CASSANDRA SUMMIT2016

Advanced DSE analytics client configuration

Jacek Lewandowski
Software engineers in DataStax Analytics Team
jacek.lewandowski@datastax.com

September 8, 2016

1 DSE configures client applications automatically

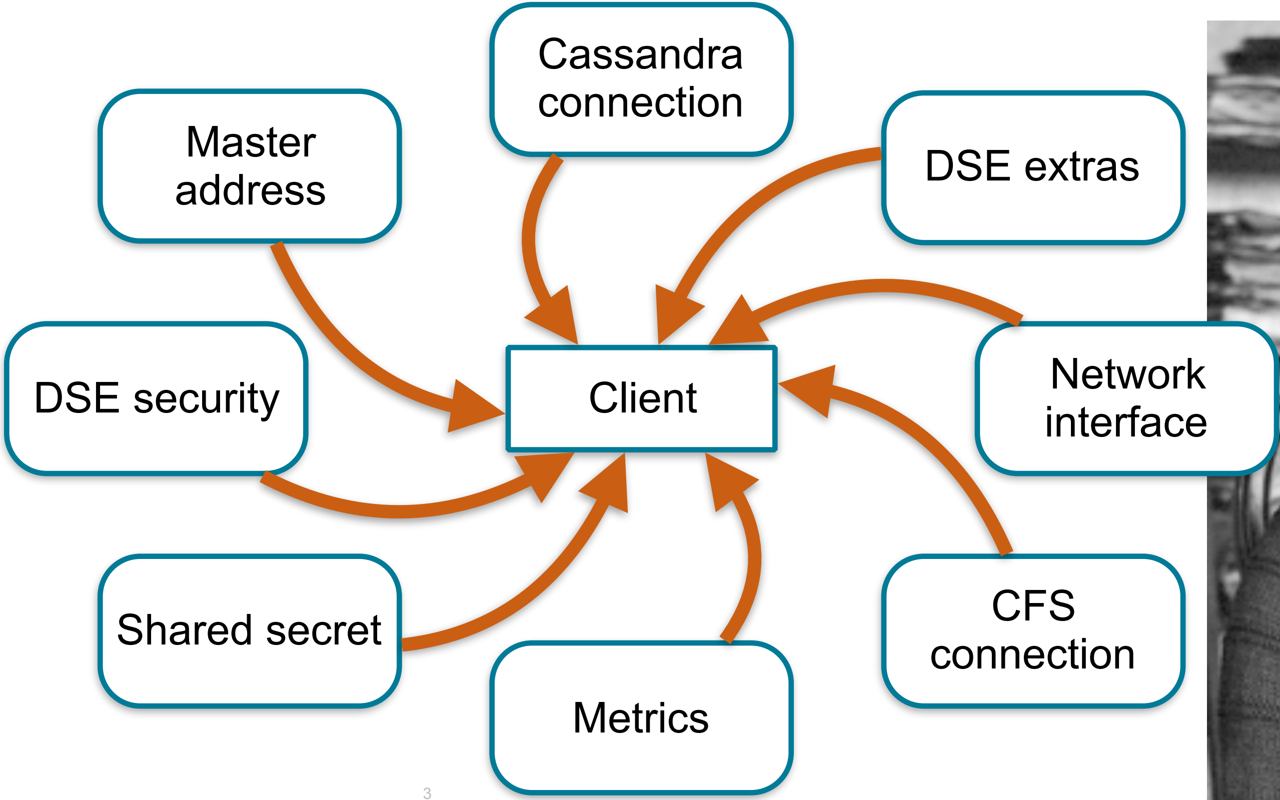
2 DSE supports Spark HA, even tricky cases

3 DSE brings useful improvements to Spark

4 DSE extends usability of Spark shared secret security

5 No more manual configuration of remote DSE nodes!

How would you configure your client manually?



Cassandra connection

Master address

DSE extras

DSE security

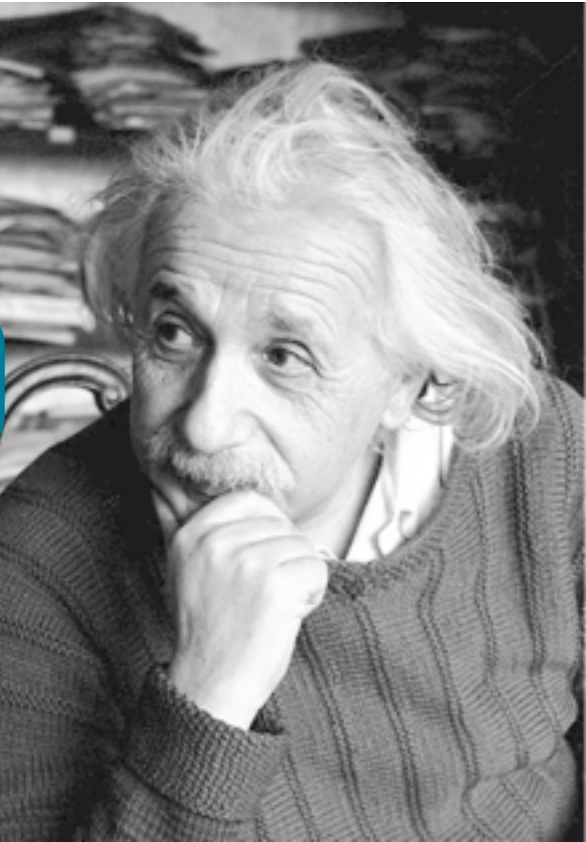
Client

Network interface

Shared secret

Metrics

CFS connection



No worries, you are using DataStax Enterprise
It does the job for you

Spark	`$ dse spark`
Hive	`$ dse hive`
Hadoop	`$ dse hadoop`
Sqoop	`$ dse sqoop`
Pig	`$ dse pig`
Client-tool	`$ dse client-tool`

