Εικονικές Μηχανές και μετρήσεις απόδοσης

Gzip, ApacheBench, PHPBench

Πραγματοποίηση μετρήσεων απόδοσης σε διαφορετικές εικονικές μηχανές της υπηρεσίας ~Okeanos του ΕΔΕΤ και παρουσίαση των αποτελεσμάτων.

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).

Είδη Μετρήσεων Απόδοσης (benchmarks)

Τα benchmarks είναι τα εργαλεία που μας βοήθησαν στην προσέγγιση του προβλήματος μετρήσεων απόδοσης υποδομής υπολογιστικού νέφους. Benchmark είναι η εκτέλεση ενός προγράμματος ή ενός συνολου προγραμμάτων που αποσκοπεί -ούν στην μέτρηση απόδοσης συγκεκριμένων στοιχείων μιας συσκευής. Υπάρχουν διάφοροι τύποι Benchmark που μπορούν να μετρήσουν από την απόδοση CPU εως και χρόνους ταχύτητας μια προγραμματιστικής γλώσσας κ.α. Με την βοήθεια των Benchmark έχουμε την δυνατότητα να καταλάβουμε τα όρια της εικονικής μηχανής μας.

Σκοπός και αντικείμενο μελέτης

Σκοπός της εργασίας είναι η πραγματοποίηση μετρήσεων απόδοσης σε διαφορετικές εικονικές μηχανές της υπηρεσίας Okeanos με σκοπό να διαπιστωθεί η απόδοση σε διαφορετικές παραμετροποιήσεις των εικονικών μηχανημάτων. Για τις ανάγκες των μετρήσεων, δημιουργήθηκαν δύο εικονικές μηχανές με διαφορετική σύνθεση (configuration).Το λειτουργικό συστημα που χρησιμοποιήθηκε είναι Linux.Οι τύποι των Benchmark που χρησιμοποιήθηκαν είναι ο Gzip, ο ApacheBench και ο PHPBench.

Τεχνικά Χαρακτηριστικά

Το Linux λειτουργικό σύστημα που χρησιμοποιήθηκε για την δημιουργία των εικονικών μηχανημάτων ήταν το Ubuntu LTS. Παρακάτω παρατίθεται ο πίνακας με τα τεχνολογικά χαρακτηριστικά των εικονικών μηχανών.

Όνομα VM	**CPUs**	**Μνήμη**	**χώρος αποθήκευσης**
Medium	2x	2.00 GB	10.00 GB
Large	4x	6.00 GB	30.00 GB

Είδη Μετρήσεων Απόδοσης (benchmarks)

Τα εργαλεία, τα οποία χρησιμοποιήθηκαν για τις διάφορες μετρήσεις που έτρεξαν είναι:

1. Gzip: Ένα script [2] για την μέτρηση του χρόνου που χρειάζεται ένα αρχείο 2GB να συμπιεστεί ή να αποσυμπιεστεί, την απόδοση CPU και το μέγεθος αποσυμπίεσης.
2. ApacheBench: Χρησιμοποιήθηκε Το Apache HTTP server benchmarking tool [3], το οποίο μετράει τα αιτήματα ανά δευτερόλεπτο που μπορεί να εξυπηρετήσει το σύστημά μας (Server). Δημιουργήθηκε ένα custom script [4], κατά το οποίο μετρήσαμε πόσα αιτήματα αντέχει το συστημά μας όταν εξυπηρετεί 1000 αιτήσεις, με 10 αιτήσεις να εξυπηρετούνται ταυτόχρονα.
3. PHPBench:Χρησιμοποιήθηκε το PHP benchmark Script[5], που μετράει τον χρόνο εκτέλεσης σε μαθηματικές πράξεις, string manipulating, loop και if else.

Scripts Μετρησεων Αποδοσης

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
```

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
```

▶ PHPBench

```
# 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();
```

```
?>
```

αποτελέσματα των μετρήσεων

1. Για το πεδίο μέτρησης CPU, έτρεξαν 2 τεστ απο 3 επαναλήψεις: ένα για την φάση της συμπίεσης και ενα για την αποσυμπίεσην με την χρήση του Gzip.
2. Για το πεδίο μέτρησης Μεικτό, έτρεξαν 2 τεστ από 3 επαναλήψεις. Το πρώτο για τα αποτελέσματα του apacheBench και το δεύτερο για του PHPBench. Για το κάθε τεστ δημιουργήθηκε ένα γράφημα, το οποίο προέκυψε από την κάθε μέτρηση.

Fig. 1 Στις μετρήσεις του Gzip Compress παρατηρούμε ότι στο Large μηχανήμα μειώνεται αισθητά ο χρόνος συμπίεσης.Τα labels Large, Medium αναφέρονται στην εικονική μηχανή και τα fast, default, best στο εύρος συμπίεσης.

Fig. 2 Στις μετρήσεις του Gzip Uncompress παρατηρούμε ότι έχουμε σχεδόν τον ίδιο χρόνο αποσυμπίεσης, το οποίο είναι φυσιολογικό γιατί κατά την αποσυμπίεση χρειάζεται μικρότερη υπολογιστική ισχύς.

Fig. 3 και 4. Στις μετρήσεις του ApacheBench, οι ετικέτες Fastest , Slowest, Mean αναφέρονται στον χρόνο εκτέλεσης ενός αιτήματος. Οι διαφορές απόδοσης των δύο μηχανών είναι εμφανείς. Όσο πιο ισχυρό το μηχάνημα, τόσο πιο γρήγορη απόκριση, σχεδόν 50% πιο γρήγορη. Επίσης και στο διάγραμμα 4 παρατηρούμε ότι τα αιτήματα ανά δευτερόλεπτο είναι σχεδόν 50% περισσότερα για μια μεγαλύτερη μηχανή.

