

# Своя песочница

Data Engineer  
А. А. Селезнев



tg: @SeleznevArtem

 /NameArtem

 /seleznev-artem

 /seleznev.artem.info



[https://github.com/NameArtem/apache\\_cluster](https://github.com/NameArtem/apache_cluster)

tg: @SeleznevArtem



/NameArtem



/seleznev-artem



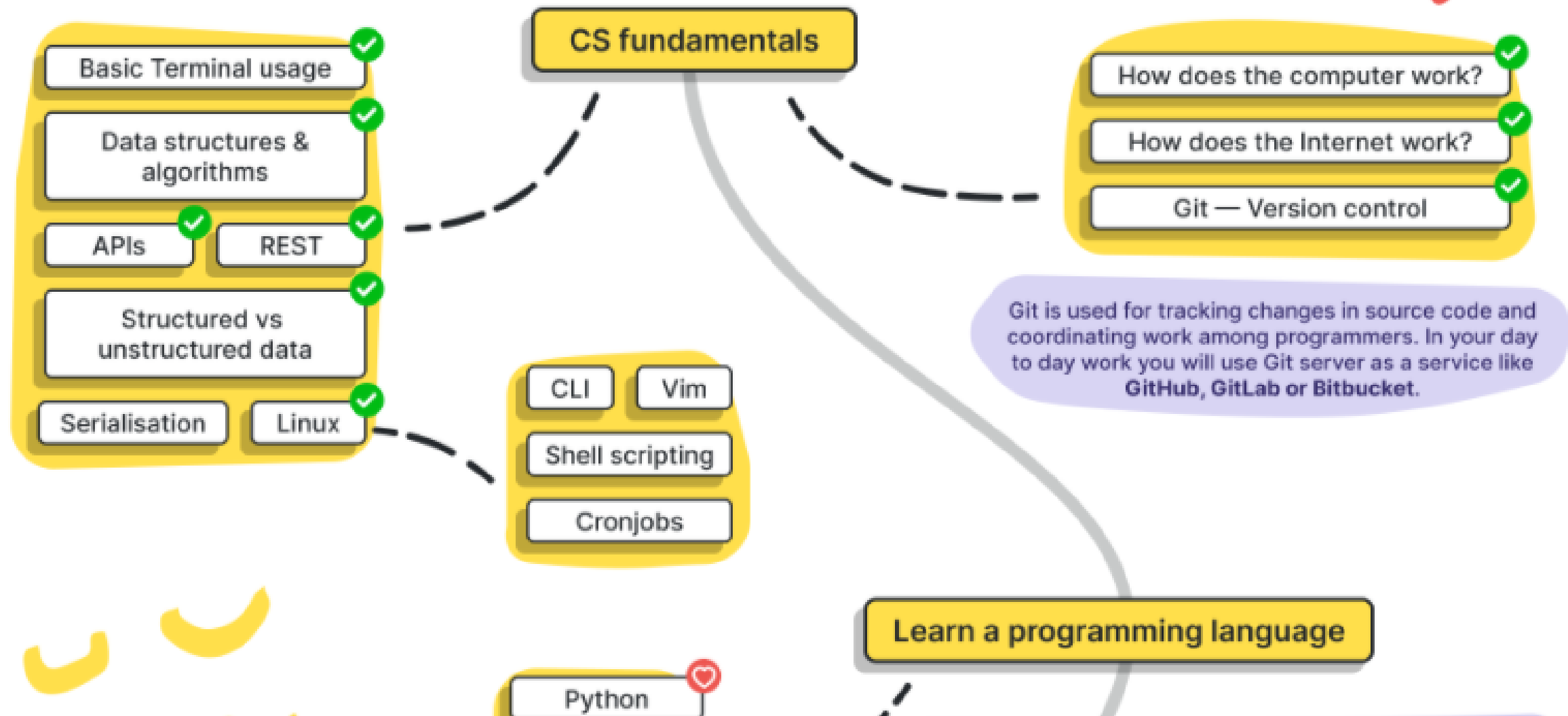
/seleznev.artem.info





- Personal recommendation
- General recommendation
- Cloud based

# Data Engineer in 2020



Nice to have 😎

Nice to have 😎

- Machine Learning



Nice to have 😎

- Machine Learning
- Infra as Code

Nice to have 😎

- Machine Learning
- Infra as Code
- ML/Data Ops

# Зачем?

(есть ли жизнь без Cloudera)

# UNDERSTANDABLE ENV

## STACKTRACE / DEBUG

Stack trace:

```
/Users/alexhall/Desktop/python/hearttrate/ignoreme/sandbox.py : 59 : <module>
    assert merge_sort(testcase) == testset
/Users/alexhall/Desktop/python/hearttrate/ignoreme/sandbox.py : 39 : merge_sort
    merge_sort(left), merge_sort(right)
/Users/alexhall/Desktop/python/hearttrate/ignoreme/sandbox.py : 39 : merge_sort
    merge_sort(left), merge_sort(right)
/Users/alexhall/Desktop/python/hearttrate/ignoreme/sandbox.py : 39 : merge_sort
    return merge(
        merge_sort(left), merge_sort(right)
    )
/Users/alexhall/Desktop/python/hearttrate/ignoreme/sandbox.py : 48 : merge
    left_idx += 1
```

## ВЫПОЛНЕНИЕ КОДА

```
27 1      def merge_sort(m):
28          """
29          Return a sorted copy of m
30          Uses the recursive merge sort algorithm
31          """
32
33 75001     if len(m) <= 1:
34 37500         return m
35 37501     middle = len(m) // 2
36 37501     left = m[:middle]
37 37501     right = m[middle:]
38 37501     return merge(
39 37501         merge_sort(left), merge_sort(right)
40         )
41
42 1      def merge(left, right):
43 37498         result = []
44 37498         left_idx, right_idx = 0, 0
45 518646     while left_idx < len(left) and right_idx < len(right):
46 481149         if left[left_idx] <= right[right_idx]:
47 237261             result.append(left[left_idx])
48 237261             left_idx += 1
49
50 243887         else:
51 243887             result.append(right[right_idx])
52 37497             right_idx += 1
53 16843         if left_idx < len(left):
54 37497             result.extend(left[left_idx:])
55 20654         if right_idx < len(right):
56 37497             result.extend(right[right_idx:])
            return result
```

# UNDERSTANDABLE ENV

## ART OF SPARK

```
In [ ]: def f(x):  
        global a  
        a+=x  
        RDD9.foreach(f)  
        RDD9.foreach(f)  
        print(a.value)  
        #Display should appear automatically
```

---

С чего начать?

