**ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2024-2025**

**ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ**

**ΜΑΘΗΜΑ: Συστήματα Διαχείρισης Μεγάλου Όγκου Δεδομένων**

**Οδηγός Σύνδεσης και Εκτέλεσης Apache Spark στο Kubernetes του Εργαστηρίου vdcloud**

1. **Εισαγωγή**

Αυτός ο οδηγός παρέχει λεπτομερείς οδηγίες για τη σύνδεση στην υποδομή του εργαστηρίου VDCLOUD μέσω VPN, την εγκατάσταση των απαραίτητων εργαλείων, τη ρύθμιση του περιβάλλοντος εργασίας ενός τοπικού υπολογιστή και την εκτέλεση εργασιών Apache Spark στο Kubernetes[[1]](#footnote-1) (k8s).

Θα λάβετε επίσης ένα email με δυο αρχεία ρυθμίσεων και ένα username που θα έχετε στην υποδομή. Στον παρακάτω οδηγό, όπου βλέπετε το **<username>** θα το αντικαθιστάτε με το όνομα χρήστη που λάβατε στο email σας.

1. **Εγκατάσταση OpenVPN Client**

Για να συνδεθείτε στην υποδομή, εγκαταστήστε τον OpenVPN client από τον παρακάτω σύνδεσμο: [OpenVPN Community Downloads](https://openvpn.net/community-downloads/)

Αφού εγκαταστήσετε τον client, εισάγετε το αρχείο .ovpn που σας έχει αποσταλεί με e-mail και συνδεθείτε.

1. **Έλεγχος συνδεσιμότητας υποδομής από το WSL Ubuntu**

Μετά την σύνδεση με το vpn, συνήθως η σύνδεση με την υποδομή μέσω windows λειτουργεί χωρίς πρόβλημα. Σε ορισμένες εγκαταστάσεις Linux WSL μπορεί να μην ρυθμίζονται ορισμένα θέματα που αφορούν την ονοματοδοσία (DNS) αυτόματα. Ανοίξτε το WSL Linux που έχετε στήσει και τρέξτε.

ping source-code-master

Εάν **ΔΕΝ** έχει ρυθμιστεί σωστά, η εντολή δεν ολοκληρώνεται και δεν βγάζει κανένα αποτέλεσμα, καθώς δεν λειτουργεί η μετάφραση του ονόματος του server σε IP. Σε αυτή την περίπτωση κάντε το εξής.

Ανοίξτε το αρχείο ρυθμίσεων /etc/resolv.conf με την παρακάτω εντολή

sudo nano /etc/resolv.conf

Αλλάξτε την γραμμή:

nameserver 10.255.255.254

με την γραμμή:

nameserver 10.42.0.1

Επίσης, εξασφαλίστε ότι υπάρχει και το παρακάτω κείμενο:

search default.svc.cluster.local

Αποθηκεύστε το αρχείο και βγείτε (πατήστε Ctrl+X και μετά Y)

Δοκιμάστε πάλι την εντολή

ping source-code-master

και θα πρέπει να δείτε ένα αποτέλεσμα σαν το παρακάτω (τερματίστε την εκτέλεσή του μέσω Ctrl+C αλλιώς θα τρέχει συνέχεια)

ikons@ikons-desktop:~$ ping source-code-master

PING source-code-master (10.42.0.1) 56(84) bytes of data.

64 bytes from source-code-master.cluster.local (10.42.0.1): icmp\_seq=1 ttl=63 time=9.34 ms

64 bytes from source-code-master.cluster.local (10.42.0.1): icmp\_seq=2 ttl=63 time=9.33 ms

64 bytes from source-code-master.cluster.local (10.42.0.1): icmp\_seq=3 ttl=63 time=52.2 ms

64 bytes from source-code-master.cluster.local (10.42.0.1): icmp\_seq=4 ttl=63 time=10.9 ms

64 bytes from source-code-master.cluster.local (10.42.0.1): icmp\_seq=5 ttl=63 time=10.9 ms

^C

--- source-code-master ping statistics ---

5 packets transmitted, 5 received, 0% packet loss, time 4180ms

rtt min/avg/max/mdev = 9.325/18.530/52.237/16.867 ms

Δυστυχώς, θα πρέπει την ρύθμιση αυτή να την κάνετε **κάθε φορά που συνδέεστε με το VPN**.

