Το ταξίδι

Τα βήματα που ακολουθήσαμε για την πραγματοποίηση μετρήσεων απόδοσης σε διαφορετικές εικονικές μηχανές της υπηρεσίας ~Okeanos του ΕΔΕΤ.

Εισαγωγή

> Ο Okeanos [1] είναι μια ελληνική υπηρεσία Cloud Computing με την μορφή δημόσιας υποδομής ως υπηρεσία (Infastracture as a service). Μέσω αυτής, ο χρήστης έχει την δυνατότητα να δημιουργεί και να διαχειρίζεται εικονικές μηχανές αναθέτοντας συγκεκριμένους πόρους σε CPUs, μνήμη (Memory size) και χώρο αποθήκευσης (Disk size).

Σύνδεση με την Υπηρεσία

Μέσω των υπηρεσιών ~okeanos, οποιοσδήποτε χρήστης της ακαδημαϊκής και ερευνητικής κοινότητας μπορεί να δημιουργήσει μια πολυ-επίπεδη εικονική υποδομή, συνδυάζοντας απλούστερα εικονικά δομικά στοιχεία. Περισσότερες πληροφορίες είναι διαθέσιμες στον ιστότοπο: https://okeanos.grnet.gr/about/who/ Για τη δημιουργία λογαριασμού και την αρχική σύνδεση στην υπηρεσία, ακολουθήστε τα βήματα:

1. Συνδεόμαστε στην διεύθυνση https://okeanos.grnet.gr
2. Επιλέγουμε **create an account now**
3. Επιλέγουμε την επιλογή **Academic**

Ανάκτηση Πόρων

Ο κάθε χρήστης, μετά τη σύνδεσή του με το σύστημα, έχει είτε ελάχιστους είτε καθόλου διαθέσιμους πόρους για την εγκατάσταση εικονικών μηχανών. Για να αποκτήσει κανείς πόρους, θα πρέπει να συμμετέχει σε ένα project ειδικού σκοπού.

Δημιουργία, Σύνδεση και Αναβάθμιση Πακέτων Εικονικής Μηχανής

1. Για τη δημιουργία της εικονικής μηχανής ακολουθούμε τον παρακάτω οδηγό: https://okeanos.grnet.gr/support/user-guide/cyclades-how-to-create-a-vm/
2. Για τη σύνδεση της εικονικής μηχανής : https://okeanos.grnet.gr/support/user-guide/cyclades-how-do-i-connect-to-a-vm/
3. Εφόσων καταφέρουμε να συνδεθούμε επιτυχώς, τότε μπορούμε να αλλάξουμε τον κωδίκο χρήστη με την ακόλουθει εντολή :

```
$ passwd
```

3. Για την αναβάθμιση των πακέτων :

```
$ sudo apt update
$ sudo apt upgrade
```

4. Για την εγκατάσταση των πακέτων monitoring

```
$ sudo apt install htop
```

Εγκατάσταση Apache Web-Server, PHP Και MySQL server

1. Για την εγκατάσταση του Apache :

```
$ sudo apt-get -y install apache2
```

2. Για την εγκατάσταση PHP :

```
$ sudo apt-get -y install php7.0-mysql php7.0-curl php7.0-json php7.0-cgi
php7.0 libapache2-mod-php7.0
```

3. Για την εγκατάσταση του MySQL server και του phpMyAdmin :

```
$ sudo apt-get -y install mysql-server
$ sudo apt-get -y install phpmyadmin
```

Τώρα μπορείτε να επισκεφτείτε το παρακάτω link : http://your-ipv4/phpmyadmin και να συνδεθείτε απομαρυσμένα στον MySQL server.

Εγκατάσταση WordPress

Με σκοπό να γίνει πιο σωστή προσομοίωση των πραγματικών συνθηκών ενός web server, έγινε εγκατάσταση wordpress

1. Για την εγκατάσταση του WordPress ακολουθούμε τον παρακάτω οδηγό: https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-lamp-on-ubuntu-18-04

Τεχνικά Χαρακτηριστικά Των Εικονικών Μηχανών

Εφόσον έχουμε εκτελέσει σωστά τα παραπάνω βήματα μπορούμε τώρα να ξεκινήσουμε, αρχικά με τον τρόπο που μπορούμε να βρούμε τα χαρακτηριστικά των μηχανών και να κάνουμε τα απαραίτητα Benchmark απόδοσης.

1. Με τις παρακάτω εντολές έχουμε πρόσβαση στις επιμέρους δυνατότητες των μηχανών οπως CPU και πληροφορίες σχετικά με την μνήμη

```
# The /proc/cpuinfo file contains details about individual cpu cores.
Output its contents with less or cat
```

```
$ cat /proc/cpuinfo
# To count the number of processing units use grep with wc:
$ cat /proc/cpuinfo | grep processor | wc -l
# To get the actual number of cores, check the core id for unique values
$ cat /proc/cpuinfo | grep 'core id'
# to report the amount of free and used memory
$ cat /proc/meminfo
```

Scripts Μετρησεων Αποδοσης (Benchmarks)