# С КЛАСТЕРА

VIRTUAL



VDS RUS



VDS OTHERS



# С КЛАСТЕРА

VIRTUAL



VDS RUS



VDS OTHERS



= 7499 атак



# С КЛАСТЕРА

```
Sep 10 07:37:56 cnt-cls-m1 sshd[4478]: Failed password for invalid user loser from 118.188.20.5 port 45382 ssh2
Sep 10 07:39:11 cnt-cls-m1 sshd[4578]: Failed password for invalid user chef from 200.73.128.148 port 32832 ssh2
Sep 10 07:40:26 cnt-cls-m1 sshd[4663]: Failed password for invalid user dpi_clean from 118.89.236.249 port 55114 ssh2
Sep 10 07:42:22 cnt-cls-m1 sshd[4792]: Failed password for invalid user jiong from 118.188.20.5 port 37306 ssh2
Sep 10 07:42:28 cnt-cls-m1 sshd[4799]: Failed password for invalid user willie from 118.89.236.249 port 48640 ssh2
Sep 10 07:45:19 cnt-cls-m1 sshd[4993]: Failed password for invalid user ts3audiobot from 123.58.5.243 port 40125 ssh2
Sep 10 07:45:42 cnt-cls-m1 sshd[5029]: Failed password for invalid user security from 102.65.152.96 port 52626 ssh2
Sep 10 07:46:39 cnt-cls-m1 sshd[5087]: Failed password for invalid user squid from 118.89.236.249 port 35692 ssh2
Sep 10 07:48:05 cnt-cls-m1 sshd[5197]: Failed password for invalid user jboss from 118.188.20.5 port 45358 ssh2
Sep 10 07:50:08 cnt-cls-m1 sshd[5333]: Failed password for invalid user pruebas from 149.202.175.255 port 39357 ssh2
Sep 10 07:50:23 cnt-cls-m1 sshd[5349]: Failed password for invalid user alka from 102.65.152.96 port 59496 ssh2
Sep 10 07:50:29 cnt-cls-m1 sshd[5345]: Failed password for invalid user user3 from 69.250.156.161 port 41850 ssh2
Sep 10 07:52:14 cnt-cls-m1 sshd[5476]: Failed password for invalid user prueba from 118.188.20.5 port 37282 ssh2
Sep 10 07:52:42 cnt-cls-m1 sshd[5504]: Failed password for invalid user bialkowski from 49.232.106.176 port 39676 ssh2
Sep 10 07:53:39 cnt-cls-m1 sshd[5572]: Failed password for invalid user stephanie from 200.73.128.148 port 46662 ssh2
Sep 10 07:54:59 cnt-cls-m1 sshd[5663]: Failed password for invalid user news from 118.188.20.5 port 41312 ssh2
Sep 10 07:55:01 cnt-cls-m1 sshd[5666]: Failed password for invalid user user3 from 37.152.183.18 port 47628 ssh2
Sep 10 07:58:36 cnt-cls-m1 sshd[5937]: Failed password for invalid user smmsp from 176.192.126.27 port 57344 ssh2
Sep 10 07:59:06 cnt-cls-m1 sshd[5985]: Failed password for invalid user icinga from 118.188.20.5 port 33236 ssh2
Sep 10 07:59:15 cnt-cls-m1 sshd[5990]: Failed password for invalid user rstudio-server from 123.58.5.243 port 37313 ssh2
Sep 10 08:00:01 cnt-cls-m1 sshd[6051]: Failed password for invalid user redis from 102.65.152.96 port 45030 ssh2
Sep 10 08:02:18 cnt-cls-m1 sshd[6219]: Failed password for invalid user cfaniger from 69.250.156.161 port 48572 ssh2
Sep 10 08:02:37 cnt-cls-m1 sshd[6260]: Failed password for invalid user hone from 51.68.88.26 port 40048 ssh2
Sep 10 08:02:47 cnt-cls-m1 sshd[6284]: Failed password for invalid user netzke from 176.192.126.27 port 38316 ssh2
Sep 10 08:06:03 cnt-cls-m1 sshd[6598]: Failed password for invalid user openelec from 51.68.88.26 port 46070 ssh2
Sep 10 08:06:45 cnt-cls-m1 sshd[6651]: Failed password for invalid user enderdirt from 49.232.106.176 port 34416 ssh2
[root@cnt-cls-m1 ~]# journalctl | grep "Failed password for invalid user" | wc -l
7499
```

# С КЛАСТЕРА

- Ansible

# С КЛАСТЕРА

- Ansible
  - Apache Hadoop3

# С КЛАСТЕРА

- Ansible
  - Apache Hadoop3
    - Apache Spark 3

# С КЛАСТЕРА

- Ansible
  - Apache Hadoop3
    - Apache Spark 3
      - Apache Drill

# С КЛАСТЕРА

- Ansible
  - Apache Hadoop3
    - Apache Spark 3
      - Apache Drill
        - JupyterHub + Kernel

# С КЛАСТЕРА

- Ansible
  - Apache Hadoop3
    - Apache Spark 3
      - Apache Drill
        - JupyterHub + Kernel
          - Feature Store

