ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΞΙΟΛΟΓΗΣΗ ΤΕΧΝΟΛΟΓΙΩΝ ΕΛΑΦΡΑΣ ΕΙΚΟΝΙΚΟΠΟΙΗΣΗΣ

Διπλωματική Εργασία

του

Ψαρρά Δημητρίου

Θεσσαλονίκη, Σεπτέμβριος 2020

ΑΞΙΟΛΟΓΗΣΗ ΤΕΧΝΟΛΟΓΙΩΝ ΕΛΑΦΡΑΣ ΕΙΚΟΝΙΚΟΠΟΙΗΣΗΣ

Ψαρράς Δημήτριος

Πτυχίο Μηχανικού Λογισμικού , ΑΤΕΙ Θεσσαλονίκης 2010

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής
Μαμάτας Ελευθέριος

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Μαμάτας Ελευθέριος	Χατζηγεωργίου Αλέξανδρος	Παπαδημητρίου Παναγιώτης
.....	.....	.....

Ψαρράς Δημήτριος

.....

Περίληψη

Τα τελευταία χρόνια η Microsoft εισάγει όλο και περισσότερο διάφορα κομμάτια και λειτουργίες του Λειτουργικού Συστήματος Linux στα δικά της Λειτουργικά Συστήματα. Η παρούσα διπλωματική εργασία μελετά τις τεχνολογίες εικονικοποίησης λογισμικού και προσπαθεί να αναδείξει την απόδοση των υποδοχέων (containers) Linux όταν αυτοί εκτελούνται σε διαφορετικά Λειτουργικά Συστήματα της Microsoft. Ο κύριος στόχος της εργασίας είναι να εξεταστεί η απόδοση τους και να συγκριθεί η ιστορική εξέλιξη της δυνατότητας εκτέλεσης υποδοχέων Linux μέσα σε περιβάλλον Microsoft. Στο θεωρητικό υπόβαθρο της εργασίας παρουσιάζονται οι τεχνολογίες των υποδοχέων αλλά και των εικονικών μηχανών μέσω των οποίων μπορεί να γίνεται η εκτέλεση των πρώτων, οι διαφορές, τα πλεονεκτήματα και τα μειονεκτήματά τους. Ακόμη, γίνεται πλήρη αναφορά στα εργαλεία που χρησιμοποιήθηκαν για την εκτέλεση και συλλογή αποτελεσμάτων, αλλά και στον τρόπο λειτουργίας τους. Τέλος, τα συμπεράσματα που εξήχθησαν από συγκριτικές δοκιμές που πραγματοποιήθηκαν με διαφορετικά πειραματικά σενάρια, αναδεικνύουν την ιστορική εξέλιξη της εκτέλεσης των υποδοχέων Linux στα Λειτουργικά συστήματα Windows αξιοποιώντας την τεχνολογία των εικονικών μηχανών.

Abstract

Since the last few years, Microsoft is implementing more and more functions of Linux Operating Systems to Windows. This master thesis investigates software oriented virtualization methods and tries to highlight the performance of Linux containers, in association to Microsoft operating systems. The main purpose of this thesis is to compare the results and examine the historical evolution of this procedure. The background of this work includes the technologies of hypervisors and containers as well as highlights the advantages and disadvantages of both. In addition, we detail all applications and tools we are using in our experimentation analysis that includes: i) Docker ToolBox, ii) Docker For Windows, iii) Phoronix Test Suite. Last but not least, all the conclusions from the above experimental results are highlighting the expected evolution in the manipulation of Linux containers in a Microsoft Operating System.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή κ. Μαμάτα Ελευθέριο για την υποδειγματική βοήθεια, την υποστήριξη αλλά και την υπομονή του.

Ευχαριστώ επίσης την οικογένειά μου για την υποστήριξη της και τις θυσίες που έκαναν ώστε να έχω τον απαιτούμενο χρόνο για την συγγραφή αυτής της εργασίας.

Περιεχόμενα

Κεφάλαιο 1 – Σκοπός και στόχοι

1.1 Πρόβλημα - Σημαντικότητα του θέματος

Στις μέρες μας, έχει παρατηρηθεί μια αυξανόμενη προσπάθεια διαφορετικών λειτουργικών συστημάτων να μοιράζονται τις δυνατότητες μεταξύ τους. Έτσι μετά από αρκετά χρόνια προσπαθειών για το πιο λειτουργικό σύστημα θα επικρατήσει, φτάσαμε στην πρόσφατη περίοδο, όπου η Microsoft ανακοίνωσε την ενσωμάτωση πυρήνα Linux στα Microsoft Windows. Τα παραπάνω, σε συνδυασμό με την αυξανόμενη χρήση των Εικονικών Μηχανών (Virtual Machines –VMs) και των Υποδοχέων (Containers) έχει δώσει πλέον τη δυνατότητα σε διαχειριστές συστημάτων Microsoft Windows, να δημιουργούν, να εκτελούν, αλλά και να διαχειρίζονται Linux Containers. Η διπλωματική εργασία θα προσπαθήσει να διερευνήσει την εξέλιξη αυτής της προσπάθειας, ώστε να εξαχθούν συμπεράσματα για την απόδοση των Linux Containers μέσα από λειτουργικά συστήματα Microsoft.

1.2 Σκοπός – Στόχοι

Ο κύριος σκοπός αυτής της μελέτης είναι η μελέτη της απόδοσης των υποδοχέων σε περιβάλλοντα Windows, μέσω σύγκρισης της απόδοσης αντιπροσωπευτικών υποδοχέων ανάμεσα σε διαφορετικά λειτουργικά συστήματα της Microsoft και το Linux. Για την επίτευξη του παραπάνω σκοπού υλοποιήσαμε τα μεθοδολογικά βήματα:

- Εγκαταστήσαμε την προτεινόμενη λύση για Docker Containers σε τρία διαφορετικά λειτουργικά συστήματα Microsoft αλλά και σε Linux Ubuntu.
- Εγκαταστήσαμε τις απαραίτητες εφαρμογές για την δημιουργία, εκτέλεση και διαχείριση των Containers.
- Καταγράψαμε και αναλύσαμε την απόδοση των παραπάνω μέσω αντίστοιχων εργαλείων καταγραφής απόδοσης.
- Συγκρίναμε και αξιολογήσαμε τα αποτελέσματα.

1.3 Ερευνητικά ερωτήματα

Σκοπός των παρακάτω ερευνητικών ερωτημάτων είναι να οδηγήσουν την έρευνα μας στην επίτευξη των στόχων της.

Ερώτημα 1: Υπάρχει η δυνατότητα χρήσης Microsoft Windows λειτουργικών συστημάτων για την εκτέλεση υποδοχέων Linux, χωρίς να επηρεάζεται σημαντικά η απόδοση τους;

Ερώτημα 2: Υπάρχει βαθμιαία βελτίωση των δυνατοτήτων εκτέλεσης των υποδοχέων, ταυτόχρονα με την εξέλιξη των λειτουργικών συστημάτων της Microsoft, από τα Windows 10 Professional ως και τα Windows Server 2019;

1.4 Οργάνωση Κειμένου

Στο **πρώτο κεφάλαιο** παρουσιάζεται το πρόβλημα που έχουμε να αντιμετωπίσουμε και η σημαντικότητα αυτού. Ακολουθεί ο σκοπός και ο στόχος που θέσαμε μαζί με τα ερευνητικά ερωτήματα. Τέλος παρουσιάζεται το περιεχόμενο του κειμένου.

Στο **δεύτερο κεφάλαιο**, γίνεται μια εκτενείς αναφορά στις τεχνολογίες που θα χρησιμοποιηθούν. Συγκεκριμένα περιγράφονται οι τεχνολογίες των εικονικών μηχανών με τα χαρακτηριστικά τους γνωρίσματα, τα πλεονεκτήματα και τα μειονεκτήματά τους. Στη συνέχεια γίνεται αναφορά στις εφαρμογές που θα χρησιμοποιηθούν για να δημιουργήσουμε, να εκτελέσουμε και να διαχειριστούμε τους υποδοχείς, καθώς και τον τρόπο που τις χρησιμοποιούμε στη διατριβή.