1. **Εγκατάσταση kubectl**

Το kubectl είναι το εργαλείο γραμμής εντολών για τη διαχείριση Kubernetes clusters. Εγκαταστήστε το με τις παρακάτω εντολές σε ένα Linux μηχάνημα ή εικονική μηχανή:

# Εγκατάσταση βασικών πακέτων που απαιτούνται για πρόσβαση σε αποθετήρια HTTPS

**sudo** apt-get install -y apt-transport-https ca-certificates curl gnupg

# Λήψη και αποθήκευση του δημόσιου κλειδιού για το Kubernetes αποθετήριο

**curl** -fsSL https**://**pkgs.k8s.io**/**core**:/**stable**:/**v1.32**/**deb**/**Release.key **|** **sudo** gpg --dearmor -o **/**etc**/**apt**/**keyrings**/**kubernetes-apt-keyring.gpg

# Ρύθμιση σωστών δικαιωμάτων πρόσβασης στο κλειδί

**sudo** chmod 644 **/**etc**/**apt**/**keyrings**/**kubernetes-apt-keyring.gpg

# Προσθήκη του Kubernetes αποθετηρίου στη λίστα των sources του apt

**echo** 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.32/deb/ /' **|** **sudo** tee **/**etc**/**apt**/**sources.list.d**/**kubernetes.list

# Ρύθμιση σωστών δικαιωμάτων πρόσβασης στο αρχείο sources

**sudo** chmod 644 **/**etc**/**apt**/**sources.list.d**/**kubernetes.list

# Ενημέρωση της λίστας πακέτων του apt

**sudo** apt-get update

# Εγκατάσταση του εργαλείου kubectl

**sudo** apt-get install -y kubectl

# Δημιουργία του καταλόγου ~/.kube όπου αποθηκεύεται το αρχείο config

**mkdir** **~/.**kube

Εισάγετε το αρχείο config που σας έχει αποσταλεί με e-mail στην τοποθεσία ~/.kube/config για να μπορεί το εργαλείο kubectl να συνδεθεί με την υποδομή k8s.

Για να το κάνετε αυτό, θα πρέπει να αντιγράψετε το αρχείο config από την τοποθεσία του συστήματος αρχείων του host υπολογιστή σας που αρχικά το κατεβάσατε (δηλαδή windows) στον κατάλογο ~/.kube του Linux WSL.

Έστω ότι έχετε κατεβάσει το αρχείο config στον κατάλογο Downloads (Λήψεις) του χρήστη των windows.

Για να το αντιγράψετε στην σωστή θέση, πρέπει να εκτελέσετε τις παρακάτω εντολές στο WSL Linux. Θα αντικαταστήσετε το **<username>** με το όνομα χρήστη των windows της δικής σας εγκατάστασης. Για παράδειγμα, στην δική μου περίπτωση είναι /mnt/c/Users/**ikons**/Downloads/config

# Πηγαίνουμε στον προσωπικό κατάλογο του χρήστη

**cd**

# Δημιουργούμε τον κατάλογο .kube (αν δεν υπάρχει)

**mkdir** **.**kube

# Αντιγραφή του αρχείου config από το σύστημα αρχείων των Windows στο WSL

**cp** **/**mnt**/**c**/**Users**/<username>/**Downloads**/**config **~/.**kube**/**config

Ένας άλλος τρόπος να το κάνετε είναι μέσω του Windows explorer επιλέγοντας το Linux folder:

A screenshot of a computer

AI-generated content may be incorrect.