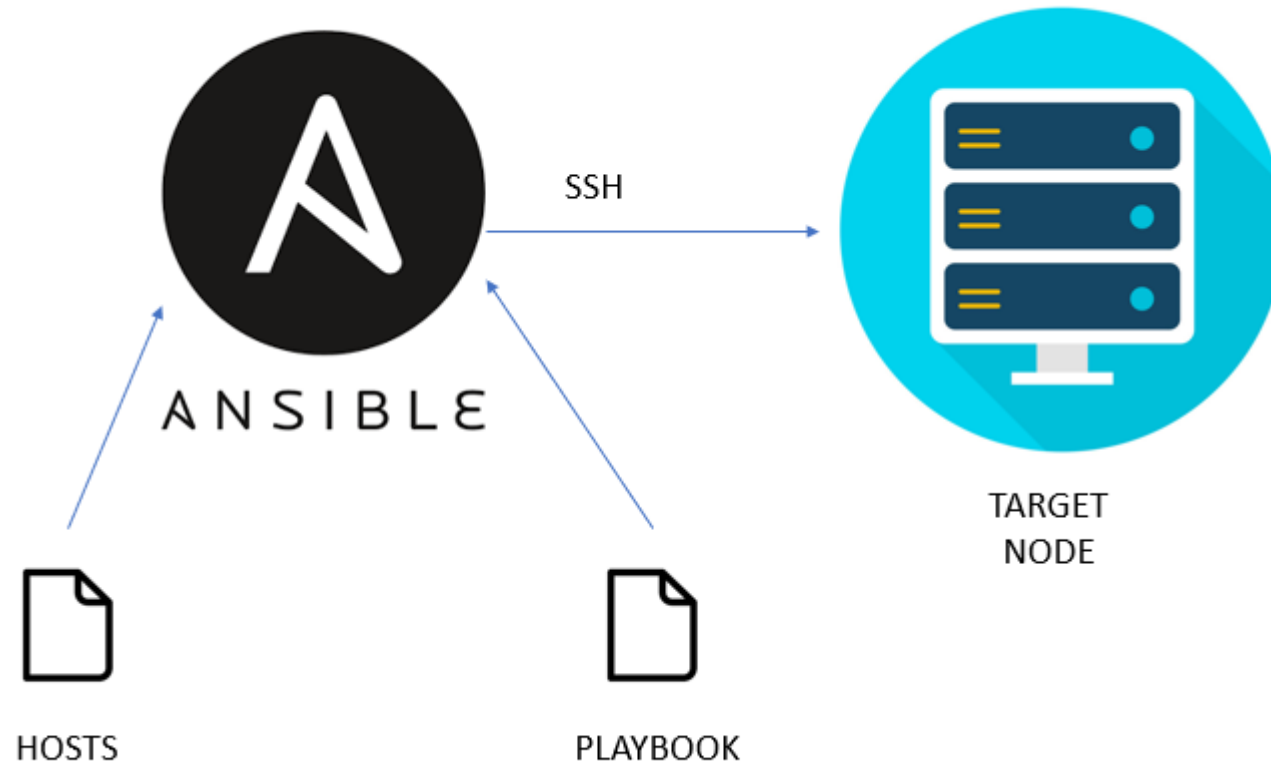
# КЛАСТЕР

- Linux Centos 7
- Java 8
- Scala 12
- Python 3
- Публичные IP /  
Внешние IP

IP адрес	Имя узла	Роли
10.0.0.2	cnt-cls-m1	NameNode, ResourceManager
10.0.0.3	cnt-cls-s1	SecondaryNameNode, DataNode, NodeManager
10.0.0.4	cnt-cls-s2	DataNode
10.0.0.5	cnt-cls-s3	DataNode



# ANSIBLE



# КЛАСТЕР

```
yum install -y net-tools openssh-server  
yum install ntp ntpdate ntp-doc -y  
yum install openssl  
yum install -y zookeeper  
yum install -y zookeeper-server
```

# КЛАСТЕР

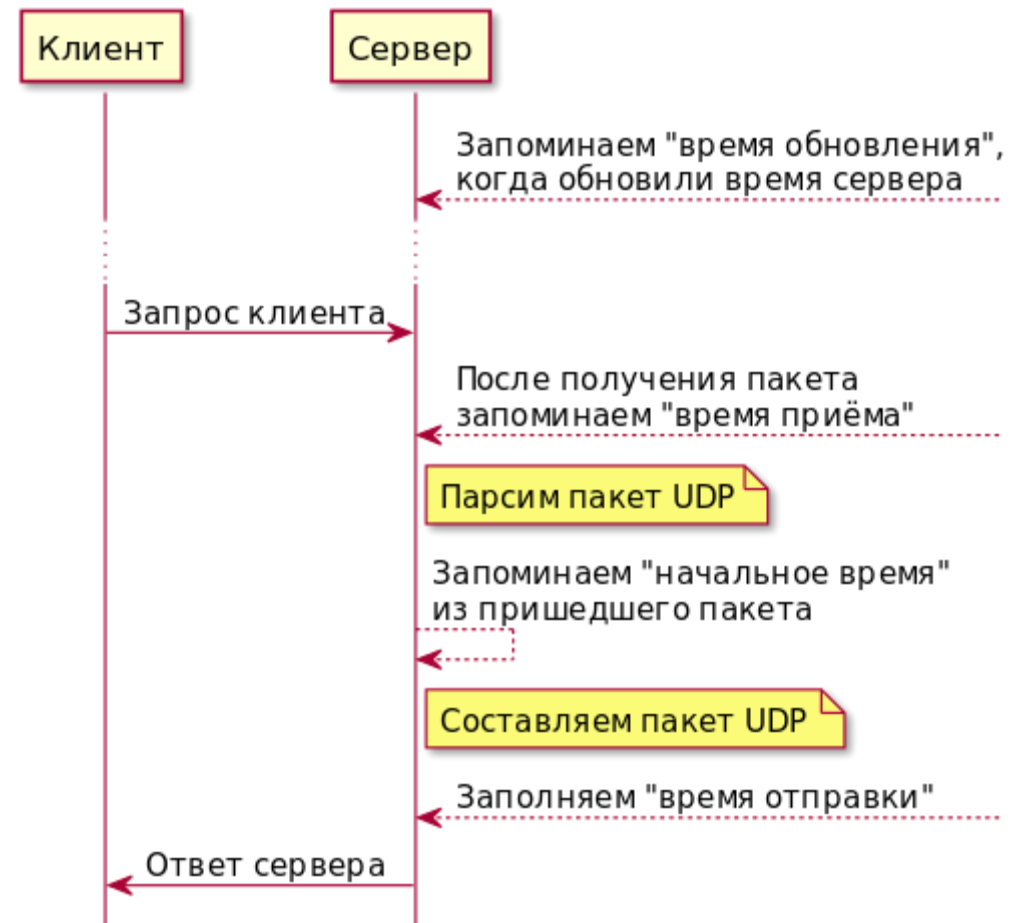
```
yum install -y net-tools openssh-server  
yum install ntp ntpdate ntp-doc -y  
yum install openssl  
yum install -y zookeeper  
yum install -y zookeeper-server
```



Clock Drift  
Clock Skew

# КЛАСТЕР

```
yum install -y net-tools openssh-server  
yum install ntp ntpdate ntp-doc -y  
yum install openssl  
yum install -y zookeeper  
yum install -y zookeeper-server
```



# КЛАСТЕР

## SSH KEY

```
ssh-keygen -t rsa -b 4096  
ssh-copy-id *имя узла*
```

## ADD USER

```
sudo groupadd hadoop  
sudo useradd -d /home/hadoop -g hadoop hadoop  
sudo passwd hadoop
```

# HADOOP

## Download

Hadoop is released as source code tarballs with corresponding binary tarballs for convenience. The downloads are distributed via mirror sites and should be checked for tampering using GPG or SHA-512.