Στο **τρίτο κεφάλαιο**, πραγματοποιείται αναφορά στην μεθοδολογία που θα ακολουθήσουμε στην πειραματική μας ανάλυση, συμπεριλαμβάνοντας τους δείκτες μέτρησης που επιλέξαμε. Στη συνέχεια του κεφαλαίου, αναλύουμε τον τρόπο υλοποίησης τεσσάρων διαφορετικών συστημάτων που υλοποιήσαμε για τις ανάγκες της πειραματικής ανάλυσης. Γίνεται πλήρη αναφορά στον τρόπο εγκατάστασης του Docker Toolbox σε περιβάλλον Windows 10, του Docker for Windows σε περιβάλλον Windows 10 και Windows Server 2019 και τέλος του Docker σε Ubuntu Linux.

Στη συνέχεια, στα πλαίσια του **τέταρτου κεφαλαίου** υλοποιείται η πειραματική προσέγγιση μας και αναφέρονται οι συγκριτικές δοκιμές που πραγματοποιήσαμε με τα αποτελέσματά τους. Τέλος, στο **πέμπτο κεφάλαιο** γίνεται αναφορά στα συμπεράσματα της συγκριτικής μελέτης μας και δίνονται μερικές προτάσεις για μελλοντική εξελιξη της υπάρχουσας έρευνας.

Κεφάλαιο 2 - Θεωρητικό υπόβαθρο

Στο δεύτερο κεφάλαιο αναλύουμε το θεωρητικό υπόβαθρο και τα εργαλεία που χρησιμοποιήσαμε για τη συγκριτική μελέτη μας, ώστε να εξάγουμε χρήσιμα συμπεράσματα στα πλαίσια των στόχων της διατριβής. Συγκεκριμένα γίνεται αναφορά στις εικονικές μηχανές, στους υποδοχείς και στα πλεονεκτήματα και μειονεκτήματα που έχει η κάθε μία τεχνολογία. Στη συνέχεια εξετάζουμε το εργαλείο που χρησιμοποιήσαμε για την συγκριτική μελέτη μας, το Phoronix Test Suite και τέλος το Openbenchmark.org, στο οποίο γίνεται η προβολή και απεικόνιση των αποτελεσμάτων που εξάγουμε.

2.1 Εικονικές Μηχανές - Εικονικοποίηση υλικού

2.1.1 Χαρακτηριστικά των Εικονικών Μηχανών

Μια εικονική μηχανή είναι ένα αρχείο υπολογιστή που τυπικά ονομάζεται εικόνα, το οποίο λειτουργεί σαν ένας κανονικός υπολογιστής. Με άλλα λόγια, μια εικονική μηχανή είναι ένας υπολογιστής μέσα σε κάποιον άλλο υπολογιστή. Εκτελείται σαν όλα τα υπόλοιπα προγράμματα και δίνει στον τελικό χρήστη τις ίδιες δυνατότητες και εμπειρίες σαν να είναι ένας φυσικός υπολογιστής. Η τεχνολογία αυτή, μέσω συγκεκριμένων εργαλείων, μας δίνει τη δυνατότητα να μπορούμε να διαμοιραστούμε το υλικό ενός υπολογιστή σε πολλές διαφορετικές εικονικές μηχανές. Η εικονικοποίηση αποτελεί μια τεχνική επιπέδου προγράμματος η οποία μπορεί να εκτελείται είτε απευθείας στο υλικό, είτε μέσα σε ένα λειτουργικό σύστημα.

Κάθε μια εικονική μηχανή αποτελείται από ένα σύνολο ρυθμίσεων και αρχείων που διατηρούνται στο φυσικό υπολογιστή που εκτελείται και είναι απομονωμένη η μία από

την άλλη, σαν να έχει τους δικούς της πόρους, δηλαδή το δικό της επεξεργαστή, την δική της μνήμη και τον δικό της αποθηκευτικό χώρο. Για τους παραπάνω λόγους στην επιστήμη υπολογιστών έχει επικρατήσει το λειτουργικό σύστημα που φιλοξενεί τις εικονικές μηχανές να αναφέρεται ως οικοδεσπότης (Host), ενώ οι ίδιες οι εικονικές μηχανές ως φιλοξενούμενες (Guest).

Τα τρία βασικά χαρακτηριστικά των εικονικών μηχανών είναι :

- **Partitioning**: Στην εικονικοποίηση, πολλές εφαρμογές και λειτουργικά συστήματα υποστηρίζονται από ένα φυσικό μηχάνημα μέσω του διαμοιρασμού των διαθέσιμων πόρων.
- **Isolation**: Κάθε εικονική μηχανή είναι απομονωμένη από το φυσικό υπολογιστή που τη φιλοξενεί, αλλά και από τις υπόλοιπες εικονικές μηχανές. Αυτή η απομόνωση έχει ως συνέπεια όταν υπάρχει πρόβλημα σε μια από αυτές, να μην επηρεάζονται οι υπόλοιπες διότι δεν μοιράζονται κοινούς υπολογιστικούς πόρους και δεδομένα.
- **Encapsulation**: Όλες οι εικονικές μηχανές μπορούν να αποθηκευτούν ως ένα αρχείο υπολογιστή. Έτσι έχουμε τη δυνατότητα φορητότητας μεταξύ διαφορετικών υπολογιστικών συστημάτων χωρίς να επηρεάζεται η εκτέλεση και η εύρυθμη λειτουργία των υπόλοιπων εικονικών μηχανών ακόμη και κατά την εκκίνηση νέων εικονικών μηχανών.

2.1.2 Εργαλεία διαχείρισης Εικονικών Μηχανών – Hypervisors

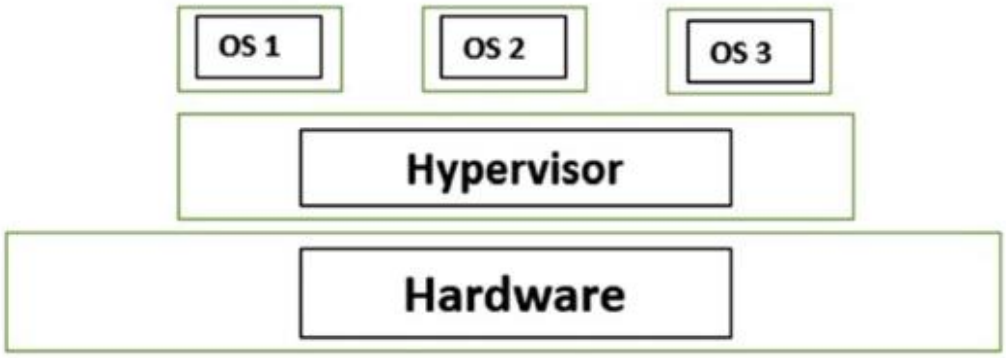
Ως Hypervisor ορίζεται ένα λογισμικό, ένα υλικολογισμικό ή ένα υλικό το οποίο δημιουργεί, εκτελεί και διαχειρίζεται εικονικές μηχανές. Ο Hypervisor είναι αυτός που ενεργοποιεί την επικοινωνία μεταξύ του υλικού και της εικονικής μηχανής και δίνει τη δυνατότητα της διαχείρισής τους μέσω μιας πλατφόρμας.

Ο Hypervisor γνωστός και με το όνομα Virtual Machine Monitor (VMM), διαχωρίζει τις εικονικές μηχανές μεταξύ τους και αναθέτει στην κάθε μία, επεξεργαστή, μνήμη και αποθηκευτικό χώρο με τέτοιο τρόπο ώστε να μην επηρεάζονται μεταξύ τους.

Υπάρχουν δύο τύποι Hypervisor:

Hypervisor τύπου 1

Ο Hypervisor τύπου 1, είναι γνωστός και ως hypervisor επιπέδου υλικού. Επικοινωνεί απευθείας με το υλικό και επιτρέπει στα φιλοξενούμενα λειτουργικά συστήματα να έχουν πρόσβαση στο υλικό του ηλεκτρονικού υπολογιστή Εικόνα 1. Είναι αυτός που παίρνει τον ρόλο του οικοδεσπότη. Οι πιο γνωστοί Hypervisors είναι οι VMware vSphere/ESXi, Microsoft Windows Server Hyper-V, Citrix XenServer, Red Hat Enterprise Virtualization και Kernel-Based Virtual machine.