No need to provide Cassandra hosts or ports

No need to configure security except providing credentials

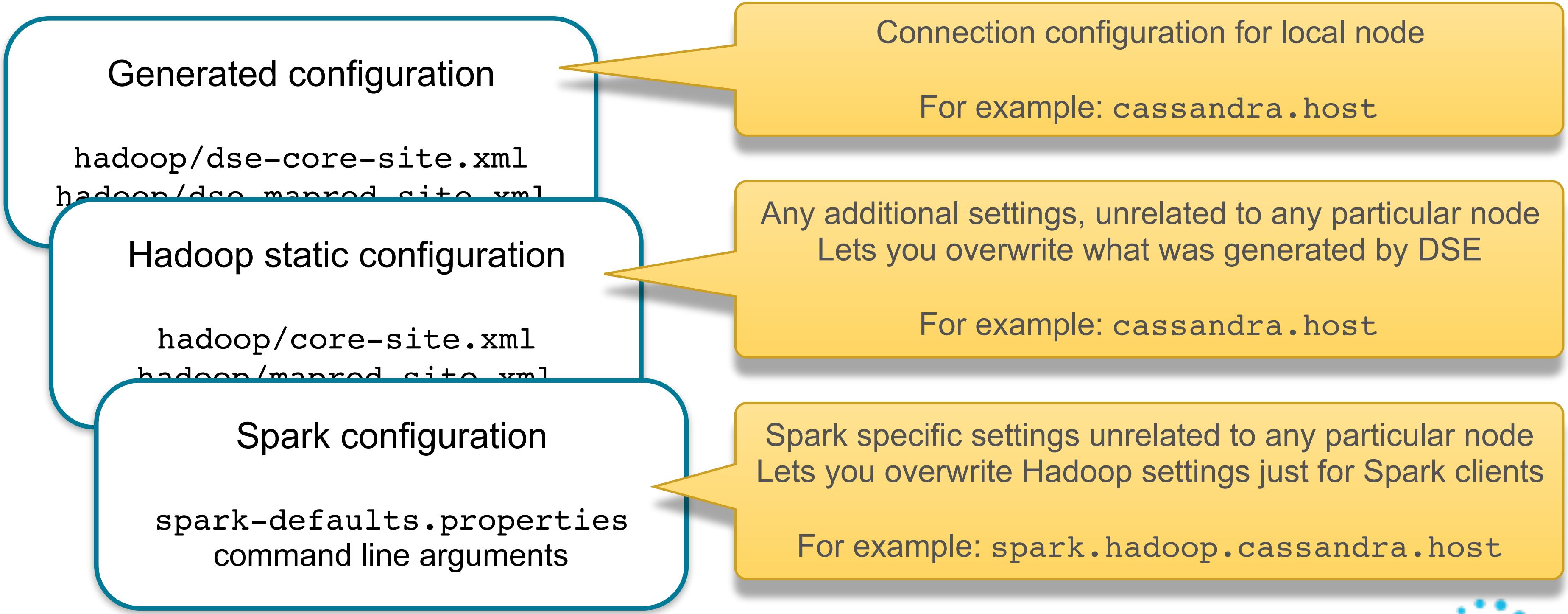
No need to specify Spark Master address

No need to specify JobTracker address

It's that simple!

No need to provide any additional configuration to start working

DataStax Enterprise generates the default configuration and lets the experts to overwrite it



Generated configuration

```
hadoop/dse-core-site.xml
hadoop/dse-mapred-site.xml
```

Connection configuration for local node

For example: `cassandra.host`

Hadoop static configuration

```
hadoop/core-site.xml
hadoop/mapred-site.xml
```

Any additional settings, unrelated to any particular node
Lets you overwrite what was generated by DSE