1. **Εγκατάσταση k9s**

Το k9s είναι ένα εργαλείο για την παρακολούθηση και τη διαχείριση Kubernetes clusters. Εγκαταστήστε το ως εξής:

# Λήψη του .deb πακέτου του k9s από το GitHub

**wget** https**://**github.com**/**derailed**/**k9s**/**releases**/**download**/**v0.40.10**/**k9s\_linux\_amd64.deb

# Εγκατάσταση του πακέτου k9s μέσω του dpkg

**sudo** dpkg -i k9s\_linux\_amd64.deb

# Ορισμός του nano ως προεπιλεγμένου editor για το k9s (και γενικά για το kubectl)

**echo** "export KUBE\_EDITOR=nano" **>>** **~/.**bashrc

Το εργαλείο k9s χρησιμοποιεί και αυτό το αρχείο ρυθμίσεων ~/.kube/config.

1. **Εγκατάσταση Hadoop και Spark clients**

Εγκαταστήστε το Hadoop και το Spark για τη διαχείριση δεδομένων και την εκτέλεση αναλύσεων τοπικά στον υπολογιστή σας, για να μπορέσετε να συνδεθείτε με την απομακρυσμένη υποδομή. Εγκαταστήστε το με τις παρακάτω εντολές σε ένα Linux μηχάνημα ή εικονική μηχανή:

# Μεταφορά στον προσωπικό κατάλογο του χρήστη (home directory)

**cd** **~**

# Εγκατάσταση του Java Development Kit 8 (απαραίτητο για Hadoop/Spark)

**sudo** apt-get install -y openjdk-8-jdk

# Λήψη του Spark (έκδοση 3.5.5 με υποστήριξη για Hadoop 3)

**wget** https**://**downloads.apache.org**/**spark**/**spark-3.5.5**/**spark-3.5.5-bin-hadoop3.tgz

# Αποσυμπίεση του αρχείου Spark

**tar** -xzf spark-3.5.5-bin-hadoop3.tgz

# Λήψη του Hadoop (έκδοση 3.4.1)

**wget** https**://**dlcdn.apache.org**/**hadoop**/**common**/**hadoop-3.4.1**/**hadoop-3.4.1.tar.gz

# Αποσυμπίεση του αρχείου Hadoop

**tar** -xzf hadoop-3.4.1.tar.gz

Προσθέστε τις παρακάτω μεταβλητές περιβάλλοντος στο αρχείο ρυθμίσεων περιβάλλοντος του χρήστη σας. Όπου **<username>** θα το αλλάξετε με το δικό σας. Στην δική μου περίπτωση είναι **export** HADOOP\_USER\_NAME**=ikons**

**export** JAVA\_HOME**=/**usr**/**lib**/**jvm**/**java-8-openjdk-amd64

**export** SPARK\_HOME=**$HOME/**spark-3.5.5-bin-hadoop3

**export** PATH**=$HOME/**spark-3.5.5-bin-hadoop3**/**bin**:$HOME/**hadoop-3.4.1**/**bin**:$PATH**

**export** HADOOP\_USER\_NAME**=<username>**

Αποθηκεύστε το αρχείο (Crtl+X και Y) και ENTER

Τέλος, εκτελέστε την παρακάτω εντολή για να φορτωθούν οι νέες ρυθμίσεις στο περιβάλλον:

. ~/.bashrc

1. **Ρύθμιση HDFS client**

Δημιουργήστε το αρχείο ρυθμίσεων core-site.xml:

nano hadoop-3.4.1**/**etc**/**hadoop**/**core-site.xml

<configuration>

<property>

<name>**fs.default.name**</name>

<value>**hdfs://hdfs-namenode:9000**</value>

</property>

</configuration>

Εάν όλα ρυθμίστηκαν σωστά, τότε ανοίξτε στον υπολογιστή σας την παρακάτω σελίδα

<http://hdfs-namenode:9870/>

όπου επιλέξτε Utilities -> Browse the file system κατόπιν δώστε τον κατάλογο /user/<username>. Εκεί θα μπορείτε να βλέπετε τα αποτελέσματα της εκτέλεσης των εργασιών σας.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Εκτέλεση δοκιμαστικού προγράμματος WordCount**