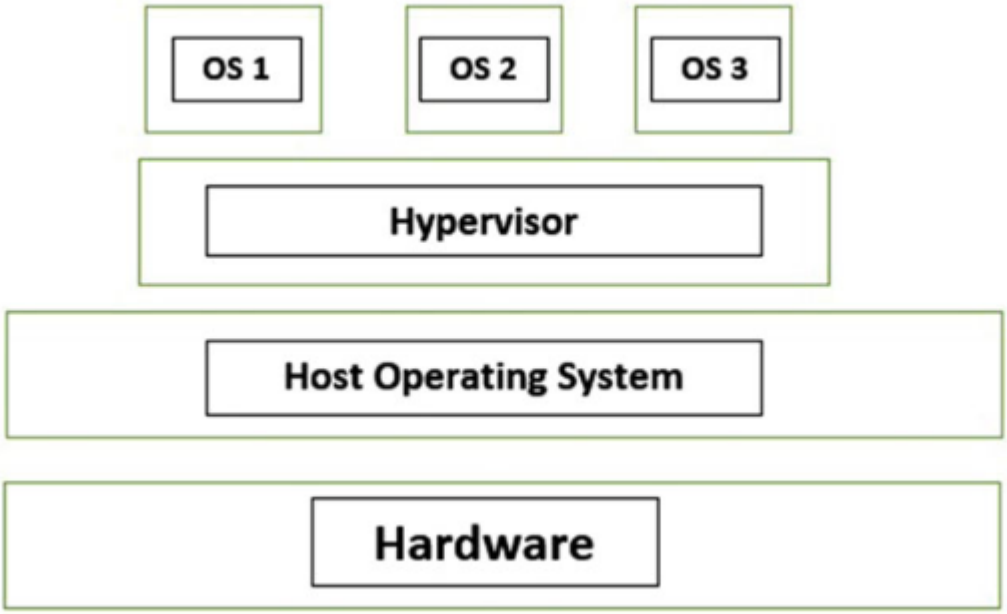


Εικόνα 1. HyperVisor τύπου 1

Βασικό πλεονέκτημα αυτού του τύπου είναι η αποτελεσματικότητα του λόγω της άμεσης επικοινωνίας με το υλικό. Ακόμη έχει υψηλό επίπεδο ασφάλειας διότι δεν υπάρχει κάτι άλλο ανάμεσα στον Hypervisor και στον επεξεργαστή που θα μπορούσε να εκμεταλλευτεί ένας κακόβουλος χρήστης.

Hypervisor τύπου 2

Ο Hypervisor τύπου 2 δεν επικοινωνεί απευθείας με το υλικό, αλλά λειτουργεί ως μια εφαρμογή μέσα σε ένα λειτουργικό σύστημα. Η βασική διαφορά με του τύπου 1 είναι, σε αυτήν την περίπτωση, η κάθε εικονική μηχανή εκτελείται σαν μια διαδικασία (process). Οι πιο γνωστοί Hypervisors τύπου 2 είναι το VirtualBox και το VMWare Workstation. Στην Εικόνα 2 βλέπουμε τον Hypervisor τύπου 2 και συγκρίνοντας με την προηγούμενη εικόνα του Hypervisor τύπου 1 η διαφορά τους είναι ξεκάθαρη.



Εικόνα 2. Hypervisor τύπου 2

2.1.3 Πλεονεκτήματα και μειονεκτήματα των Εικονικών Μηχανών

Πλεονεκτήματα

Βασικό πλεονέκτημα της χρήσης των εικονικών μηχανών, σε αντίθεση με τις παραδοσιακές μορφές φυσικών συστημάτων είναι η μείωση του κόστους. Η μείωση αυτή δεν αφορά μόνο στην αγορά και εγκατάσταση του υλικού, αλλά ακόμη και στην κατανάλωση ενέργειας κάτι που στις μέρες μας θεωρείται μείζον θέμα[].

Ακόμη, με τη δημιουργία του επιπέδου εικονικοποίησης (virtualization layer) διαχωρίζεται το υλικό από το λογισμικό. Αυτό, δίνει τη δυνατότητα στους διαχειριστές των συστημάτων να έχουν πλήρη πρόσβαση να προσαρμόζουν τις ανάγκες τους σε σύντομο χρονικό διάστημα ή ακόμη και με μηδενικό downtime. Έτσι, ανά πάσα στιγμή μπορεί να αυξηθεί η επεξεργαστικής ισχύ, η μνήμη και ο αποθηκευτικός χώρος μιας εικονικής μηχανής, εάν οι ανάγκες το απαιτούν. Η διαχείριση και η μεταφορά μιας εικονικής μηχανής σε νέο υλικό μπορεί να γίνει άμεσα σε ελάχιστο χρονικό διάστημα όπως και η επαναφορά σε περίπτωση failover μέσω της δυνατότητας των snapshots που έχει ο διαχειριστής.

Πολύ σημαντικό είναι και το όφελος του χρόνου δημιουργίας ενός ολόκληρου Data Center. Με την τεχνολογία της εικονικοποίησης η δημιουργία ενός ολόκληρου Data Center απαιτεί ελάχιστο δυνατό υλικό, αλλά και χρόνο ώστε να είναι έτοιμο προς χρήση. Αποτελεί ίσως τον κυριότερο τρόπο για να διαχειριστούν οι εταιρείες κρίσεις, όπως τα προβλήματα υλικού.

Μειονεκτήματα

Αν και η εικονική μηχανή δεν μπορεί να καταλάβει ότι δεν έχει άμεση επαφή με το υλικό, παρά μόνο μέσω του ενδιάμεσου Hypervisor, η εκτέλεση ενός λειτουργικού συστήματος μέσω αυτού δημιουργεί μικρά προβλήματα απόδοσης.

Ένα δεύτερο μειονέκτημα των εικονικών μηχανών είναι ότι ο οικοδεσπότης έχει περιορισμένους πόρους με αποτέλεσμα να επηρεάζεται η απόδοση των μηχανών, όταν δεν υπάρχουν οι απαιτούμενοι πόροι για να μοιραστούν μεταξύ τους.

Τέλος, θα πρέπει να αναφέρουμε ότι η κάθε εικονική μηχανή είναι εξαρτημένη από τον Hypervisor. Αυτό έχει ως αποτέλεσμα, εάν υπάρχει δυσλειτουργία σε αυτόν, να επηρεάζονται και όλες οι εικονικές μηχανές τις οποίες διαχειρίζεται.

2.1.4 Microsoft Hyper-V Server

Η λύση της Microsoft για τη δημιουργία, εκτέλεση και διαχείριση εικονικών μηχανών είναι ο Microsoft Hyper-V Server. Πρωτοεμφανίστηκε με τα Windows Server 2008 και η πρώτη επίσημη έκδοση ανακοινώθηκε στις 26 Ιουνίου 2008. Από τότε και στο εξής αποτελεί αναπόσπαστο κομμάτι των Windows Server και ενσωματώθηκε και στην έκδοση Windows 10 Version 1709 2018.

2.2 Υποδοχείς - Εικονικοποίηση λογισμικού

2.2.1 Χαρακτηριστικά των Υποδοχέων

Container ονομάζουμε μια εκτελέσιμη μονάδα λογισμικού μέσα στην οποία περιέχονται εκτός από την ίδια την εφαρμογή και οι απαραίτητες βιβλιοθήκες και εξαρτήσεις που έχει σε επίπεδο λειτουργικού συστήματος. Με άλλα λόγια μπορεί να εκτελεστεί σε οποιοδήποτε περιβάλλον, είτε αυτός είναι ένας Server, είτε ένας προσωπικός υπολογιστής, είτε το σύννεφο (cloud). Η τεχνολογία των Containers είναι γνωστή ως εικονικοποίηση σε επίπεδο λειτουργικού συστήματος, διότι είναι μια διαδικασία μέσω της οποίας ο πυρήνας του λειτουργικού συστήματος επιτρέπει πολλαπλά απομονωμένα στιγμιότυπα. Αυτό έχει ως αποτέλεσμα κάθε εφαρμογή που τρέχει στον δικό της Container να μην έχει πρόσβαση στο υπόλοιπο λειτουργικό σύστημα.

Σύμφωνα με το Open Container Initiative (OCI), τον οργανισμό που είναι υπεύθυνος για την οργάνωση και τη δημιουργία των προτύπων για τους Containers, τα δύο σημαντικότερα χαρακτηριστικά τους είναι τα Cgroups και τα Namespaces.

Control groups (Cgroups)

Τα Cgroups είναι αυτά που καθορίζουν τη διαθεσιμότητα των πόρων του κοινού πυρήνα που έχει ο Container στη διάθεσή του. Μέσω αυτού του μηχανισμού, ο διαχειριστής του έχει τη δυνατότητα να ρυθμίσει ξεχωριστά τους πόρους του συστήματος, δηλαδή τον επεξεργαστή, την μνήμη, κλπ.

