-glossary/big-data>. Acesso em: 23 Nov 2018.

HUE. Hue Overview. 2018. Disponível em: <http://http://gethue.com/overview/>. Acesso em: 13 Dez 2018.

IMPALA. Apache Impala Overview. 2019. Disponível em: <https://impala.apache.org/>. Acesso em: 10 Jan 2019.

KENT, K.; SOUPPAYA, M. Guide to computer security log management. NIST specialpublication, v. 92, 2006. Disponível em: <https://csrc.nist.gov/publications/detail/sp/800-92/final>. Acesso em: 01 Jan 2019.

SÊMOLA, M. Gestão da segurança de informação: uma visão executiva. 11o reimpressão. ed. [S.l.]: Rio de Janeiro. Elsevier, 2003.

SIMON, F.; SANTOS, A. L. dos; HARA, C. S. Um sistema de auditoria baseado na análise de registros de log. Escola Regional de Banco de Dados (ERBD'2008), Departamento de Informática - Universidade Federal do Paraná (UFPR), 2008. Disponível em: <https://www.researchgate.net/profile/Aldri_Santos/publication/239928992_Um_Sistema_de_Auditoria_baseado_na_An_alise_de_Registros_de_Log/links/53e51d2a0cf21cc29fcb4232/Um-Sistema-de-Auditoria-baseado-na-An-alise-de-Registros-de-Log.pdf>. Acesso em: 23 Jan 2019.

TAURION, Cezar. **Big Data:** Velocidade, Volume, Variedade, Veracidade Valor. Rio de Janeiro: Brasport Livros e Multimídia, 2013. (978-85-7452-608-9).

TEOH, S. T. et al. Visual data analysis for detecting flaws and intruders in computernetwork systems. IEEE Computer Graphics and Applications, Citeseer, v. 24, n. 5,2004. Disponível em: <https://www.usenix.org/legacy/event/lisa02/tech/full_papers/takada/takada_html/>. Acesso em: 23 Abril 2019.