For example: `cassandra.host`

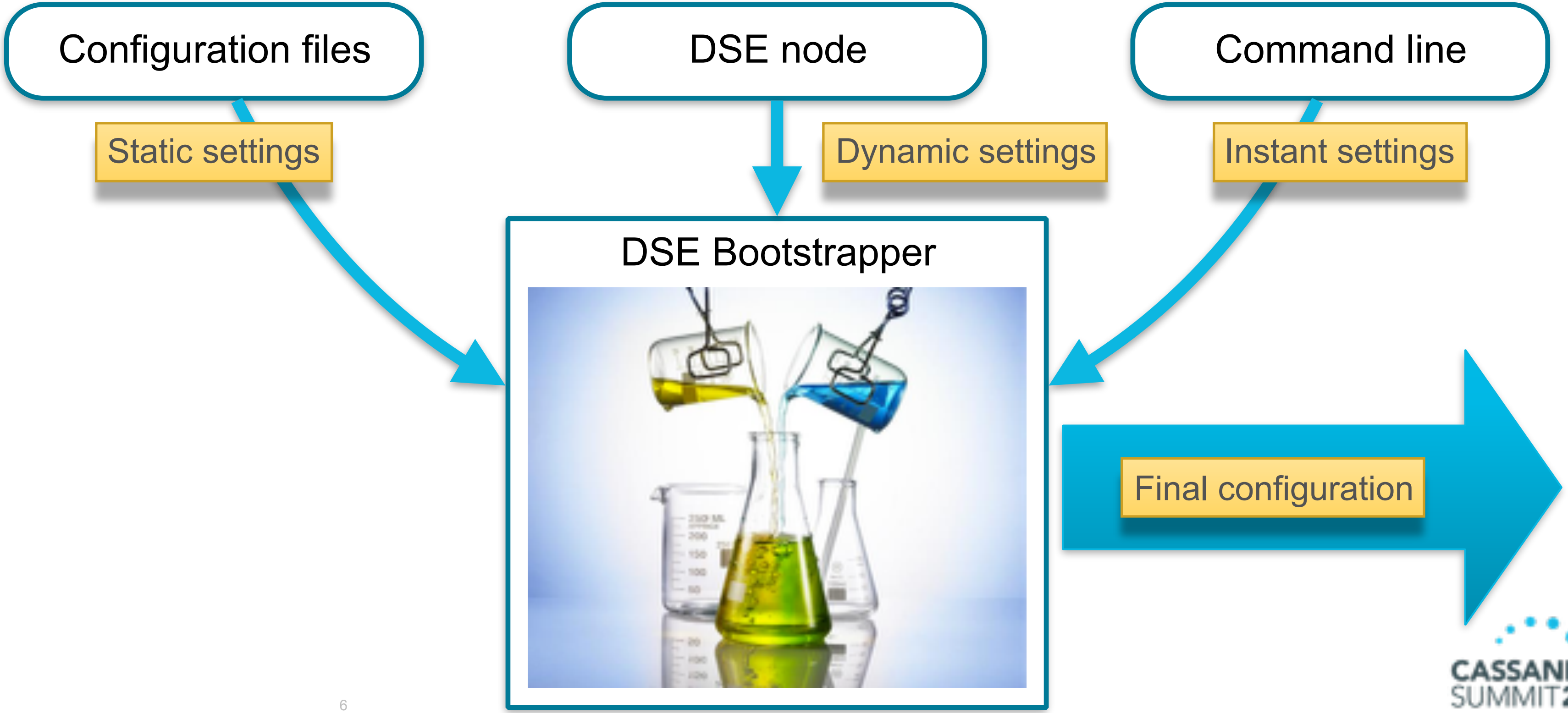
Spark configuration

`spark-defaults.properties`
command line arguments

Spark specific settings unrelated to any particular node
Lets you overwrite Hadoop settings just for Spark clients

For example: `spark.hadoop.cassandra.host`

DSE bootstrapper retrieves dynamic settings from a node and merges them with local settings



Configuration files

DSE node

Command line

Static settings

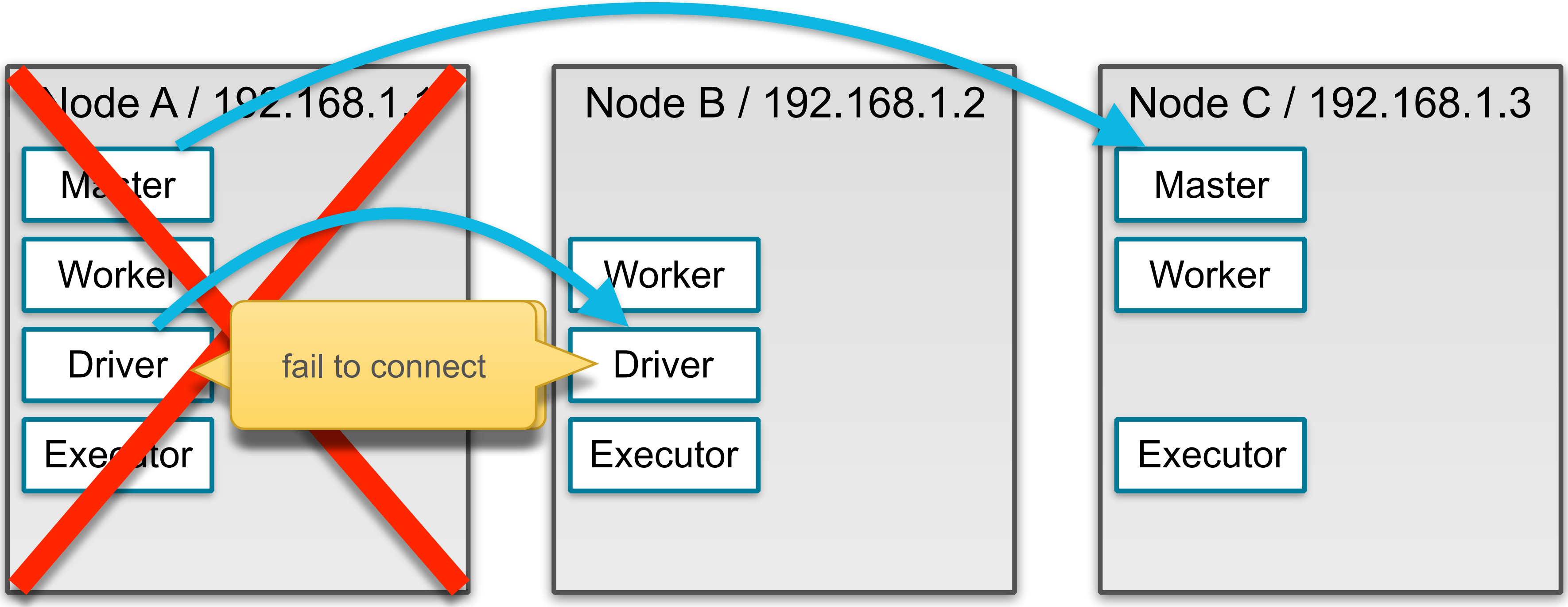
Dynamic settings

Instant settings

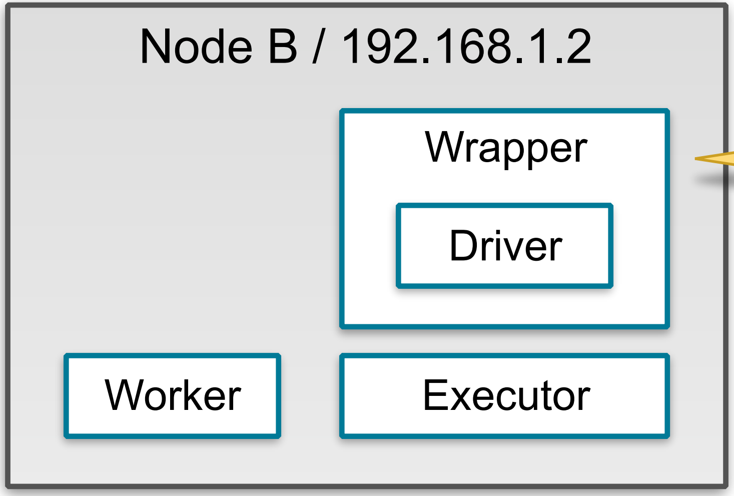
DSE Bootstrapper

Final configuration

Now, what would happen if a setting became out of date?