Namespaces

Τα Namespaces είναι αυτά που καθορίζουν τι μπορεί να δει ο Container και πιο συγκεκριμένα τι προσβάσεις έχει. Είναι η τεχνολογία αυτή, που δίνει τη δυνατότητα της απομόνωσης στον κάθε Container. Ένα παράδειγμα είναι η απομόνωση από πόρους δικτύου ή συγκεκριμένων διεργασιών του λειτουργικού.

2.2.2 Πλεονεκτήματα και μειονεκτήματα των Containers

Πλεονεκτήματα

Το βασικό πλεονέκτημα των Containers είναι η αποδοτικότητα των πόρων του συστήματος. Σε σύγκριση με τις εικονικές μηχανές οι Containers, χρειάζονται πολύ λιγότερους πόρους όσον αφορά τη μνήμη, την επεξεργαστική ισχύ και φυσικά τον αποθηκευτικό χώρο.

Το πλεονέκτημα του μεγέθους που αναφέραμε και στην προηγούμενη παράγραφο έχει και ως συνέπεια οι Containers να είναι και πιο γρήγοροι, όσον αφορά την εγκατάσταση, τη διαχείριση και τη μεταφορά. Αυτό συνδυάζεται με τη δυνατότητα ο διαχειριστής να βρει έτοιμους Containers, με τις βασικές ρυθμίσεις έτοιμες και ανάλογα με τις ανάγκες του.

Τέλος, ένα από τα πιο σημαντικά πλεονεκτήματά τους είναι η εύκολη και ταυτόχρονα πολύ γρήγορη δυνατότητα των containers να δημιουργούν αντίγραφα, ώστε να επιτευχθεί διαμοιρασμός του φόρτου εργασίας. Σε αντίθεση με τα VMs, ένας container απαιτεί ελάχιστα δευτερόλεπτα για να ξεκινήσει και ταυτόχρονα μπορεί, κατά τη δημιουργία του να εγκατασταθεί ταυτόχρονα σε πολλούς διαφορετικούς διακομιστές.

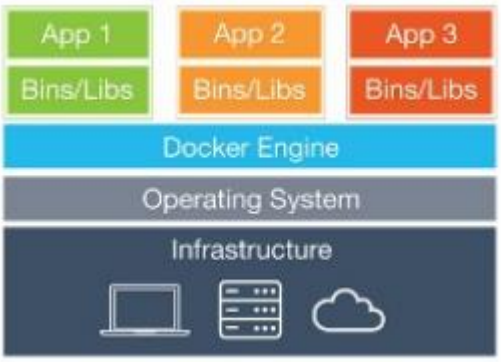
Μειονεκτήματα

Το μεγαλύτερο πρόβλημα που υπήρχε μέχρι το 2019 ήταν ο περιορισμός των Containers μόνο σε εφαρμογές Linux. Βλέποντας όμως το μέλλον και τη χρησιμότητα των Containers, η Microsoft έχει ήδη βγάλει κάποιες εκδόσεις τις δικής της έκδοσης. Εκτός όμως από τη λύση για εφαρμογές Windows η Microsoft έχει δώσει τη δυνατότητα να μπορεί ένας προγραμματιστής να έχει τους Linux Containers κάτω από μια δική της έκδοση. Αυτό όπως θα δούμε και παρακάτω το επιτυγχάνει με διαφορετικούς τρόπους, τους οποίους εξελίσσει με την πάροδο του χρόνου.

Η εικονικοποίηση που παρέχουν οι Containers είναι άμεσα σχετιζόμενη με τον πυρήνα Linux. Έτσι δεν υπάρχει πλήρης απομόνωση τους με συνέπεια να υπάρχουν κίνδυνοι ασφάλειας. Για τον παραπάνω λόγο, πολλές φορές οι διαχειριστές δημιουργούν τους Containers μέσα σε ένα VM ώστε να υπάρξει πλήρης απομόνωση.

2.2.3 Περιβάλλον υποδοχέων Docker

Το Docker αποτελεί την πιο διαδεδομένη πλατφόρμα για τη χρήση της τεχνολογίας εικονικοποίησης σε επίπεδο λειτουργικού συστήματος, με σκοπό τη δημιουργία Containers. Είναι γραμμένο σε γλώσσα προγραμματισμού GoLang και πλέον θεωρείται συνώνυμο με την τεχνολογία των Containers. Οι Containers που δημιουργεί η εφαρμογή έχουν όλα εκείνα τα χαρακτηριστικά γνωρίσματα όπως την απομόνωση αλλά και τον έλεγχο των πόρων. Πρωτοεμφανίστηκε το 2013 και γρήγορα έγινε η κύρια πλατφόρμα δημιουργίας Containers παγκοσμίως. Στην Εικόνα 3 βλέπουμε ένα διάγραμμα της τοπολογίας των Containers και πως αυτοί βρίσκονται ουσιαστικά κάτω από το ίδιο Λειτουργικό Σύστημα.



Εικόνα 3. Containers

Docker Client

Αποτελεί την εφαρμογή που χρησιμοποιεί ο χρήστης ώστε να διαχειριστεί, να δημιουργήσει και να εκτελέσει containers. Συνήθως χρησιμοποιείται ως Docker Client το Command Line το οποίο επικοινωνεί με τον Docker deamon.

Docker Engine

Είναι το πρόγραμμα που αναλαμβάνει τον έλεγχο και την εκτέλεση όλων των λειτουργιών που σχετίζονται με την διαχείριση των containers, των εικόνων και των

πόρων του συστήματος. Εκτελείται σαν αυτόνομη διεργασία χωρίς να βρίσκεται υπό τον έλεγχο του χρήστη, δηλαδή λειτουργεί ως deamon για τον Docker.

Docker image

Ένα docker image είναι το σύνολο των απαιτούμενων αρχέτυπων, βιβλιοθηκών, εντολών τα οποία εκτελούνται μέσα σε ενα Docker container. Είναι ένα σύνολο εντολών που χρειάζεται μια εφαρμογή η οποία βασίζεται σε επίπεδο πυρήνα του κεντρικού υπολογιστή. Ο χρήστης έχει τη δυνατότητα να εκτελεί πολλαπλά στιγμιότυπα του ίδιου image.

Docker Container

Αποτελούν στιγμιότυπα των Docker image. Όταν ο χρήστης εκτελεί ένα Docker image μέσω του Docker engine, δημιουργείται ενας container του image. Στο σημείο αυτό προστίθεται ένα νέο επίπεδο το οποίο δίνει τη δυνατότητα εγγραφής και ανάγνωσης και ονομάζεται container layer.

Docker Registry

Αποτελεί το αποθετήριο του Docker στο οποίο μπορεί να γίνει αποθήκευση και κατανομή των Docker images. Αυτό, μπορεί να είναι είτε δημόσιο είτε ιδιωτικό ανάλογα με τη χρήση του.

Docker Compose

Το Docker compose δίνει τη δυνατότητα σε πολύπλοκες εφαρμογές Docker, οι οποίες χρησιμοποιούν παραπάνω από έναν container, να συνδεθούν μεταξύ τους. Μέσω ενός αρχείου YAML, ο χρήστης μπορεί να συνδέσει, να εκτελέσει και να παραμετροποιήσει τους απαιτούμενους containers και πόρους που χρειάζεται η εφαρμογή του. Όλα τα παραπάνω, μπορεί και τα πετυχαίνει με την εκτέλεση μιας και μόνο εντολής Docker compose.

2.2.4 Πλεονεκτήματα της χρήσης του Docker

Παρακάτω, θα αναφέρουμε μερικά από τα σημαντικότερα πλεονεκτήματα της χρήσης του Docker. Τα σημαντικότερα πλεονεκτήματά του είναι τα παρακάτω:

Docker Repository: Στο repository του Docker υπάρχει πλέον ένας τεράστιος αριθμός από εικόνες που μπορεί να πάρει και να χρησιμοποιήσει πολύ εύκολα ο κάθε χρήστης.

Ευέλικτο: Επιτρέπει μεγάλη ευελιξία στη δημιουργία αλλά και στον συνδυασμό μεγάλου αριθμού διαφορετικών Containers.

Ευκολία στη χρήση: Λόγω της μεγάλης απήχησης του Docker έχει δημιουργηθεί ένας πολύ σωστά δομημένος οδηγός που κάνει την χρήση του πολύ απλή.

Δυνατότητα εκτέλεσης σε λειτουργικό σύστημα εκτός Linux: Το Docker έδωσε τη δυνατότητα να γίνεται χρήση των Linux Containers και σε λειτουργικά συστήματα όπως Windows 7, Windows 10, MacOS X.