Version	Release date	Source download	Binary download	Release notes
3.1.4	2020 Aug 3	<a href="#">source (checksum signature)</a>	<a href="#">binary (checksum signature)</a>	<a href="#">Announcement</a>
3.3.0	2020 Jul 14	<a href="#">source (checksum signature)</a>	<a href="#">binary (checksum signature)</a> <a href="#">binary-aarch64 (checksum signature)</a>	<a href="#">Announcement</a>
2.10.0	2019 Oct 29	<a href="#">source (checksum signature)</a>	<a href="#">binary (checksum signature)</a>	<a href="#">Announcement</a>
3.2.1	2019 Sep 22	<a href="#">source (checksum signature)</a>	<a href="#">binary (checksum signature)</a>	<a href="#">Announcement</a>
2.9.2	2018 Nov 19	<a href="#">source (checksum signature)</a>	<a href="#">binary (checksum signature)</a>	<a href="#">Announcement</a>

# HADOOP

```
Sep  2 15:50 capacity-scheduler.xml
Sep  2 15:49 configuration.xsl
Sep  2 15:50 container-executor.cfg
Sep  2 15:49 core-site.xml
Sep  2 15:50 hadoop-env.cmd
Sep  2 15:50 hadoop-env.sh
Sep  2 15:50 hadoop-metrics2.properties
Sep  2 15:50 hadoop-metrics.properties
Sep  2 15:50 hadoop-policy.xml
Sep  2 15:50 hdfs-site.xml
Sep  2 15:50 httpfs-env.sh
Sep  2 15:50 httpfs-log4j.properties
Sep  2 15:50 httpfs-signature.secret
Sep  2 15:50 httpfs-site.xml
Sep  2 15:50 kms-acls.xml
Sep  2 15:50 kms-env.sh
Sep  2 15:50 kms-log4j.properties
Sep  2 15:50 kms-site.xml
Sep  2 15:50 log4j.properties
Sep  2 15:50 mapred-env.cmd
Sep  2 15:49 mapred-env.sh
Sep  2 15:50 mapred-queues.xml.template
Sep  2 15:49 mapred-site.xml
Sep  2 15:50 mapred-site.xml.template
Sep  2 15:49 masters
Sep  4 20:12 slaves
Sep  2 15:50 ssl-client.xml.example
Sep  2 15:50 ssl-server.xml.example
Sep  2 15:49 yarn-env.cmd
Sep  2 15:50 yarn-env.sh
Sep  2 15:50 yarn-site.xml
```

- core-site.xml
- hdfs-site.xml
- mapred-site.xml
- yarn-site.xml

# HADOOP

## ADD DIRS

```
mkdir -p $HADOOP_HOME/tmp  
mkdir -p $HADOOP_HOME/hdfs/name  
mkdir -p $HADOOP_HOME/hdfs/data
```

## COPY SETTINGS

```
scp ~/.bashrc cnt-cls-m2:~/ #для всех 2, 3, 4)  
scp -r /opt/hadoop3/etc/hadoop/ cnt-cls-m2:/opt/hadoop3/etc/ #для всех 2, 3, 4)
```



# HADOOP

## ADD HOSTS

```
cnt-cls-m1> $HADOOP_HOME/etc/hadoop/workers  
cnt-cls-m2> $HADOOP_HOME/etc/hadoop/workers  
cnt-cls-m3> $HADOOP_HOME/etc/hadoop/workers  
cnt-cls-m4> $HADOOP_HOME/etc/hadoop/workers
```

# HADOOP

```
export HADOOP_HOME=/opt/hadoop3
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_ROOT_LOGGER=INFO,console
export HADOOP_SECURITY_LOGGER=INFO,NullAppender
export HADOOP_INSTALL=$HADOOP_HOME
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_PREFIX=$HADOOP_HOME
export HADOOP_LIBEXEC_DIR=$HADOOP_HOME/libexec
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native:$JAVA_LIBRARY_PATH
export HADOOP_YARN_HOME=$HADOOP_HOME
```

# HADOOP

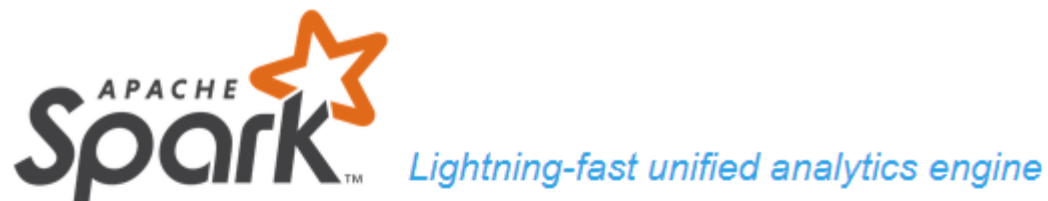
## ПЕРВЫЙ ЗАПУСК

```
hdfs namenode -format
```

```
start-dfs.sh
```

```
start-yarn.sh
```

# SPARK



[Download](#) [Libraries ▾](#) [Documentation ▾](#) [Examples](#) [Community ▾](#) [Developers ▾](#)

## Download Apache Spark™

1. Choose a Spark release:
2. Choose a package type:
3. Download Spark: [spark-3.0.0-bin-hadoop3.tgz](#)
4. Verify this release using [SHA-256 checksums](#) or [PGP signatures](#). Please [see KEYS](#).

Note that, Spark 2.x is pre-built with Scala 2.11 except version 2.4.2, which is pre-built with Scala 2.12. Spark 3.0+ is pre-built with Scala 2.12.

```
pip3 install pyspark
pip3 install py4j
```