Whenever driver is restarted, DSE updates settings which denotes DSE nodes' addresses



Wrapper updates address which may change

```
spark.hadoop.cassandra.host
spark.cassandra.connection.host
spark.hadoop.fs.default.name
spark.hadoop.fs.defaultFS

spark.master
```

These properties may become outdated

Now, let’s talk about confidentiality of settings
It is not guaranteed for command line arguments

```
$ sh test.sh --password=qaz123
```

Any user in the system can see this

```
$ ps
  PID TTY           TIME CMD
 8640 ttys000    0:00.03 -bash
 8672 ttys000    0:00.00 sh test.sh --password=qaz123
 8673 ttys000    0:00.00 sleep 60
 8641 ttys001    0:00.01 -bash
```

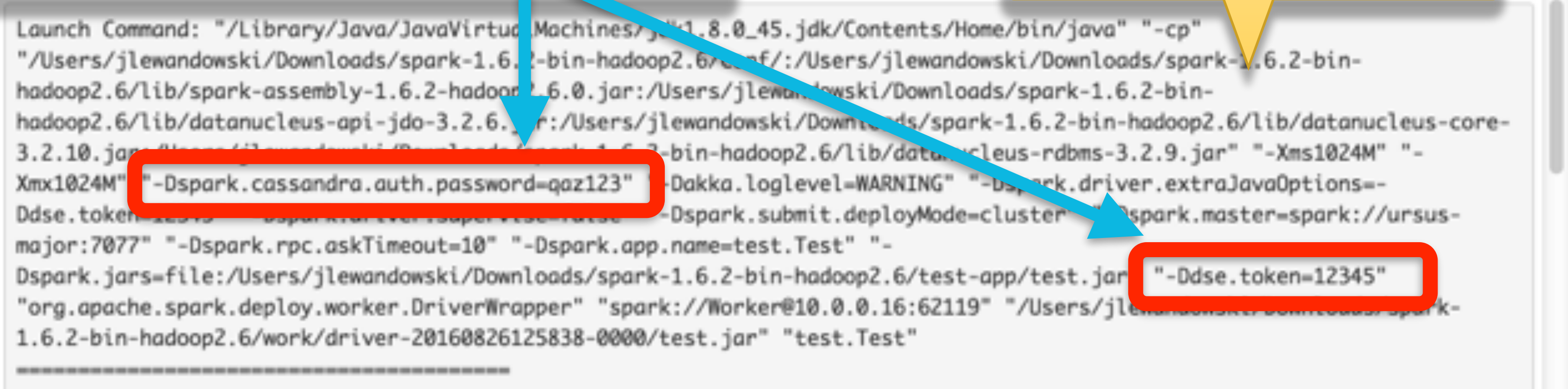
CASSANDRA
SUMMIT2016

This problem surfaces in Spark as well :-(

```
$ bin/spark-submit --deploy-mode cluster --master spark://ursus-major:7077 \
                   --properties-file test.properties \
                   --class test.Test test-app/test.jar
```

```
spark.driver.extraJavaOptions     = -Ddse.token=12345
spark.cassandra.auth.password     = qaz123
```

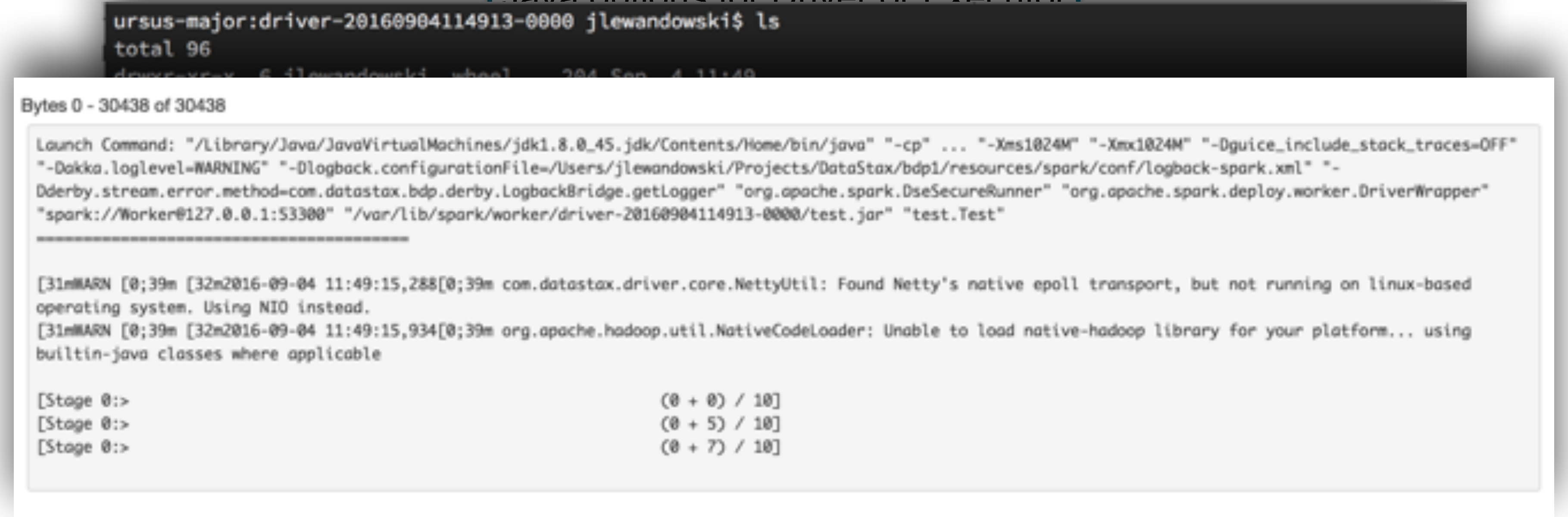
Visible in Driver logs, Worker logs, and processes list



```
Launch Command: "/Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java" "-cp"
"/Users/jlewandowski/Downloads/spark-1.6.2-bin-hadoop2.6/conf/:/Users/jlewandowski/Downloads/spark-1.6.2-bin-
hadoop2.6/lib/spark-assembly-1.6.2-hadoop2.6.0.jar:/Users/jlewandowski/Downloads/spark-1.6.2-bin-
hadoop2.6/lib/datanucleus-api-jdo-3.2.6.jar:/Users/jlewandowski/Downloads/spark-1.6.2-bin-hadoop2.6/lib/datanucleus-core-
3.2.10.jar:/Users/jlewandowski/Downloads/spark-1.6.2-bin-hadoop2.6/lib/datanucleus-rdbms-3.2.9.jar" "-Xms1024M" "-
Xmx1024M" "-Dspark.cassandra.auth.password=qaz123" "-Dakka.loglevel=WARNING" "-Dspark.driver.extraJavaOptions=-
Ddse.token=12345" "-Dspark.driver.supervise=false" "-Dspark.submit.deployMode=cluster" "-Dspark.master=spark://ursus-
major:7077" "-Dspark.rpc.askTimeout=10" "-Dspark.app.name=test.Test" "-
Dspark.jars=file:/Users/jlewandowski/Downloads/spark-1.6.2-bin-hadoop2.6/test-app/test.jar" "-Ddse.token=12345"
"org.apache.spark.deploy.worker.DriverWrapper" "spark://Worker@10.0.0.16:62119" "/Users/jlewandowski/Downloads/spark-
1.6.2-bin-hadoop2.6/work/driver-20160826125838-0000/test.jar" "test.Test"
========================================
```