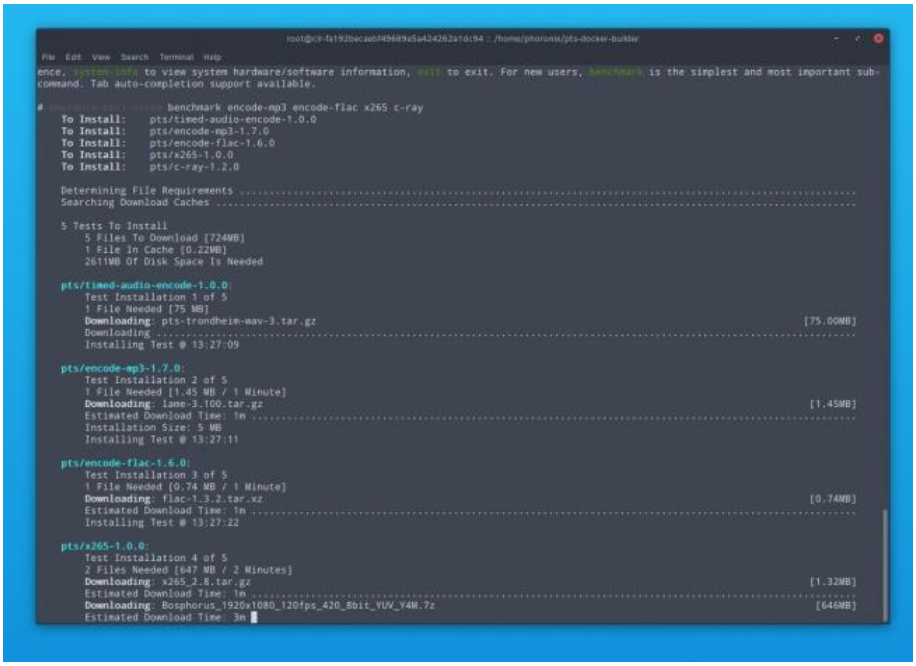
Στη συνέχεια θα γίνει μια αναφορά στην πλατφόρμα ελεύθερου κώδικα που χρησιμοποιήσαμε για την συγκριτική μελέτη της διατριβής μας, το Phoronix Test Suite

2.3 Phoronix Test Suite

Το Phoronix Test Suite αποτελεί μια ολοκληρωμένη πλατφόρμα ελεύθερου κώδικα που δίνει τη δυνατότητα για συγκριτική αξιολόγηση ενός λειτουργικού συστήματος. Η ανάπτυξη έγινε από τους Michael Larabel και Matthew Tippett και εμφανίστηκε για πρώτη φορά το 2008. Βρίσκεται υπό την αιγίδα του GNU GPLv3 και υπάρχει συνεχής ανάπτυξη μέσα από μια μεγάλη κοινότητα. Πλέον θεωρείται από τα πιο αξιόπιστα εργαλεία για συγκριτική αξιολόγηση συστημάτων που βασίζονται σε πυρήνα Linux. Δίνει τη δυνατότητα στον διαχειριστή να επιλέξει μέσα από πάρα πολλούς αυτοματοποιημένους μηχανισμούς σύγκρισης, ανάλογα με την κατηγορία που επιθυμεί. Η εκτέλεση των δοκιμών είναι εύκολη, αναπαραγώγιμη και αυτοματοποιημένη και βλέπουμε στην Εικόνα 4 ένα παράδειγμα εκτέλεσης συγκριτικής δοκιμής.

Υπάρχουν πάνω από 400 διαφορετικές συγκριτικές δοκιμές μέσα από τις οποίες μπορεί να γίνει συγκριτική δοκιμή όσον αφορά την επεξεργαστική ισχύ, την μνήμη, το δίκτυο κλπ. Τέλος, με τη βοήθεια του Openbenchmarking.org ο χρήστης έχει τη δυνατότητα να δει και να συγκρίνει τα αποτελέσματά του με άλλων χρηστών ή διαφορετικών συστημάτων όπως θα δούμε και στη δική μας μελέτη. Ένα μεγάλο πλεονέκτημα του

Phoronix είναι η δυνατότητα του χρήστη να αναπαράγει ακριβώς το ίδιο τεστ ώστε να υπάρχει μεγάλη αξιοπιστία στα αποτελέσματα.

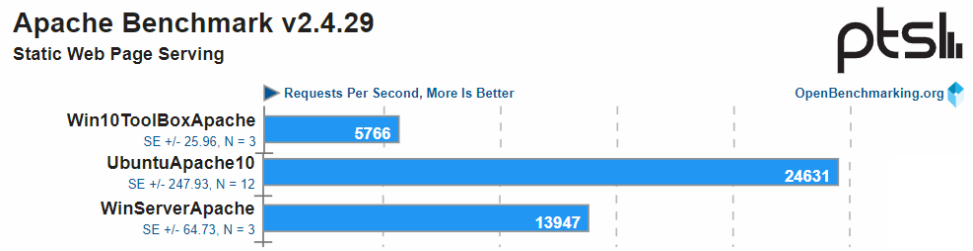


Εικόνα 4. Εκτέλεση Συγκριτικής Δοκιμής

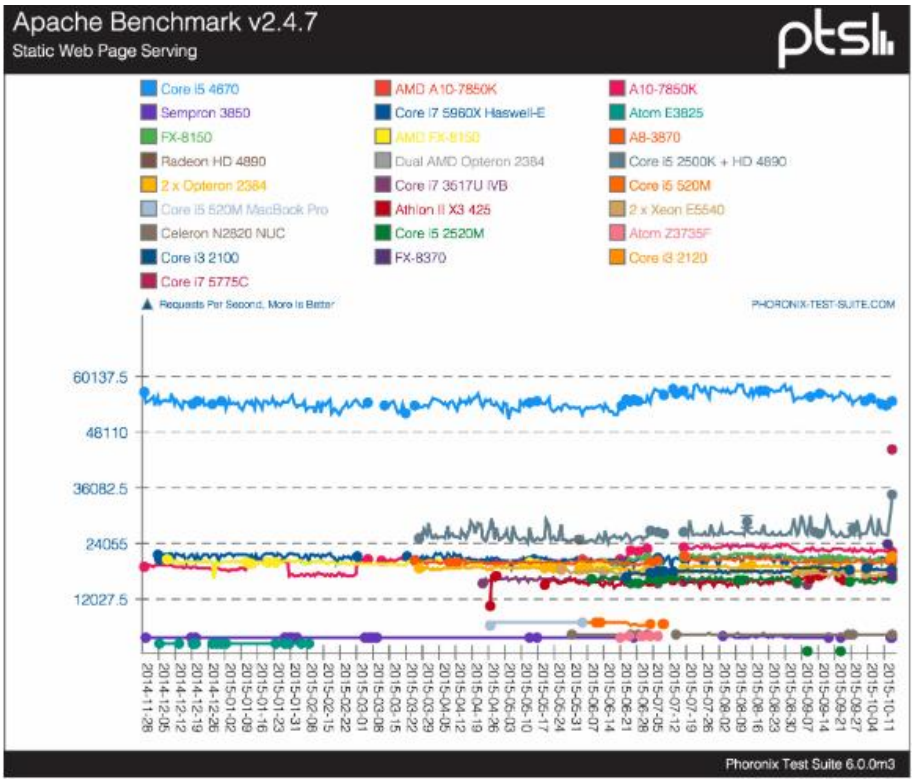
2.4 OpenBenchmarking.org

Το OpenBenchmarking.org αποτελεί μια ανοιχτή, συνεργατική πλατφόρμα δοκιμών μέσω της οποίας το Phoronix Test Suite επεκτείνεται ακόμη περισσότερο όσον αφορά την αυτοματοποίηση και σύγκριση των δοκιμών. Ουσιαστικά αποτελεί ένα αποθετήριο από χρήστες, δοκιμές και αποτελέσματα. Δίνει την δυνατότητα στον χρήστη να εμφανίσει τα αποτελέσματα των δικών του δοκιμών (Εικόνα 5) αλλά και να τα συγκρίνει με αποτελέσματα άλλων χρηστών από όλων τον κόσμο εφόσον το επιθυμεί όπως φαίνεται και στην Εικόνα 6.

Apache Benchmark



Εικόνα 5. Εμφάνιση αποτελεσμάτων συγκριτικής δοκιμής στο Openbenchmark.org



Εικόνα 6. Σύγκριση αποτελεσμάτων διαφορετικών χρηστών

Κεφάλαιο 3 – Πειραματικό Περιβάλλον

Στο τρίτο κεφάλαιο παρουσιάζουμε το περιβάλλον υλοποίησης της συγκριτικής μελέτης και την ανάπτυξη των προτύπων. Γίνεται αναλυτική αναφορά σε όλα τα συστήματα που δημιουργήσαμε και τις διαφορές στις τεχνολογίες που υποστηρίζει το κάθε πρότυπο.