# SPARK

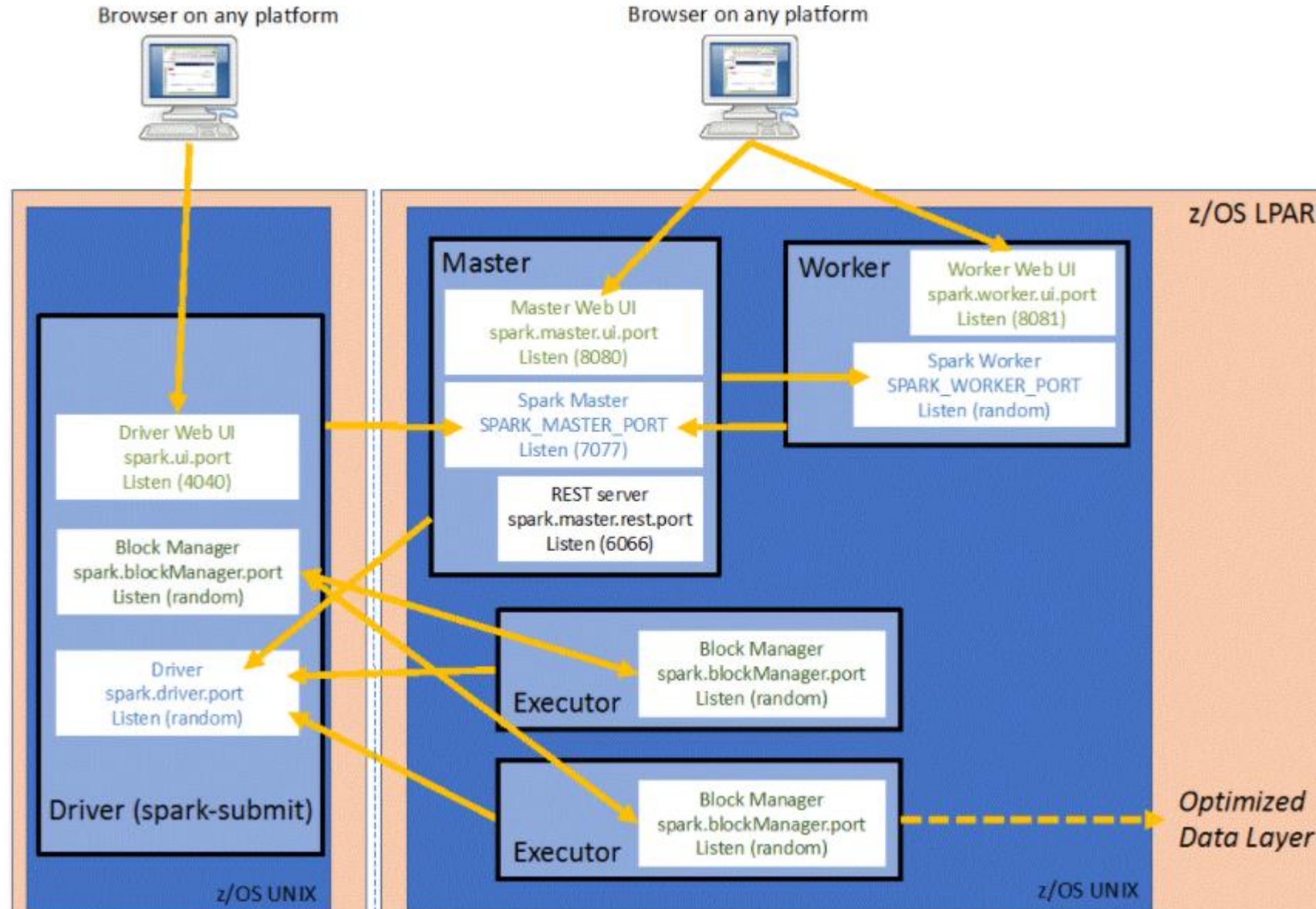
**/opt/spark3/conf/spark-env.sh**

```
SPARK_LOCAL_IP=cnt-cls-m1
SPARK_MASTER_IP=pub-cnt-cls-m1
SPARK_MASTER_HOST=cnt-cls-m1
SPARK_MASTER_PORT=7070
PYSPARK_PYTHON=/usr/bin/python3
PYSPARK_DRIVER_PYTHON=/usr/bin/python3
```

# SPARK

`/opt/spark3/conf/spark-env.sh`

```
SPARK_LOCAL_IP=cnt-cls-m1
SPARK_MASTER_IP=pub-cnt-cls-m1
SPARK_MASTER_HOST=cnt-cls-m1
SPARK_MASTER_PORT=7070
PYSPARK_PYTHON=/usr/bin/python3
PYSPARK_DRIVER_PYTHON=/usr/bin/python3
```



# SPARK

- `start-master.sh`
- `start-slave.sh spark://cnt-cls-m1:7070` (выполнить на каждой ноде)

# DRILL

The banner features a blue background with a geometric pattern. On the left, the text 'Apache Drill' is prominently displayed in white, followed by the subtitle 'Schema-free SQL Query Engine for Hadoop, NoSQL and Cloud Storage'. Below this is a white button with the text 'DOWNLOAD NOW'. On the right, there is a video player showing a man pointing at a whiteboard with a diagram. The video title 'Deployment Options and BI Tools' is visible below the player. Navigation arrows are present on either side of the video player. Below the banner, a yellow section contains links for 'Learning Apache Drill', 'News: Drill 1.17 Released (Bridget Bevans)', and 'Drill User Meetup 2019 (Bridget Bevans)'.

**Apache Drill**  
Schema-free SQL Query Engine for Hadoop, NoSQL and Cloud Storage

[DOWNLOAD NOW](#)

[Learning Apache Drill](#)

**News:** | [Drill 1.17 Released](#) (Bridget Bevans) | [Drill User Meetup 2019](#) (Bridget Bevans)

```
drillbit.sh --site $DRILL_SITE  
drill-on-yarn.sh --site $DRILL_SITE
```