Fortunately DSE Spark passes system properties in a safer way

Java options for Driver or Executor



```
ursus-major:driver-20160904114913-0000 jlewandowski$ ls
total 96
drwxr-xr-x  6 jlewandowski  wheel    204 Sep  4 11:49
```

Bytes 0 - 30438 of 30438

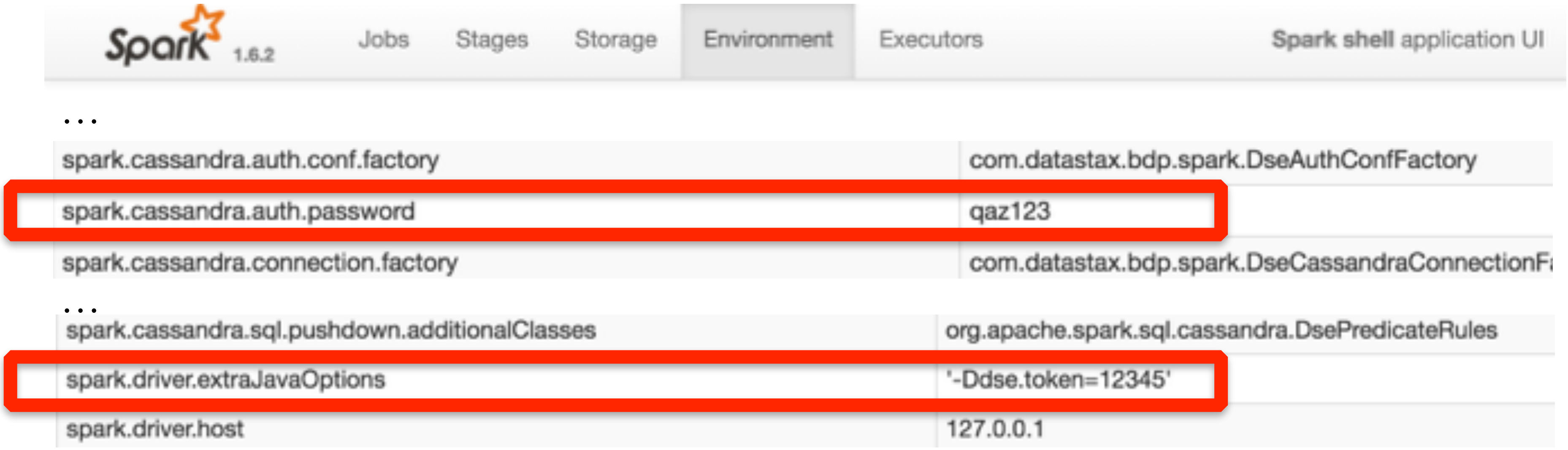
```
Launch Command: "/Library/Java/JavaVirtualMachines/jdk1.8.0_45.jdk/Contents/Home/bin/java" "-cp" ... "-Xms1024M" "-Xmx1024M" "-Dguice_include_stack_traces=OFF"
"-Dakka.loglevel=WARNING" "-Dlogback.configurationFile=/Users/jlewandowski/Projects/DataStax/bdp1/resources/spark/conf/logback-spark.xml" "-
Dderby.stream.error.method=com.datastax.bdp.derby.LogbackBridge.getLogger" "org.apache.spark.DseSecureRunner" "org.apache.spark.deploy.worker.DriverWrapper"
"spark://Worker@127.0.0.1:53300" "/var/lib/spark/worker/driver-20160904114913-0000/test.jar" "test.Test"
========================================


[31mWARN [0;39m [32m2016-09-04 11:49:15,288[0;39m com.datastax.driver.core.NettyUtil: Found Netty's native epoll transport, but not running on linux-based
operating system. Using NIO instead.
[31mWARN [0;39m [32m2016-09-04 11:49:15,934[0;39m org.apache.hadoop.util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using
builtin-java classes where applicable

[Stage 0:>                                                          (0 + 0) / 10]
[Stage 0:>                                                          (0 + 5) / 10]
[Stage 0:>                                                          (0 + 7) / 10]
```

But this is not enough — system properties and configuration are still exposed in Driver UI

```
$ bin/spark-shell --conf spark.cassandra.auth.password=qaz123 \
                  --driver-java-options=-Ddse.token=12345
```



...

spark.cassandra.auth.conf.factory	com.datastax.bdp.spark.DseAuthConfFactory
spark.cassandra.auth.password	qaz123
spark.cassandra.connection.factory	com.datastax.bdp.spark.DseCassandraConnectionF