3.1 Μεθοδολογία

Στη συγκριτική μας μελέτη, η μέθοδος που ακολουθήθηκε χωρίζεται στα εξής στάδια.

- Επιλογή συγκεκριμένων συγκριτικών δοκιμών μέσω του Phoronix Suite, ώστε να καλυφθεί όσο το δυνατόν μεγαλύτερο εύρος.
- Υλοποίηση τεσσάρων διαφορετικών συστημάτων, τα οποία είναι τα παρακάτω:
    - Microsoft Windows 10 με Docker For Windows
    - Microsoft Windows 10 with Hyper-V role
    - Microsoft Windows Server 2019
    - Linux Ubuntu
- Κάθε δοκιμή εκτελέστηκε δέκα φορές, ώστε να υπάρχουν αξιόπιστα αποτελέσματα που δεν επηρεάζονται από τον παράγοντα τύχη.
- Εξαγωγή συμπερασμάτων από τα αποτελέσματα και την ανάλυσή τους.

3.1.1 Δείκτες μέτρησης

Οι δείκτες μέτρησης που χρησιμοποιούμε στην συγκριτική μελέτη μας είναι οι παρακάτω:

- **Χρόνος απόκρισης** ονομάζεται ο συνολικός χρόνος που απαιτείται από τη στιγμή που ο χρήστης δημιουργεί ένα αίτημα μέχρι την απάντηση του εξυπηρετητή. Είναι από τους σημαντικότερους δείκτες γιατί παρουσιάζει το χρόνο που χρειάζεται να περιμένει ο χρήστης για να γίνει επεξεργασία του αιτήματός του. Αργός χρόνος απόκρισης συνήθως συνεπάγεται και μη ικανοποιητικό περιβάλλον για τον πελάτη - χρήστη.

- **Χρόνος καθυστέρησης** ονομάζεται ο χρόνος που χρειάζεται για να λάβει την πρώτη απάντηση ο χρήστης, μετά το αίτημά του. Βασική διαφορά με τον χρόνο απόκρισης είναι ότι ο χρόνος απόκρισης εκφράζει το σύνολο του χρόνου καθυστέρησης, συμπεριλαμβάνοντας το χρόνο επεξεργασίας του αιτήματος.
- **Η διέλευση αιτημάτων** αντανακλά τον αριθμό των συναλλαγών που μπορεί να εξυπηρετήσει το σύστημά μας ανά δευτερόλεπτο. Είναι ένας δείκτης που χρησιμοποιείται κατά κόρον στη σημερινή εποχή του Διαδικτύου.
- **Τυπική απόκλιση** μέσω της οποίας θα απορριφθούν μετρήσεις οι οποίες βρίσκονται εκτός ορίων.

3.1.3 Ανάπτυξη πρωτοτύπων

Για την συγκριτική μας μελέτη και την εξαγωγή συμπερασμάτων έγινε ανάπτυξη τεσσάρων διαφορετικών πρωτοτύπων, ώστε να υλοποιήσουμε ρεαλιστικά και αντιπροσωπευτικά πειράματα. Συγκεκριμένα, δημιουργήσαμε τρία διαφορετικά συστήματα με υποδομή της Microsoft και ένα με Linux Ubuntu, ενώ και στις τέσσερις περιπτώσεις, έγινε εγκατάσταση της έκδοσης Docker που υποστηρίζεται κατά περίπτωση. Έπειτα σε όλα τα συστήματα δημιουργήθηκε ένας container με την εφαρμογή Phoronix Test Suite, μέσω της οποίας εκτελέστηκαν οι ίδιες δοκιμές σε κάθε διαφορετικό σύστημα. Τα αποτελέσματα έγιναν μεταφόρτωση στον ιστότοπο openbenchmarking.org και είναι διαθέσιμα προς όλους.

3.2 Ανάπτυξη των προτύπων του πειράματος

3.2.1 Περιβάλλον υλοποίησης

Για λόγους αξιοπιστίας των αποτελεσμάτων επιλέχθηκε η εγκατάσταση και η εκτέλεση όλων των δοκιμών να γίνει στο ίδιο υλικό. Παρακάτω αναφέρονται με λεπτομέρεια όλες οι τεχνικές προδιαγραφές του υλικού που χρησιμοποιήθηκε. Η επιλογή έγινε έτσι ώστε να είναι ένα υλικό το οποίο θα έχει τις απαιτούμενες δυνατότητες για τη δημιουργία εικονικών μηχανών, ώστε να μην επηρεαστεί η αξιοπιστία των αποτελεσμάτων.

MotherBoard	MSI H310M PRO-VH
CPU	Intel Core i3
RAM	8GB DDR4 - 2400
Hard Drive	Samsung SSD 256GB

Πίνακας 1. Τεχνικές Προδιαγραφές

3.3 Docker ToolBox on Windows

Την πρώτη λύση για τη δημιουργία και εκτέλεση Linux Containers σε περιβάλλον Microsoft Windows, την έδωσε το Docker μέσω του εργαλείου που ονόμασε Docker ToolBox on Windows (Εικόνα 7).

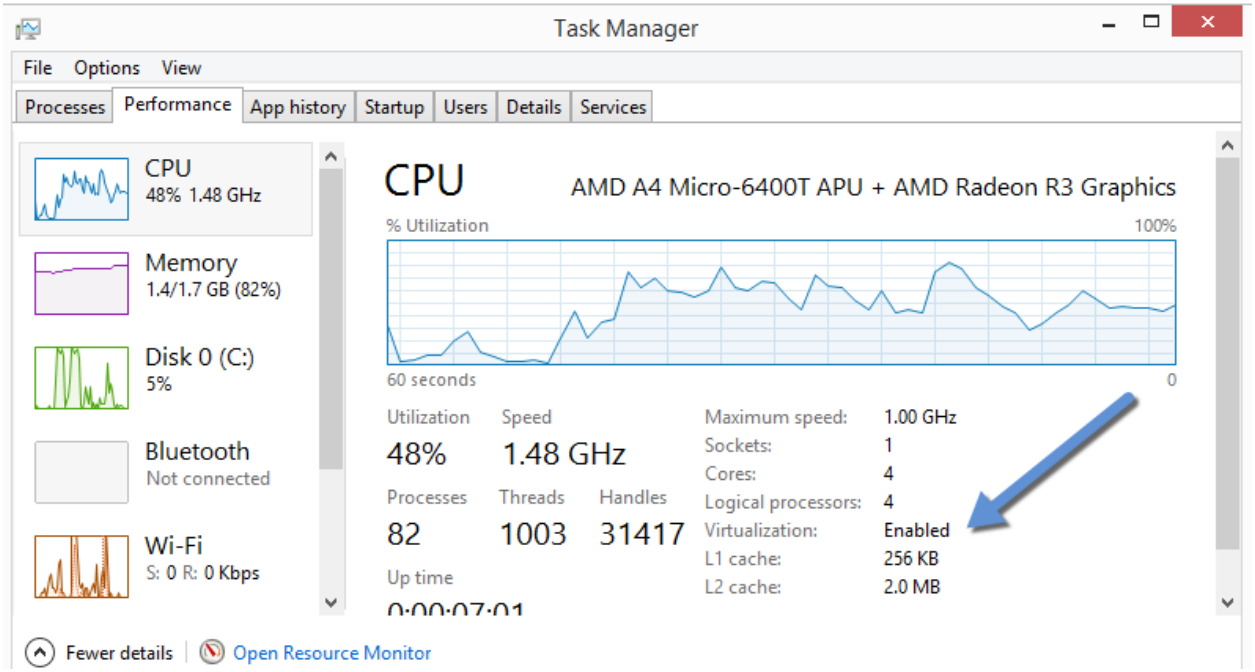
Η έκδοση αυτή περιλαμβάνει τα παρακάτω εργαλεία:

- Docker CLI, ώστε να μπορεί να δημιουργήσει Containers.
- Docker Machine, ώστε να μπορούν να εκτελεστούν εντολές του Docker Engine σε περιβάλλον Windows terminal.
- Docker Compose, ώστε να μπορεί να γίνει χρήση της εντολής docker-compose
- Kitematic, το οποίο αποτελεί την έκδοση του Docker με γραφικό περιβάλλον
- Docker QuickStart κέλυφος, το οποίο αποτελεί ένα προκαθορισμένο κέλυφος με εντολές docker.
- Oracle VM VirtualBox

Εικόνα 7.