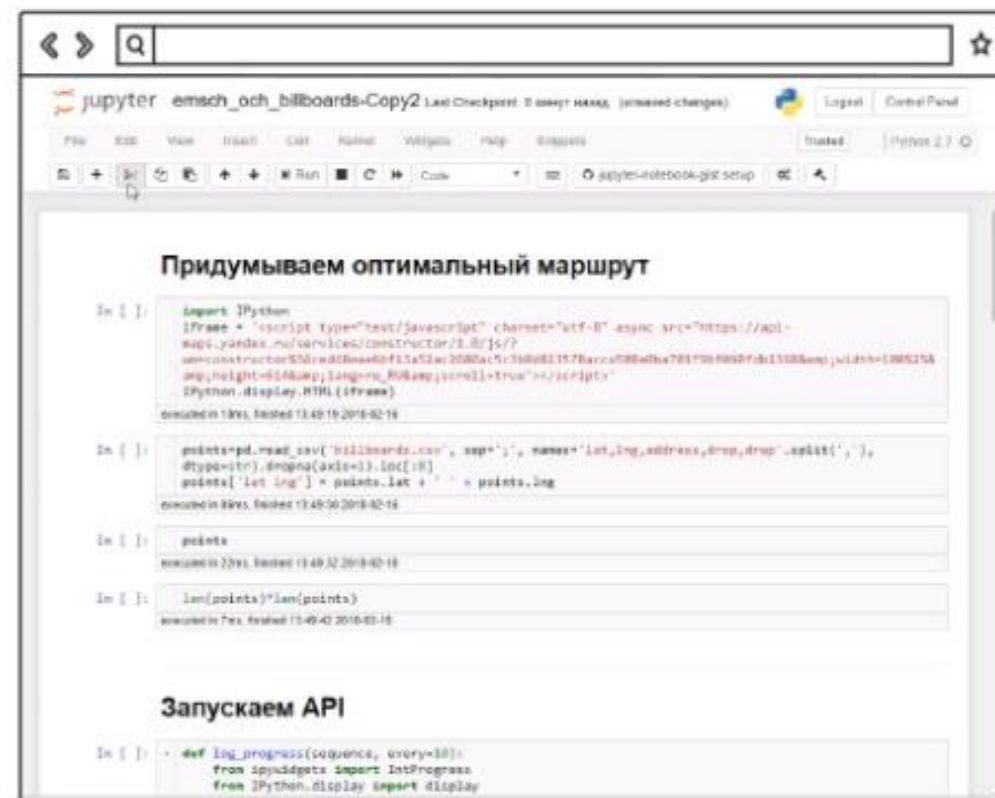


# JUPYTERHUB

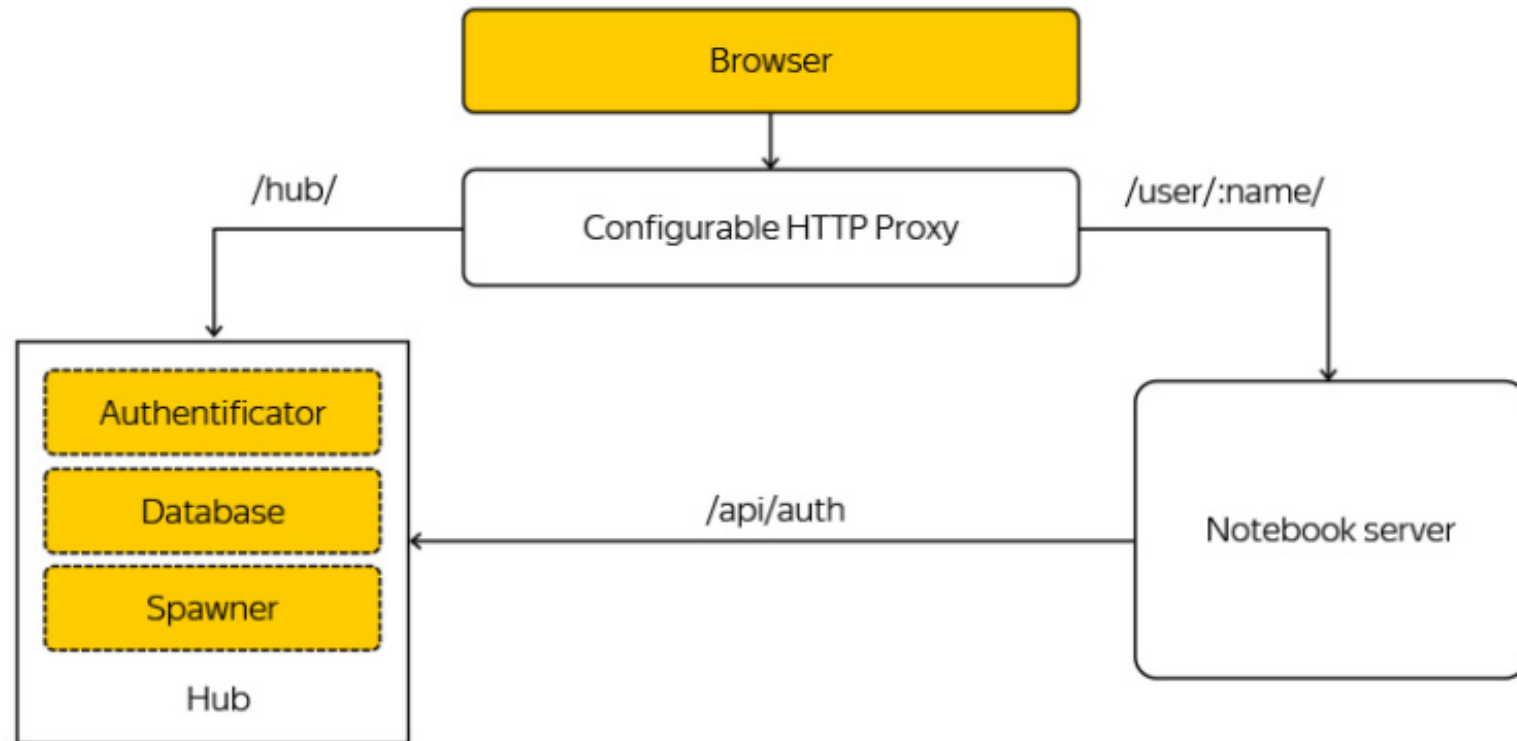


```
yum install install npm nodejs-legacy  
pip3 install jupyterhub  
npm install -g configurable-http-proxy
```

- › Классический «ноутбук»
- › Различные языки программирования
- › Интерактивный код, легко менять на лету
- › Визуализации, произвольный output



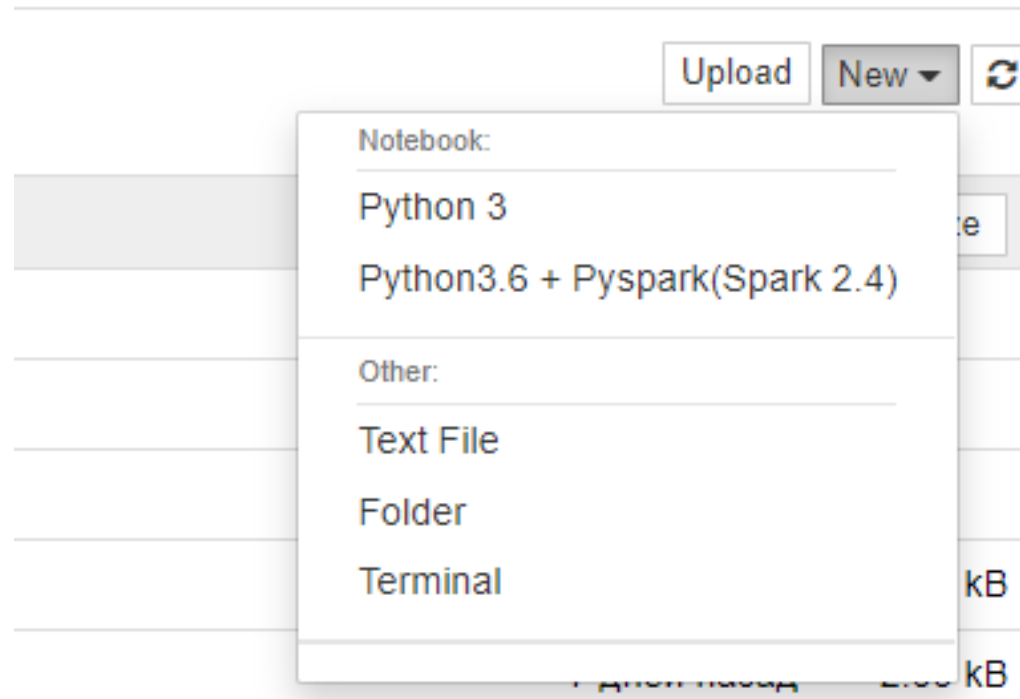
# JUPYTERHUB



# JUPYTERHUB

```
# базовый путь и публичный IP адрес для хаба
c.JupyterHub.base_url = '/'
c.JupyterHub.bind_url = 'http://pub-cnt-cls-m1:8765'
# если планируется использование более чем для 1 пользователя
c.JupyterHub.spawner_class = 'jupyterhub.spawner.SimpleLocalProcessSpawner'
c.Spawner.args = ['--allow-root', '--debug', '--profile=PHYS131']
# пользователь в linux- это пользователь в jupyterhub
c.Authenticator.admin_users = {'добавляем админов кластера',}
c.Authenticator.whitelist = {'список пользователей Linux, которые будут заходить на jupyterhub'}
# так как у нас кластер на внутренней сети, то добавляем параметр прокси
# localhost (127.0.0.1) меняем на внутреннюю сеть
c.ConfigurableHTTPProxy.api_url='http://10.0.0.2:8108'
c.JupyterHub.proxy_api_ip = '10.0.0.2'
c.JupyterHub.proxy_api_port = 5678
c.JupyterHub.hub_ip = '10.0.0.2'
c.JupyterHub.hub_port = 5678
# переменные среды для spark окружения в jupyterhub
c.YarnSpawner.environment = {
    'PYTHONPATH': 'opt/spark3/python',
    'SPARK_CONF_DIR': '/opt/spark3/conf'
}
```