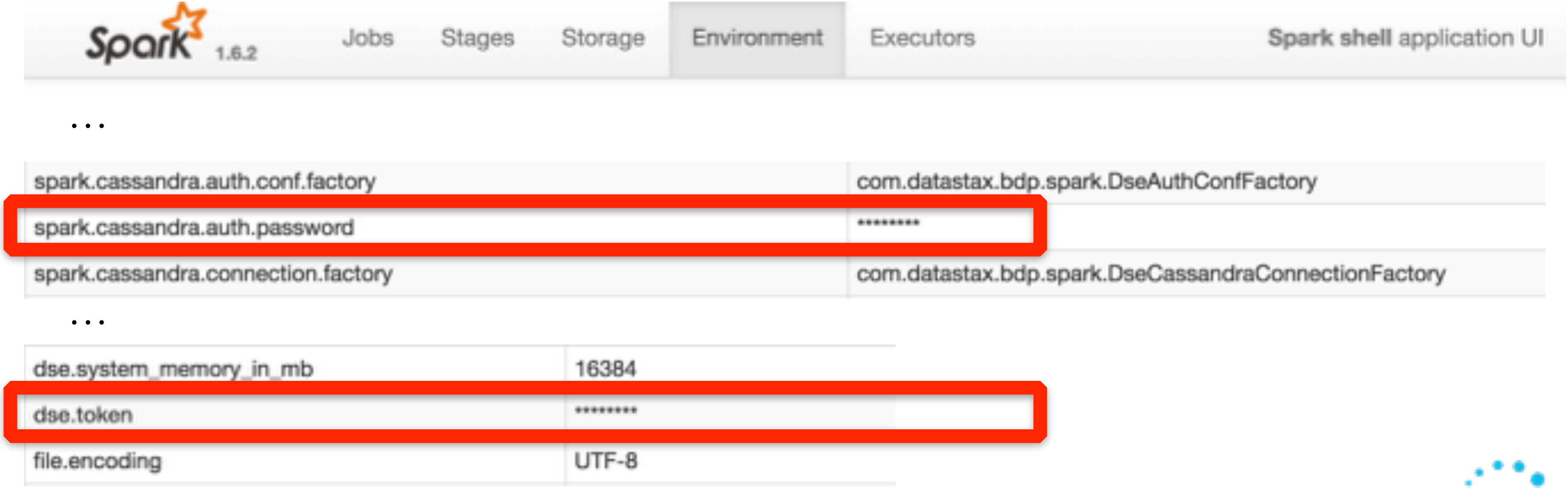
...

spark.cassandra.sql.pushdown.additionalClasses	org.apache.spark.sql.cassandra.DsePredicateRules
spark.driver.extraJavaOptions	'-Ddse.token=12345'
spark.driver.host	127.0.0.1

DSE Spark addresses this problem too!

```
spark.ui.confidentialKeys=password,token
```

Just set this property in your Spark configuration



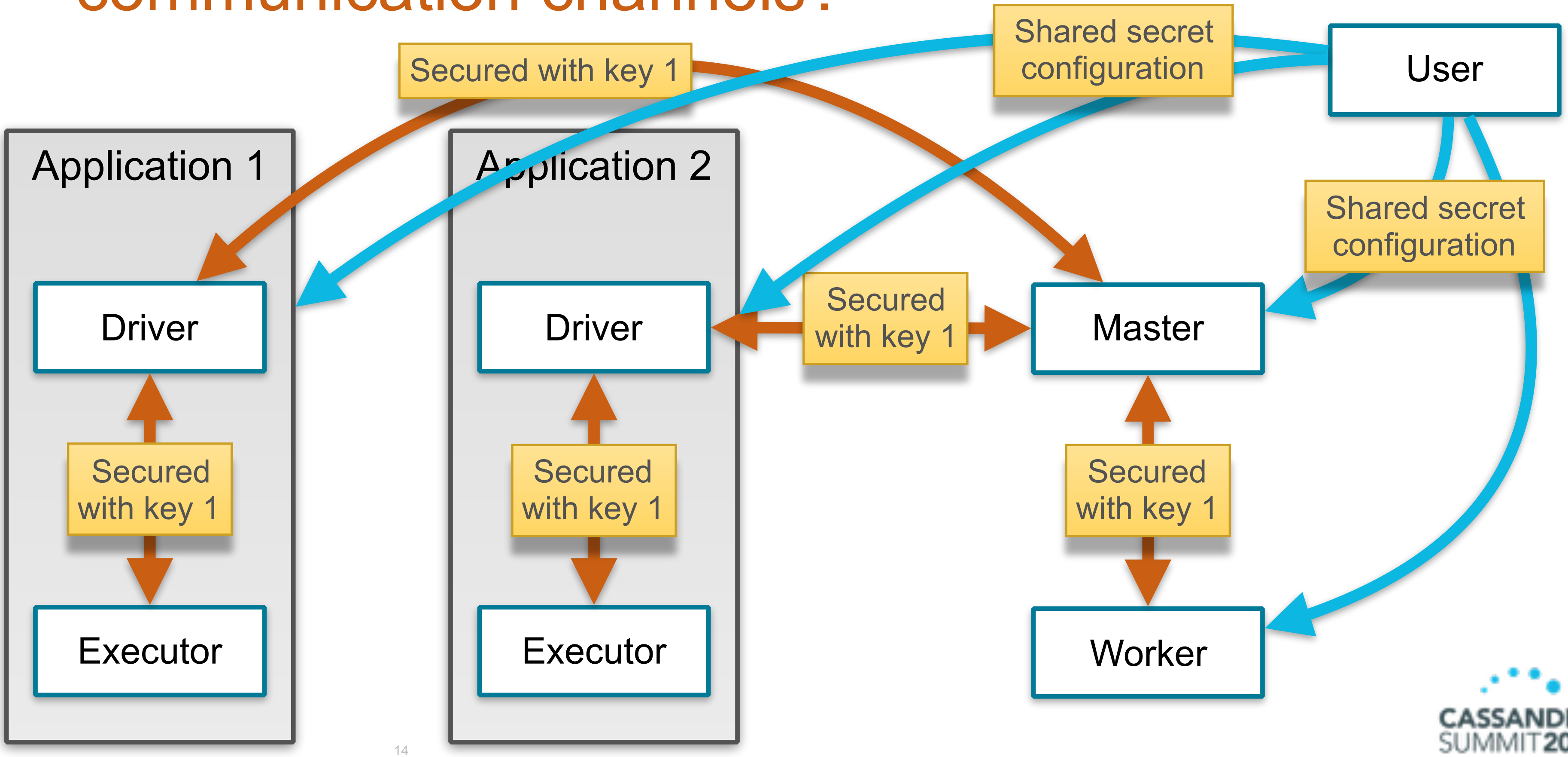
...

spark.cassandra.auth.conf.factory	com.datastax.bdp.spark.DseAuthConfFactory
spark.cassandra.auth.password	*********
spark.cassandra.connection.factory	com.datastax.bdp.spark.DseCassandraConnectionFactory