3.2.3 Προδιαγραφές Λογισμικού

Η πρώτη εμφάνιση των Linux Containers προϋποθέτει λειτουργικό σύστημα 64 bit, με αποτέλεσμα να μπορεί να εκτελεστούν σε εκδόσεις Windows 7 64 bit και πάνω. Ο τρόπος που λειτουργεί το Docker ToolBox είναι με τη βοήθεια του VirtualBox. Το σύστημά μας θα πρέπει να υποστηρίζει αλλά και να είναι ενεργοποιημένη η Hardware Virtualization Technology. Αυτό μπορεί πολύ εύκολα να το εντοπίσει ο χρήστης εάν υποστηρίζεται από το σύστημά του μέσω του Task Manager (Εικόνα 8), όπως βλέπουμε και στην παρακάτω εικόνα.



Εικόνα 8. Task Manager – Virtualization

3.2.4 Εγκατάσταση του Docker ToolBox

Τα απαιτούμενα αρχεία για την εγκατάσταση του Docker Toolbox on Windows μπορούμε να τα κατεβάσουμε από τον παρακάτω σύνδεσμο:

https://github.com/docker/toolbox/releases

Η έκδοση αυτή έχει πλέον σταματήσει να υποστηρίζεται.

Το πακέτο εγκατάστασης θα εγκαταστήσει όλα τα εργαλεία που προαναφέραμε

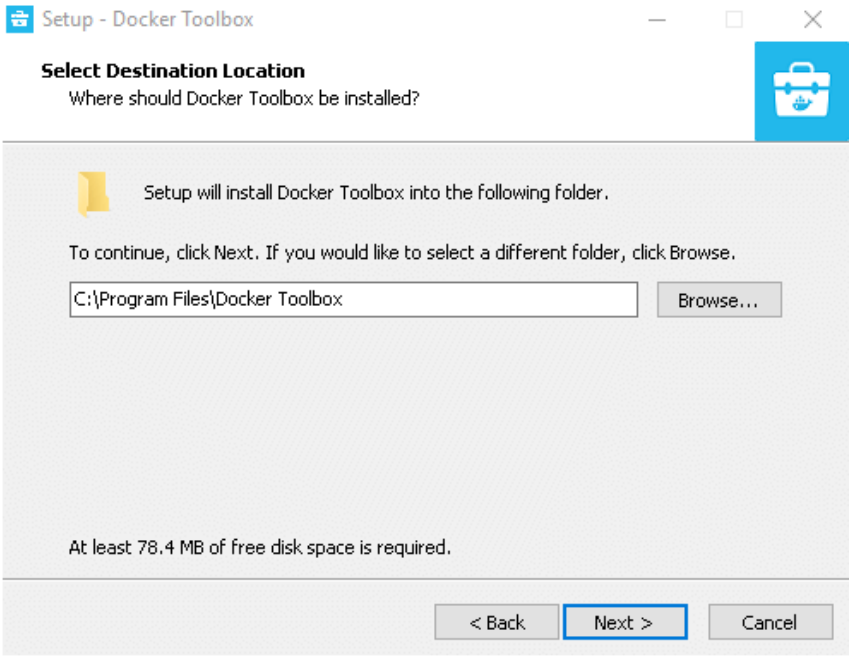
- Docker Client for Windows
- Docker Toolbox management tool and ISO
- Oracle VM VirtualBox
- Git MSYS-git UNIX tools

Διαδικασία εγκατάστασης

Βήμα 1

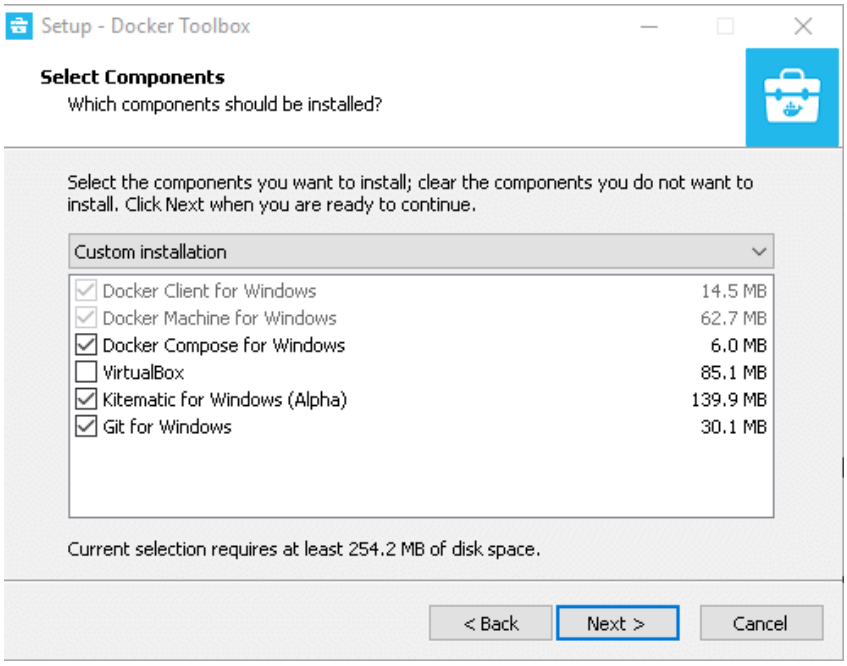
Εικόνα 9. Εγκατάσταση Docker ToolBox

Βήμα 2



Εικόνα 10. Φάκελος Εγκατάστασης της Εφαρμογής

Βήμα 3



Εικόνα 11. Components που γίνονται εγκατάσταση

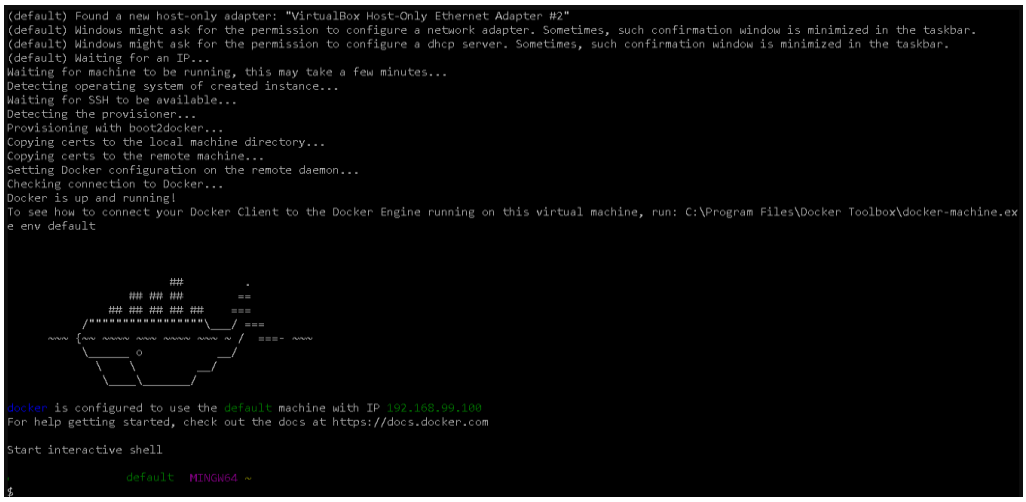
Όπως βλέπουμε και στην Εικόνα 11, γίνεται εγκατάσταση όλων των απαιτούμενων εργαλείων. Αν έχουμε ήδη εγκατεστημένο το Oracle VirtualBox μπορούμε να μην το συμπεριλάβουμε.

Μετά το τέλος της εγκατάστασης έχουν δημιουργηθεί στην επιφάνεια εργασίας τα τρία εικονίδια που απεικονίζονται στην Εικόνα 12.