# JUPYTERHUB KERNEL



# JUPYTERHUB KERNEL

**/usr/share/jupyter/kernels/**

```
{
  "argv": [
    "python3.6",
    "-m",
    "ipykernel_launcher",
    "-f",
    "{connection_file}"
  ],
  "display_name": "Python3.6 + Pyspark(Spark 3.0)",
  "language": "python",
  "env": {
    "PYSPARK_PYTHON": "/usr/bin/python3.6",
    "SPARK_HOME": "/opt/spark3",
    "HADOOP_CONF_DIR": "/etc/spark3/conf/yarn-conf",
    "HADOOP_CLIENT_OPTS": "-Xmx2147483648 -XX:MaxPermSize=512M -Djava.net.preferIPv4Stack=true",
    "PYTHONPATH": "/opt/spark3/python/lib/py4j-0.10.4-src.zip:/opt/spark3/python/",
    "PYTHONSTARTUP": "/opt/spark3/python/pyspark/shell.py",
    "PYSPARK_SUBMIT_ARGS": " --master yarn --deploy-mode client pyspark-shell"
  }
}
```

---

А дальше?



Кто это?



---

Делаем DS счастливее...



# Убираем страшный Spark Config

# JUPYTERHUB KERNEL DYNAMIC

**/usr/share/jupyter/kernels/dynamicone**

```
"env": {  
  "JAVA_HOME": "/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.e17_8.x86_64",  
  "PYSPARK_PYTHON": "python3",  
  "PYSPARK_DRIVER_PYTHON": "jupyter",  
  "PYSPARK_DRIVER_PYTHON_OPTS": "notebook",  
  "SPARK_HOME": "/opt/spark3",  
  "HADOOP_CONF_DIR": "/opt/spark3/conf/yarn-conf",  
  "PYTHONPATH": "/opt/spark3/python/lib/py4j-0.10.4-src.zip:/opt/spark3/python/",  
  "PYTHONSTARTUP": "/opt/spark3/python/pyspark/shell.py",  
  "DM": "readDM(){ $(cat ~/params | grep -oP 'drmem=\\d+' | tr "=" "\\n" | grep -oP \\d+) 'g' }"  
  "EM": "readDM(){ $(cat ~/params | grep -oP 'emem=\\d+' | tr "=" "\\n" | grep -oP \\d+) 'g' }"  
  "DC": "readDM(){ $(cat ~/params | grep -oP 'dcmem=\\d+' | tr "=" "\\n" | grep -oP \\d+) }"  
  "E": "readDM(){ $(cat ~/params | grep -oP 'emem=\\d+' | tr "=" "\\n" | grep -oP \\d+) }"  
  "PYSPARK_SUBMIT_ARGS": " --master yarn --deploy-mode client  
    --driver-memory $DM  
    --executor-memory $EM  
    --driver-cores $DC  
    --executor-cores $DC  
    --num-executors $E  
    --name studtask  
    pyspark-shell"  
}
```

# JUPYTERHUB KERNEL DYNAMIC

yarn top

```
YARN top - 12:12:36, up 34d, 2:28, 0 active users, queue(s): root
NodeManager(s): 4 total, 4 active, 0 unhealthy, 0 decommissioned, 0 lost, 0 rebooted
Queue(s) Applications: 1 running, 1277 submitted, 0 pending, 1262 completed, 14 killed, 0 failed
Queue(s) Mem(GB): 0 available, 2 allocated, 0 pending, 0 reserved
Queue(s) VCores: 0 available, 1 allocated, 0 pending, 0 reserved
Queue(s) Containers: -2 allocated, -2 pending, -2 reserved
```

APPLICATIONID	USER	TYPE	QUEUE	#CONT	#RCONT	VCORES	RVCORES	MEM	RM
application_1598597016512_1669	seleznev		spark	root.users.seleznev					1

# JUPYTERHUB KERNEL DYNAMIC

**yarn application -status app\_id**

```
Application Report :  
  Application-Id : application_1598597016512_1669  
  Application-Name : autopay  
  Application-Type : SPARK  
  User : seleznev  
  Queue : root.users.seleznev  
  Start-Time : 1601466651473  
  Finish-Time : 0  
  Progress : 10%  
  State : RUNNING  
  Final-State : UNDEFINED  
  Tracking-URL : http://pklis  
  RPC Port : -1  
  AM Host :  
  Aggregate Resource Allocation : 168064457 MB-seconds, 79216 vcore-seconds  
  Log Aggregation Status : NOT_START  
  Diagnostics :
```

# JUPYTERHUB KERNEL DYNAMIC

**ResourceManager REST API**  
(curl -v -X GET -H "Content-Type: application/json")

**+**  
**Apache DRILL**

```
with tmp as
```

```
(
```

```
select flatten(t.apps.app) as col
```

```
from dfs.tmp.`restapi/data.json` t
```

```
)
```

```
select tmp.col.id
```

```
, tmp.col.`user` as `user`
```

```
, tmp.col.runningContainers as `runningContainers`
```

```
, tmp.col.allocatedMB as `allocatedMB`
```

```
, tmp.col.allocatedVCores as `allocatedVCores`
```

```
from tmp
```


```
where tmp.col.state='RUNNING'
```

```
order by tmp.col.runningContainers desc;
```

# JUPYTERHUB KERNEL DYNAMIC

```
with tmp as
(
select flatten(t.apps.app) as col
from dfs.tmp.`restapi/data.json` t
)

select tmp.col.id
      , tmp.col.`user` as `user`
      , tmp.col.runningContainers as `runningContainers`
      , tmp.col.allocatedMB as `allocatedMB`
      , tmp.col.allocatedVCores as `allocatedVCores`
from tmp
where tmp.col.state='RUNNING'
order by tmp.col.runningContainers desc;
```