...

dse.system_memory_in_mb	16384
dse.token	*********
file.encoding	UTF-8

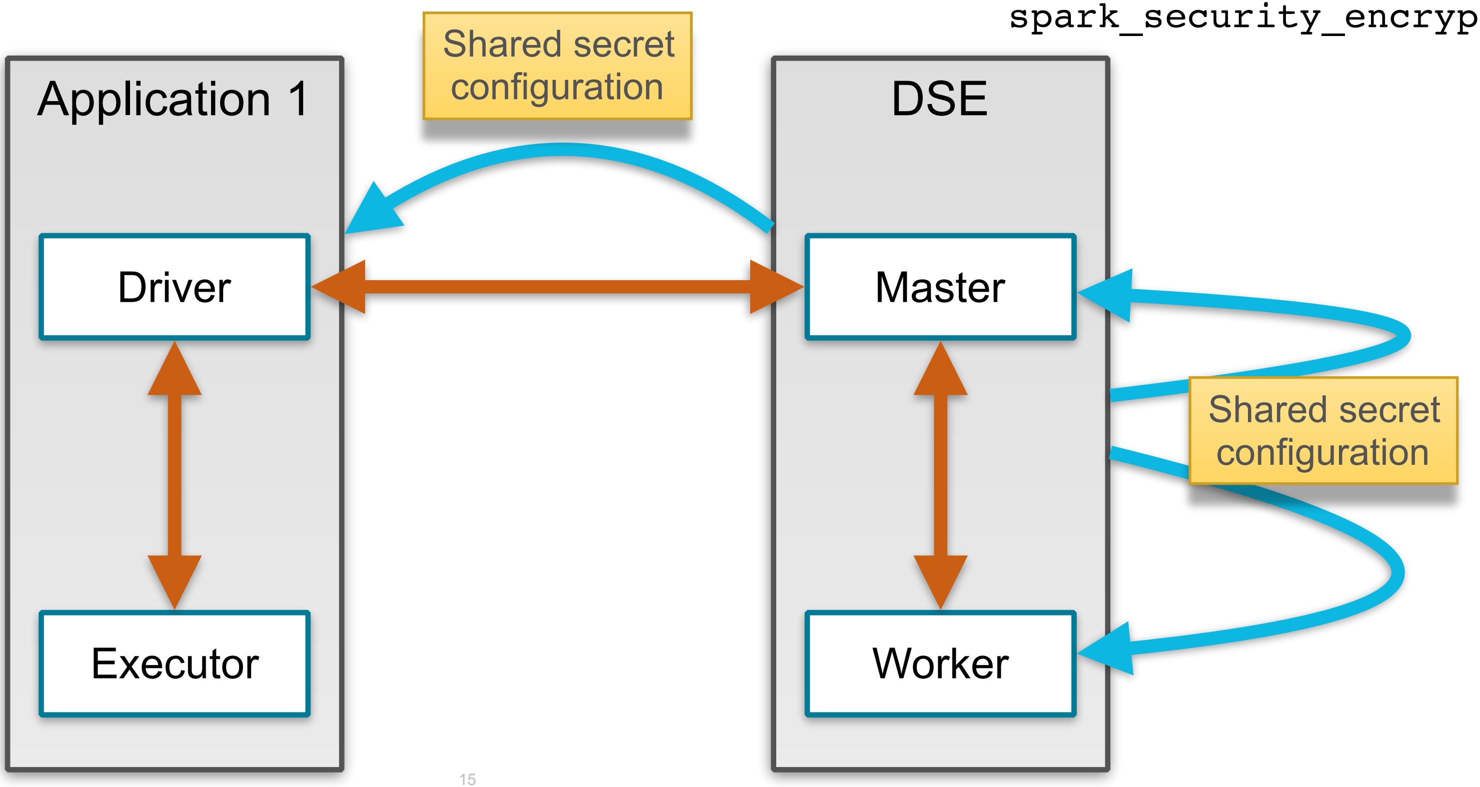
What about confidentiality of Spark communication channels?



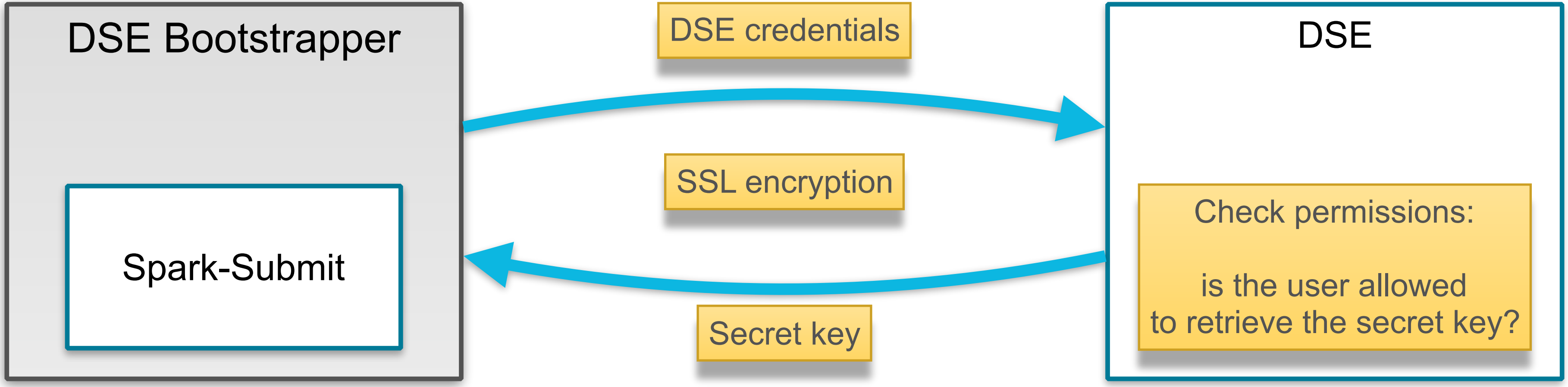
CASSANDRA SUMMIT 2016

DSE manages shared secret for Spark

```
spark_security_enabled: true
spark_security_encryption_enabled: true
```