1. Gzip: script για την μέτρηση του χρόνου που χρειάζεται ένα αρχείο 2GB να συμπιεστεί ή να αποσυμπιεστεί, την απόδοση CPU και το μέγεθος αποσυμπίεσης.

```
# First we have to create a 2gb file with the command below :
dd if=/dev/zero of=testfile bs=1 count=0 seek=2G
# Gzip Compress
for r in 1 2 3
do
  echo "Round ${r}"
  echo "----------------------"
  for i in 1 6 9
    do
      cat testfile | /usr/bin/time -f "%E %S %U %P" -o
timerOutputCompress.${r}.${i}.txt gzip -${i} > testfile.${r}.${i}.gz
      cat timerOutputCompress.${r}.${i}.txt
      gzip -lv testfile.${r}.${i}.gz
  done
done

# Gzip Uncompress
for r in 1 2 3
do
  echo "Round ${r}"
  echo "----------------------"
  for i in 1 6 9
    do
       cat testfile.${r}.${i}.gz | /usr/bin/time -f "%E %S %U %P" -o
timerOutputUncompress.${r}.${i}.txt gzip -d > /dev/null
       cat timerOutputUncompress.${r}.${i}.txt
   done
done
```

2. ApacheBench: Χρησιμοποιήθηκε Το Apache HTTP server benchmarking tool,κατά το οποίο μετρήσαμε πόσα αιτήματα αντέχει το συστημά μας όταν εξυπηρετεί 1000 αιτήσεις, με 10 αιτήσεις να εξυπηρετούνται ταυτόχρονα.

```
#Install the apache2-utils package to get access to ApacheBench :
sudo apt-get install apache2-utils
# ApacheBench
```

```
for r in 1 2 3
do
  echo "Round ${r} - $(date +%Y%m%d)"
  ab -n 1000 -c 10 -g without_flag_${r}.data http://83.212.112.180/ >
without_flag_${r}.txt
  echo "Without Flags"
  echo "----------------------"
  cat without_flag_${r}.txt
  echo "----------------------"
  ab -l -r -n 1000 -c 10 -g with_flag_${r}.data -k -H "Accept-Encoding:
gzip, deflate" http://83.212.112.180/ > with_flag_${r}.txt
  echo "With Flags"
  echo "----------------------"
  cat with_flag_${r}.txt
  echo "----------------------"
done
```

3. PHPBench:Χρησιμοποιήθηκε το παρακάτω code block, το οποιο μετράει τον χρόνο εκτέλεσης σε μαθηματικές πράξεις, string manipulating, loop και if else.

▶ PHPBench Code

```
# see http://www.php-benchmark-script.com/
<?php

class benchmark {

        private static function test_Math($count = 140000) {
                $time_start = microtime(true);
                $mathFunctions = array("abs", "acos", "asin", "atan",
"bindec", "floor", "exp", "sin", "tan", "pi", "is_finite", "is_nan",
"sqrt");
                foreach ($mathFunctions as $key => $function) {
                        if (!function_exists($function))
unset($mathFunctions[$key]);
                }
                for ($i=0; $i < $count; $i++) {
                        foreach ($mathFunctions as $function) {
                                $r = call_user_func_array($function,
array($i));
                        }
                }
                return number_format(microtime(true) - $time_start, 3);
        }


        private static function test_StringManipulation($count = 130000) {
                $time_start = microtime(true);
                $stringFunctions = array("addslashes", "chunk_split",
"metaphone", "strip_tags", "md5", "sha1", "strtoupper", "strtolower",
"strrev", "strlen", "soundex", "ord");
```

```
                foreach ($stringFunctions as $key => $function) {
                        if (!function_exists($function))
unset($stringFunctions[$key]);
                }
                $string = "the quick brown fox jumps over the lazy dog";
                for ($i=0; $i < $count; $i++) {
                        foreach ($stringFunctions as $function) {
                                $r = call_user_func_array($function,
array($string));
                        }
                }
                return number_format(microtime(true) - $time_start, 3);
        }


        private static function test_Loops($count = 19000000) {
                $time_start = microtime(true);
                for($i = 0; $i < $count; ++$i);
                $i = 0; while($i < $count) ++$i;
                return number_format(microtime(true) - $time_start, 3);
        }


        private static function test_IfElse($count = 9000000) {
                $time_start = microtime(true);
                for ($i=0; $i < $count; $i++) {
                        if ($i == -1) {
                        } elseif ($i == -2) {
                        } else if ($i == -3) {
                        }
                }
                return number_format(microtime(true) - $time_start, 3);
        }

        public static function run ($echo = true) {
                $total = 0;
                $server = (isset($_SERVER['SERVER_NAME']) ?
$_SERVER['SERVER_NAME'] : '?') . '@' . (isset($_SERVER['SERVER_ADDR']) ?
$_SERVER['SERVER_ADDR'] : '?' );
                $methods = get_class_methods('benchmark');
                $line = str_pad("-",38,"-");
                $return = "<pre>$line\n|".str_pad("PHP BENCHMARK
SCRIPT",36," ",STR_PAD_BOTH)."|\n$line\nStart : ".date("Y-m-d
H:i:s")."\nServer : $server\nPHP version : ".PHP_VERSION."\nPlatform :
".PHP_OS. "\n$line\n";
                foreach ($methods as $method) {
                        if (preg_match('/^test_/', $method)) {
                                $total += $result = self::$method();
                                $return .= str_pad($method, 25) . " : " .
$result ." sec.\n";
                        }
                }
                $return .= str_pad("-", 38, "-") . "\n" . str_pad("Total
time:", 25) . " : " . $total ." sec.</pre>";
```

```
                if ($echo) echo $return;
                return $return;
        }

}

benchmark::run();


?>
```