**🐍 Δημιουργία του αρχείου wordcount\_localdir.py (Python)**

Δημιουργήστε ένα αρχείο Python με το όνομα wordcount\_localdir.py και το εξής περιεχόμενο:

**from** pyspark **import** SparkContext # Εισαγωγή της κλάσης SparkContext για τη δημιουργία Spark εφαρμογής

# Δημιουργία SparkContext με όνομα εφαρμογής "WordCount"

sc **=** SparkContext**(**appName**=**"WordCount"**)**

# Ορισμός εισόδου - αρχείο στο HDFS (αντικαταστήστε το <username> με το δικό σας όνομα χρήστη)

input\_dir **=** "hdfs://hdfs-namenode:9000/user/<username>/text.txt"

# Απόκτηση του μοναδικού ID της εφαρμογής Spark

job\_id **=** sc**.***applicationId*

# Δημιουργία ονόματος εξόδου με βάση το job\_id (για αποφυγή σύγκρουσης)

output\_dir **=** f"hdfs://hdfs-namenode:9000/user/**<username>**/wordcount\_output\_{job\_id}"

# Διαβάζει το αρχείο κειμένου από το HDFS

text\_files **=** sc**.***textFile***(**input\_dir**)**

# Εκτελεί την καταμέτρηση λέξεων:

# 1. flatMap: σπάει κάθε γραμμή σε λέξεις

# 2. map: δημιουργεί ζεύγη (λέξη, 1)

# 3. reduceByKey: προσθέτει τις εμφανίσεις κάθε λέξης

word\_count **=** text\_files**.***flatMap***(lambda** line**:** line**.***split***(**" "**))** \

**.map(lambda** word**:** **(**word**,** 1**))** \

**.***reduceByKey***(lambda** a**,** b**:** a **+** b**)**

# Αποθηκεύει τα αποτελέσματα στο HDFS

word\_count**.***saveAsTextFile***(**output\_dir**)**

# Τερματισμός του SparkContext

sc**.***stop***()**

✅ **Σημείωση**: Αντικαταστήστε το <username> με το όνομα χρήστη που έχετε λάβει (π.χ. ikons).

Αντιγράψτε τα απαραίτητα αρχεία στο HDFS:

# Δημιουργία δοκιμαστικού αρχείου κειμένου με ένα παράδειγμα πρότασης

**echo** "this is a text file, with text document, to be used as input for the wordcount example" **>** **~/**text.txt

# Αντιγραφή του αρχείου Python στον προσωπικό φάκελο

**cp** wordcount\_localdir.py **~/**wordcount\_localdir.py

# Ανέβασμα του αρχείου κειμένου στο HDFS (ο φάκελος user/<username> πρέπει να υπάρχει)

hdfs dfs -put -f **~/**text.txt

# Ανέβασμα του αρχείου wordcount\_localdir.py στο HDFS

hdfs dfs -put -f **~/**wordcount\_localdir.py

📂 Η εντολή -put -f αντικαθιστά το αρχείο στο HDFS αν υπάρχει ήδη (force).

1. **Εκτέλεση Spark στο Kubernetes**

Το αρχείο wordcount\_localdir.py πρέπει να αποθηκευτεί στο HDFS, ώστε να είναι προσβάσιμο από τους spark executors κατά την εκτέλεση. Η τοποθεσία του αρχείου στο HDFS καθορίζεται στην τελευταία παράμετρο της εντολής spark-submit.