Secret key is provided only to authenticated and authorized users



DSE Bootstrapper

Spark-Submit

DSE credentials

SSL encryption

Secret key

DSE

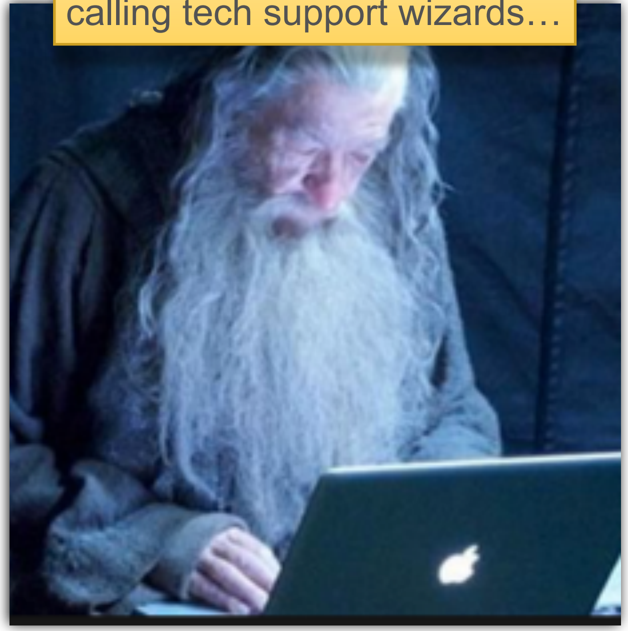
Check permissions:

is the user allowed
to retrieve the secret key?

So, you have your DSE cluster running You want to use it from you Macbook…

DSE installation on unconfigured remote node

Which usually ends up calling tech support wizards…



CASSANDRA SUMMIT2016

DSE can export configuration into a single file … and import it to any other DSE installation

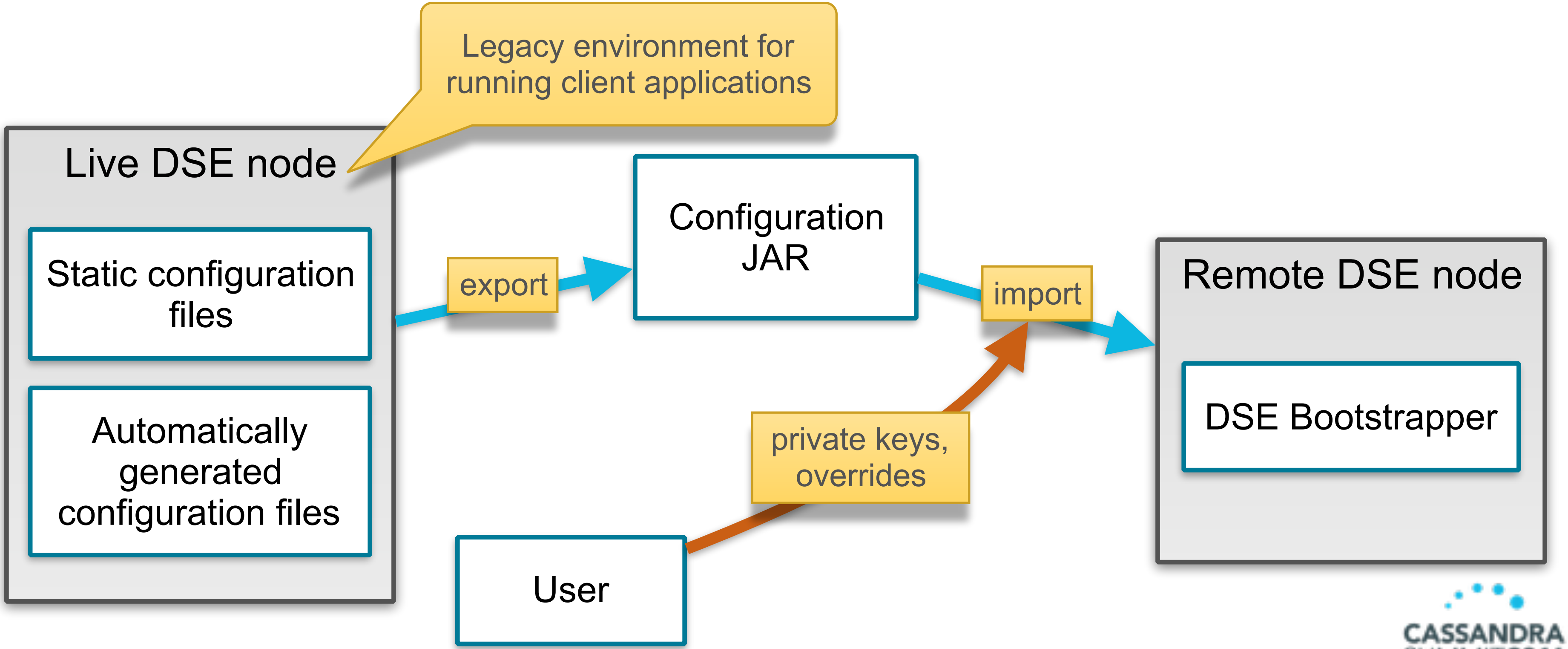
DSE installation
Local running DSE node

```
$ dse client-tool configuration export config.jar
```

DSE installation
Remote, unconfigured DSE node

```
$ dse client-tool configuration import config.jar
```

Import tool does similar job as DSE instance does on startup



Legacy environment for running client applications

Live DSE node

Static configuration files

Automatically generated configuration files

export

Configuration JAR

import

Remote DSE node

DSE Bootstrapper

private keys, overrides

User

So you may still want to call tech support…

…but probably for
a *slightly* different reason

CASSANDRA
SUMMIT2016

Thank you for your attention

May DSE be with you

- It configures your client applications automatically
- It is a provider of HA for Spark and handles even tricky problems
- It provides tools to configure DSE installations on remote nodes
- It improves confidentiality of configuration settings
- It manages distribution of shared secret automatically

Special thanks to Russell Spitzer, Rocco Varela, Piotr Kołaczkowski and the rest of DSE Analytics Team