Fig. 5 Όπως μπορούμε να παρατηρήσουμε, στις μετρήσεις του PHPBench, οι τιμές είναι σχεδόν πανομοιότυπες. Ελάχιστα πιο γρήγορους συνολικούς χρόνους έχει η Large μηχανή.

Προβλήματα

Κατά την διάρκεια των μετρήσεων απόδοσης υποδομής της υπηρεσίας Cloud ~Okeanos, αντιμετωπίσαμε διάφορα προβλήματα με κυριότερα τα εξής:

1. Κατά τη διάρκεια κάποιων μετρήσεων υπήρχε μεγάλη επιβάρυνση του μηχανήματος, οπότε τα αποτελέσματα έπαιρναν πολύ παραπάνω χρόνο από τον αναμενόμενο να ολοκληρωθούν ή και πολλές φορές έφτανε στο σημείο να υπάρχει αποτυχία στην ολοκλήρωση, δηλαδή να γίνει υπερφόρτωση. Για να αποφευχθεί αυτό το πρόβλημα, χρησιμοποιήθηκαν μετρήσεις, όχι τόσο μεγάλου φορτίου.
2. Κατα τις μετρήσεις του ApacheBench τα αρχικά αποτελέσματα φαίνονταν μη ρεαλιστικά (πολύ γρήγορα). Με σκοπό να γίνει πιο σωστή προσομοίωση των πραγματικών συνθηκών ενός web server, έγινε εγκατάσταση wordpress ιστοσελίδας (http://83.212.112.180/).
3. Κατά τις μετρήσεις του PHPBench παρατηρήσαμε ότι τα αποτελέσματα ήταν σχεδόν ίδια. Θεωρητικά η Large μηχανή θα έπρεπε να είχε καλύτερα αποτελέσματα. Επίσης αξίζει να σημειωθεί οτι το PHPBench script χρησιμοποιήθηκε και σε άλλες μεγαλύτερες μηχανές εκτός της υπηρεσίας ~Okeanos με μεγάλη διαφορά στους χρόνους εκτέλεσης.
4. Οι πόροι που μας δόθηκαν για την πραγματοποίηση αυτών των μετρήσεων ήταν περιορισμένοι. Θα ήταν χρήσιμο, μέσω περισσότερων πόρων, να είχαν δημιουργηθεί πιο πολλές εικονικές μηχανές.

Συμπεράσματα

Η υπηρεσία Cloud Computing αποτελεί μια πολύ σημαντική εξέλιξη στον χώρο της πληροφορικής. Ήδη μεγάλο σύνολο των επιχειρήσεων έχουν μεταφέρει τις εφαρμογές και τα αρχεία τους στο «σύννεφο» με θετικό αποτέλεσμα όλα αυτά να είναι προσβάσιμα από οποιοδήποτε μέρος του κόσμου και οποιαδήποτε στιγμή θελήσει ο χρήστης.

Όσον αφορά την υπηρεσία ~Okeanos οι μετρήσεις έδειξαν μια φυσιολογική αύξηση στην απόδοση συστήματος κατά την διάρκεια αναβάθμισης εικονικών μηχανών. Παρόλα αυτά ο ~Okeanos που προορίζεται για χρήστες της ακαδημαϊκής και ερευνητικής κοινότητας μπορεί δύσκολα να υποστηρίξει προγράμματα που χρειάζονται δυνατή υπολογιστική ισχύ, μεγάλες υποδομές και γρήγορη αλλαγή κλίμακας υπηρεσιών. Άλλες υπηρεσίες cloud όπως η Azure δίνουν την δυνατότητα δημιουργίας εικονικής μηχανής με χαρακτηριστικά (CPU, Memory, κ.α) που ξεπερνούν καταά πολυ αυτά που προσφέρει ο ~Okeanos. Τέλος τα αποτελέσματα των μετρήσεων κρίνονται ικανοποιητικά.

Αναφορές

1. ~Okeanos IAAS. (n.d.). (2019). [online] Διαθέσιμο: https://okeanos.grnet.gr/home/
2. George Simos. (2019). Gzip script.[online] Διαθέσιμο: https://github.com/georgesimos/virtual-machine-performance-benchmarking/blob/master/Benchmarks%20scripts/GzipBench.sh
3. ApacheBench. (2019), [online] Διαθέσιμο: https://httpd.apache.org/docs/2.4/programs/ab.html
4. George Simos. (2019). ApacheBench script. [online] Διαθέσιμο: https://github.com/georgesimos/virtual-machine-performance-benchmarking/blob/master/Benchmarks%20scripts/ApacheBench.sh
5. Alessandro Torrisi (2012). PHP performance benchmark script. [online] Διαθέσιμο: http://www.php-benchmark-script.com/
6. George Simos. (2019). Gzip results. [online] Διαθέσιμο: https://github.com/georgesimos/virtual-machine-performance-benchmarking/blob/master/Benchmarks%20results/GzipBench.xlsx
7. George Simos. (2019). ApacheBench results. [online] https://github.com/georgesimos/virtual-machine-performance-benchmarking/blob/master/Benchmarks%20results/ApacheBench.xlsx
8. George Simos. (2019). Php Results. [online] Διαθέσιμο: https://github.com/georgesimos/virtual-machine-performance-benchmarking/blob/master/Benchmarks%20results/PHPBench.xlsx