EXPR\$0	user	runningContainers	allocatedMB	allocatedVCores
application_1475192050844_0003	mapr	4	16384	4
application_1475192050844_0004	mapr	1	2048	1

---

Учимся удивлять  
Делаем умное хранилище

# HOPSWORKS



<https://github.com/logicalclocks/hopsworks>



# HOPSWORKS

## Регистрируем

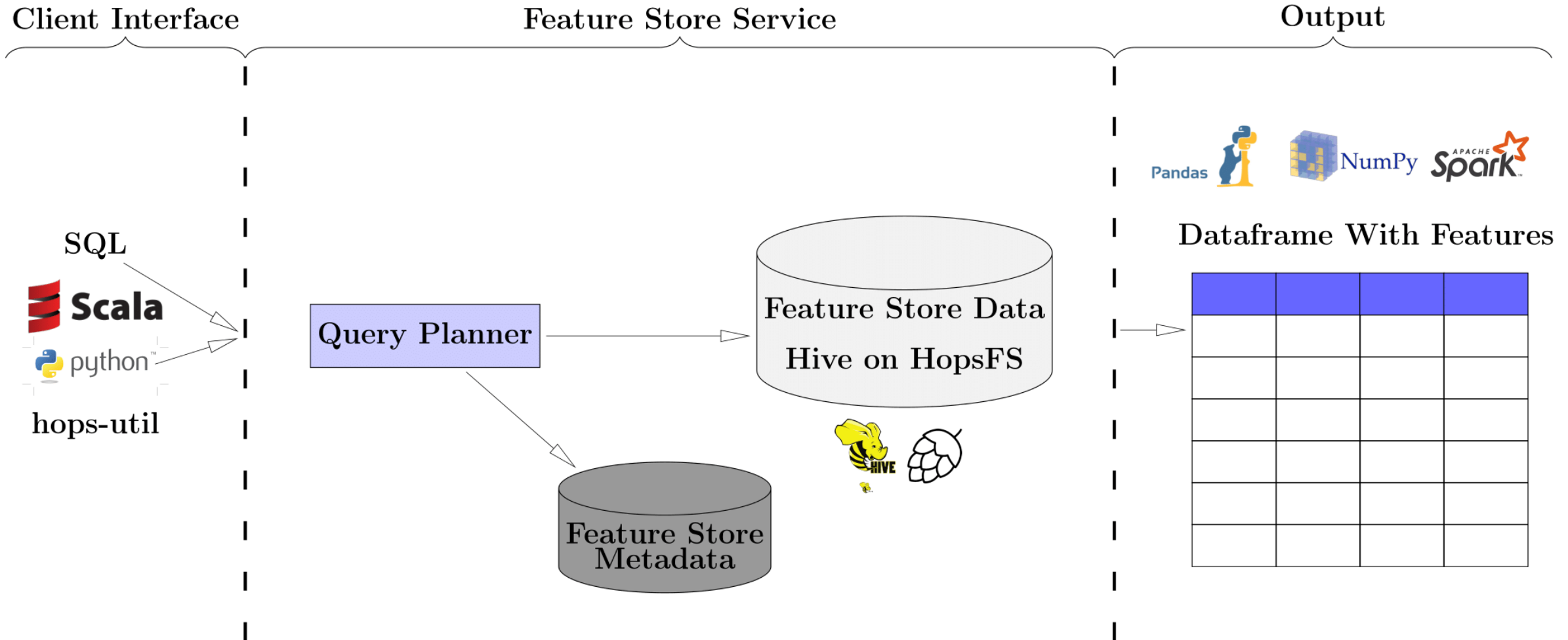
```
from hops import featurestore  
featurestore.create_featuregroup(features_df, featuregroup_name)
```

```
from hops import featurestore  
featurestore.insert_into_featuregroup(features_df, featuregroup_name)
```

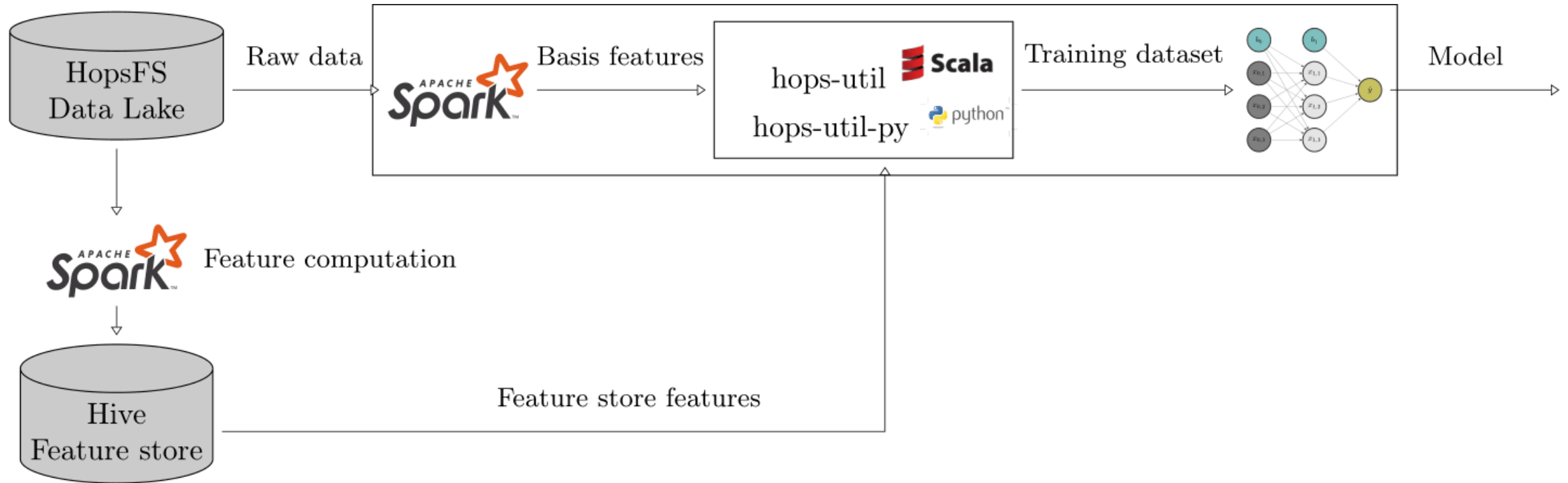
## Используем

```
from hops import featurestore  
features_df = featurestore.get_features(["average_attendance", "average_player_age"])
```

# HOPSWORKS



# HOPSWORKS





[https://github.com/NameArtem/apache\\_cluster](https://github.com/NameArtem/apache_cluster)

ВОПРОСЫ -> tg: @SeleznevArtem

tg: @SeleznevArtem



/NameArtem



/seleznev-artem



/seleznev.artem.info