Εκτελέστε το παρακάτω spark-submit command:

spark-submit **\**

--master k8s**://**https**://**source-code-master**:**6443 **\**

--deploy-mode cluster **\**

--name wordcount **\**

--conf spark.hadoop.fs.permissions.umask-mode**=**000 **\**

--conf spark.kubernetes.authenticate.driver.serviceAccountName**=**spark **\**

--conf spark.kubernetes.namespace**=<username>**-priv **\**

--conf spark.executor.instances**=**5 **\**

--conf spark.kubernetes.container.image**=**apache**/**spark **\**

--conf spark.kubernetes.submission.waitAppCompletion=false \

--conf spark.eventLog.enabled**=**true **\**

--conf spark.eventLog.dir**=**hdfs**://**hdfs-namenode**:**9000**/**user**/<username>/**logs **\**

--conf spark.history.fs.logDirectory**=**hdfs**://**hdfs-namenode**:**9000**/**user**/<username>/**logs **\**

hdfs**://**hdfs-namenode**:**9000**/**user**/<username>/**wordcount\_localdir.py

Αλλάξτε το **<username>** με το όνομα χρήστη που λάβατε στο email, για παράδειγμα εγώ είμαι **ikons**.

Αυτή η εντολή spark-submit χρησιμοποιείται για να υποβληθεί μια εργασία Spark σε ένα Kubernetes cluster. Παρακάτω εξηγούνται οι παράμετροι της:

* --master k8s://https://source-code-master:6443: Ορίζει το Kubernetes master endpoint. Το k8s:// υποδεικνύει ότι η εργασία θα εκτελείται σε Kubernetes. Το https://source-code-master:6443 είναι η διεύθυνση του Kubernetes API server.
* --deploy-mode cluster: Ορίζει την κατάσταση εκτέλεσης της εφαρμογής. Το cluster σημαίνει ότι η εργασία θα εκτελείται μέσα στο Kubernetes cluster, αντί να εκτελείται στον τοπικό υπολογιστή του χρήστη (αν ήταν client mode).
* --name wordcount: Ορίζει το όνομα της εφαρμογής Spark που θα τρέξει στο Kubernetes. Εδώ η εφαρμογή έχει το όνομα wordcount.
* --conf spark.hadoop.fs.permissions.umask-mode=000: Ορίζει τα δικαιώματα πρόσβασης του συστήματος αρχείων Hadoop.
* --conf spark.kubernetes.authenticate.driver.serviceAccountName=spark: Ορίζει το όνομα του Kubernetes service account που θα χρησιμοποιηθεί από τον driver. Εδώ, ορίζεται το spark ως το service account.
* --conf spark.kubernetes.namespace=**<username>-priv**: Ορίζει το namespace του Kubernetes που θα χρησιμοποιηθεί για να τρέξει η εργασία. Εδώ, το namespace είναι το <username>-priv.
* --conf spark.executor.instances=5: Ορίζει τον αριθμό των εκτελεστών (executors) που θα δημιουργηθούν για την εκτέλεση της εργασίας. Εδώ, δημιουργούνται 5 εκτελεστές.
* --conf spark.kubernetes.container.image=apache/spark: Ορίζει την εικόνα του container που θα χρησιμοποιηθεί για την εκτέλεση της εργασίας. Εδώ χρησιμοποιείται η εικόνα apache/spark.
* --conf spark.kubernetes.submission.waitAppCompletion=false : Καθορίζει αν θα πρέπει να αναμένει την ολοκλήρωση της εφαρμογής πριν τερματιστεί η διαδικασία εκκίνησης. Όταν οριστεί σε false, η εργασία Spark ξεκινάει και δεν περιμένει την ολοκλήρωση της εκτέλεσης (fire and forget).
* --conf spark.eventLog.enabled=true: Ενεργοποιεί την καταγραφή των γεγονότων (event logging) της εργασίας Spark. Αυτό επιτρέπει την καταγραφή της εκτέλεσης της εφαρμογής.
* --conf spark.eventLog.dir=hdfs://hdfs-namenode:9000/user/ikons/logs: Ορίζει τη θέση του καταλόγου όπου θα αποθηκεύονται τα αρχεία καταγραφής των γεγονότων. Εδώ, η καταγραφή θα αποθηκευτεί στο HDFS του hdfs-namenode.
* --conf spark.history.fs.logDirectory=hdfs://hdfs-namenode:9000/user/**<username>**/logs: Ορίζει τη θέση όπου θα αποθηκεύονται τα αρχεία καταγραφής του ιστορικού (history logs) της εφαρμογής Spark για να επιτρέπει την παρακολούθηση του ιστορικού εκτέλεσης μέσω του Spark UI.
* hdfs://hdfs-namenode:9000/user/**<username>**/wordcount\_localdir.py: Ορίζει τη διαδρομή του αρχείου Python που περιέχει τον κώδικα της εφαρμογής. Στην περίπτωση αυτή, το αρχείο wordcount\_localdir.py βρίσκεται στο HDFS.

Όλοι οι παράμετροι που μπορούν να ρυθμιστούν είναι διαθέσιμοι στην παρακάτω σελίδα

<https://spark.apache.org/docs/latest/configuration.html>

Άλλοι σημαντικοί παράμετροι που μπορούν να σας βοηθήσουν κατά την εκτέλεση είναι οι:

* --conf spark.log.level=DEBUG: Παράγει περισσότερα μηνύματα κατά την εκτέλεση της εργασίας για αποσφαλμάτωση
* --conf spark.executor.memory=2g :Δεσμεύει περισσότερη μνήμη RAM ανά executor σε περίπτωση που οι εργασίες που εκτελούνται χρειάζονται περισσότερη RAM. Αυτό βοηθάει σε περιπτώσεις τερματισμού των executors λόγω μη διαθέσιμης μνήμης (OOM – Out of Memory Errors errors)

Αυτές οι παράμετροι επιτρέπουν τη σωστή εκτέλεση μιας εφαρμογής Spark σε ένα Kubernetes cluster, ενώ παρέχουν ρυθμίσεις για τη χρήση του HDFS, των logs, των εκτελεστών και άλλων παραμέτρων που σχετίζονται με το περιβάλλον του Kubernetes.

1. **Ρύθμιση προεπιλεγμένων παραμέτρων κατά την εκτέλεση εργασιών spark.**

Για να μην χρειάζεται να γράφετε όλες αυτές τις παραμέτρους κάθε φορά που εκτελείτε εργασίες spark, μπορείτε να τις εισάγετε σε ένα αρχείο ρυθμίσεων από όπου το spark θα τις αντλεί κάθε φορά που εκτελείτε την εντολή spark-submit. Για να το κάνετε αυτό τρέξτε τον παρακάτω κώδικα. Μην ξεχάσετε να αντικαταστήσετε το όνομα χρήστη (στην προκειμένη περίπτωση ikons) με το δικό σας όνομα χρήστη.

**cat** **>** **~/**spark-3.5.5-bin-hadoop3**/**conf**/**spark-defaults.conf **<<EOF**

spark.master k8s://https://source-code-master:6443

spark.submit.deployMode cluster

spark.hadoop.fs.permissions.umask-mode 000

spark.kubernetes.authenticate.driver.serviceAccountName spark

spark.kubernetes.namespace ikons-priv

spark.executor.instances 5

spark.executor.memory 1500m

spark.driver.memory 512m

spark.kubernetes.container.image=apache/spark

spark.kubernetes.submission.waitAppCompletion false

spark.eventLog.enabled true

spark.eventLog.dir hdfs://hdfs-namenode:9000/user/ikons/logs

spark.history.fs.logDirectory hdfs://hdfs-namenode:9000/user/ikons/logs

**EOF**

Τώρα μπορείτε να τρέξετε την προηγούμενη εντολή εκτελώντας απλά

spark-submit hdfs**://**hdfs-namenode**:**9000**/**user**/<username>/**wordcount\_localdir.py

1. **Παρακολούθηση Εκτέλεσης μέσω k9s**

Για να παρακολουθήσετε την εκτέλεση της εργασίας που μόλις υποβάλλατε, χρησιμοποιήστε το k9s:

k9s

**Παραδείγματα χρήσης**

**Εμφάνιση των pods**:

:pods

**Προβολή των logs ενός pod**:

l

**Έλεγχος κατάστασης ενός pod**:

d

1. **Χρήσιμες εντολές για διαχείριση αρχείων HDFS**

Εκτυπώνει τα περιεχόμενα του καταλόγου με όνομα <path>:

hadoop fs -ls <path>

Η εντολή αυτή δημιουργεί ταυτόχρονα έναν κατάλογο με υποκατάλογο:

hadoop fs –mkdir -p <path>

Η εντολή αυτή ανεβάζει στο hdfs ένα αρχείο από την τοπική τοποθεσία <localpath> στην απομακρυσμένη τοποθεσία <hdfspath>:

hadoop fs -put <localpath> <hdfspath>

1. **Χρήσιμες εντολές Linux**

ls

pwd

cd

cp

mv

cat

echo

man

Εφαρμογή επεξεργασίας κειμένου nano:

Ctrl +X: Exit

y/n: save or not

Enter: Αποθήκευση του αρχείου στο ίδιο κατάλογο/όνομα

1. **Εκτέλεση Spark τοπικά είτε διαδραστικά μέσω κελύφους είτε μέσω αρχείου py**

Όπως έχουμε δει στην θεωρία, το Spark μπορεί να τρέξει και τοπικά στον υπολογιστή μας είτε μέσω κελύφους (διαδραστικά) είτε μέσω εκτέλεσης python αρχείων με την εντολή spark-submit

* **Διαδραστική εκτέλεση μέσω κελύφους**

Με την παρακάτω εντολή ανοίγετε ένα κέλυφος spark (spark-shell) και μπορείτε να εκτελείτε εντολές spark κατευθείαν σε python.

pyspark --deploy-mode client --master local[\*]

Μπορείτε να διαβάζετε και να γράφετε αρχεία απευθείας από το hdfs του εργαστηρίου. Οι spark executors θα εκτελούνται από τον υπολογιστή σας εκμεταλλευόμενοι την μνήμη και τους επεξεργαστές του τοπικού σας υπολογιστή. Ανάλογα με τον υπολογιστή σας, θα πρέπει να ρυθμίσετε τις παραμέτρους του αρχείου ~/spark-3.5.5-bin-hadoop3/conf/spark-defaults.conf για να μπορέσει να εκτελεστεί ο κώδικάς σας. Οι παράμετροι είναι οι spark.executor.instances και spark.executor.memory. Οι πόροι που θα θέσετε θα πρέπει να μην ξεπερνούν την διαθέσιμη RAM και τους φυσικούς επεξεργαστές του υπολογιστή σας.

Μπορείτε να βλέπετε τις εργασίες που εκτελούνται μέσω της διεύθυνσης http://localhost:4040/

Για να τερματίσετε την εκτέλεση του κελύφους πατήστε τον συνδυασμό πλήκτρων Ctrl+d

* **Τοπική εκτέλεση αρχείου Python για δοκιμή**

Με την παρακάτω εντολή μπορείτε να εκτελέσετε το αρχείο .py που έχετε κατασκευάσει τοπικά πριν το ανεβάσετε και το εκτελέσετε στον k8s cluster.

spark-submit --deploy-mode client --master local[\*] wordcount\_localdir.py

Το αρχείο python που θα εκτελέσετε με αυτόν τον τρόπο μπορεί να διαβάσει και να γράφει δεδομένα στον απομακρυσμένο hdfs cluster του εργαστηρίου. Οι υπολογιστικοί πόροι που θα χρησιμοποιηθούν για την εκτέλεση είναι οι πόροι του υπολογιστή σας (μνήμη και επεξεργαστές). Όπως και πριν, θα πρέπει να φροντίσετε στο αρχείο ρυθμίσεων οι αιτούμενοι πόροι σε μνήμη και CPU να μην ξεπερνούν τους συνολικούς πόρους του υπολογιστή σας. Στην περίπτωση αυτή δεν χρειάζεται να κάνετε upload το .py αρχείο στο hdfs πριν το τρέξετε.

1. <https://kubernetes.io/> [↑](#footnote-ref-1)