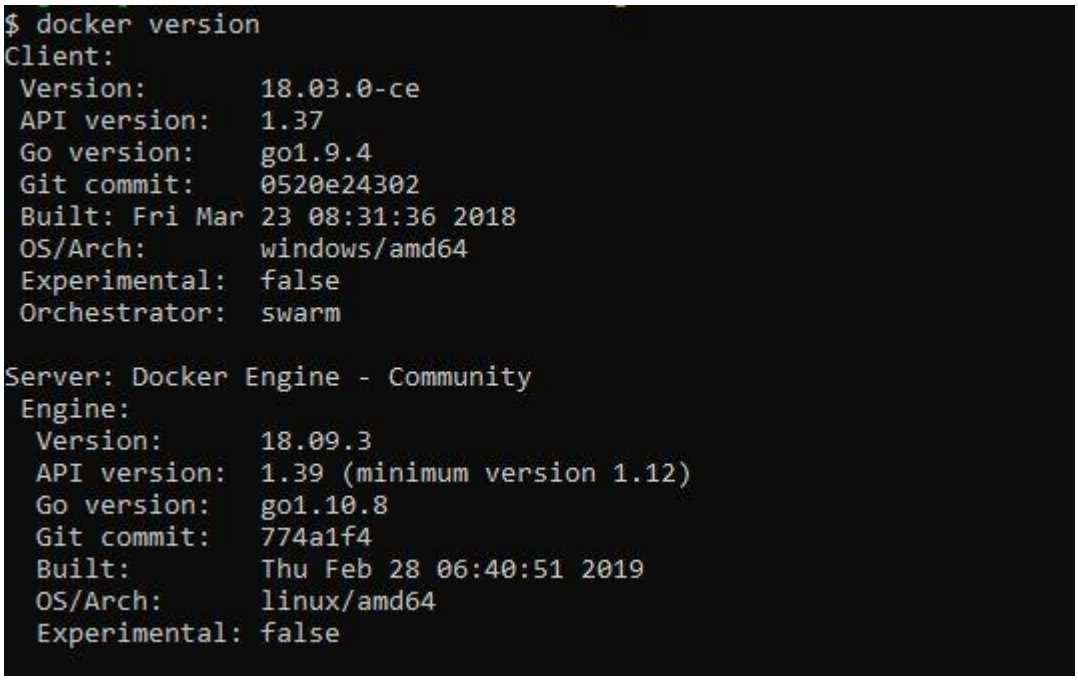
Εικόνα 12. Εικονίδια Επιφάνειας Εργασίας

Επιλέγουμε το εικονίδιο Docker Quickstart Terminal και ξεκινάει η εκτέλεση της μηχανής του Docker. Από δω και πέρα μπορούμε να δημιουργήσουμε και να εκτελέσουμε containers σαν να βρισκόμαστε σε περιβάλλον Linux. Στην Εικόνα 13 βλέπουμε το περιβάλλον εργασίας του Docker ToolBox και στην Εικόνα 14 βλέπουμε την έκδοση του Docker. Εδώ μπορούμε να αντιληφθούμε την ουσιαστική διαφορά του

client ο οποίος είναι σε περιβάλλον Windows και του Server που τρέχει στο VirtualBox ο οποίος έχει δημιουργηθεί.



Εικόνα 13. Περιβάλλον εργασίας του Docker ToolBox.



Εικόνα 14. Η έκδοση των Docker client και Server

3.4 Docker for Windows σε Windows 10 Pro

Η δεύτερη λύση που δημιουργήθηκε από το Docker Community ήταν το Docker For Windows. Πλέον μπορεί να γίνει χρήση των δυνατοτήτων του Hyper-V της Microsoft, πάνω στο οποίο δημιουργείται και εκτελείται μια εικονική μηχανή Linux. Μέσα σε αυτήν την εικονική μηχανή εκτελείται ο Docker.

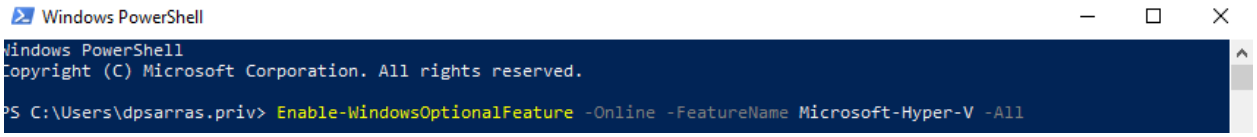
3.4.1 Προδιαγραφές υλικού και λογισμικού

Για να μπορέσει να εγκατασταθεί και να λειτουργήσει ο Docker For Windows πρέπει να τηρούνται οι παρακάτω προϋποθέσεις

- Έκδοση λειτουργικού συστήματος Windows 10 64Bit. Pro, Enterprise ή Education (Build 16299 και πάνω).
- Επεξεργαστή 64bit αρχιτεκτονικής και υποστήριξη SLAT τεχνολογίας
- Ενεργοποιημένα τα χαρακτηριστικά Hyper-V και Containers
- BIOS - Ενεργοποιημένη δυνατότητα για εικονικοποίηση

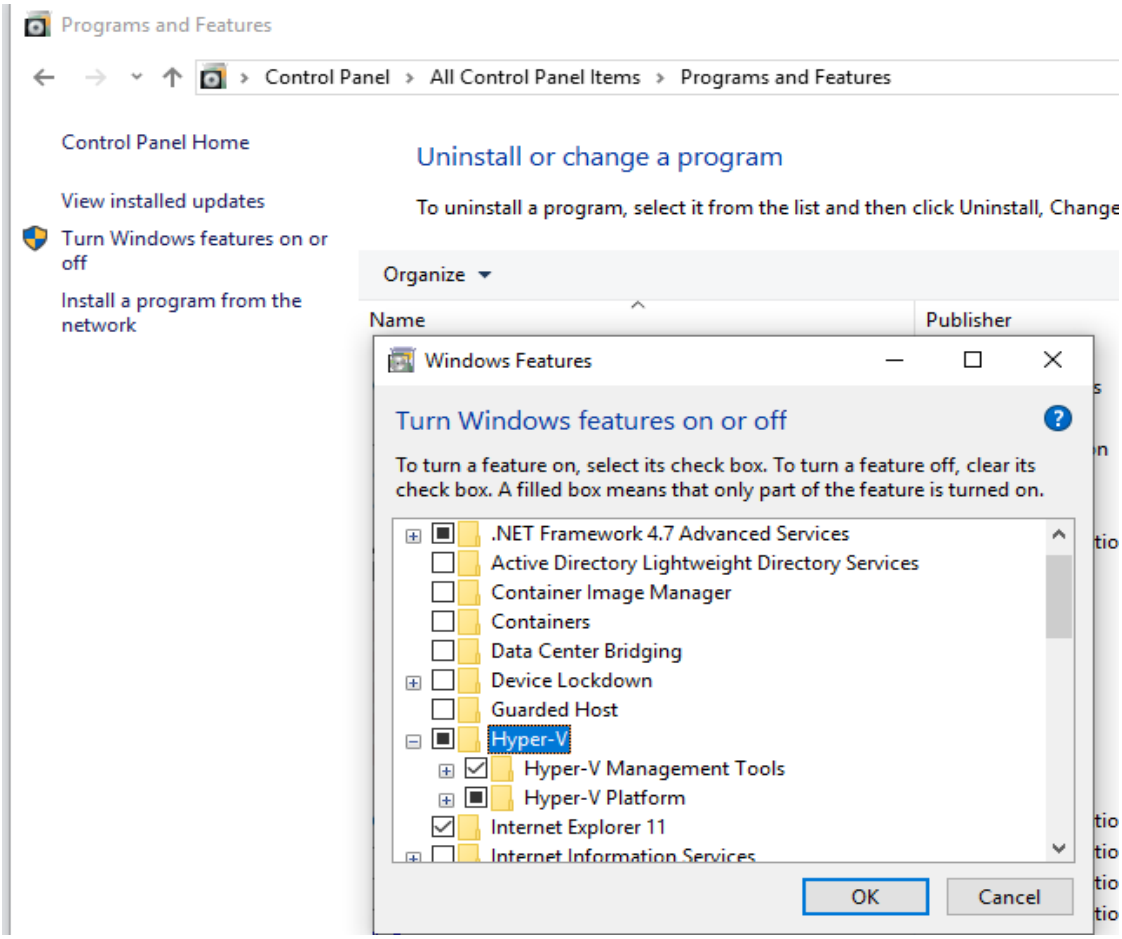
3.4.2 Ενεργοποίηση του Microsoft Hyper-V

Στα Windows 10 Professional η Microsoft δίνει δύο διαφορετικές επιλογές για την ενεργοποίηση αυτού του χαρακτηριστικού. Η πρώτη επιλογή είναι μέσω εντολών PowerShell, την οποία και επιλέξαμε. Η αντίστοιχη εντολή φαίνεται στην Εικόνα 15.



Εικόνα 15. Ενεργοποίηση Hyper-V με PowerShell

Ένας δεύτερος τρόπος και ο πιο σύνηθες είναι μέσα από τον πίνακα ελέγχου και την επιλογή Programs and Features να ενεργοποιήσουμε το Hyper-V και να κάνουμε επανεκκίνηση του συστήματος (Εικόνα 16).



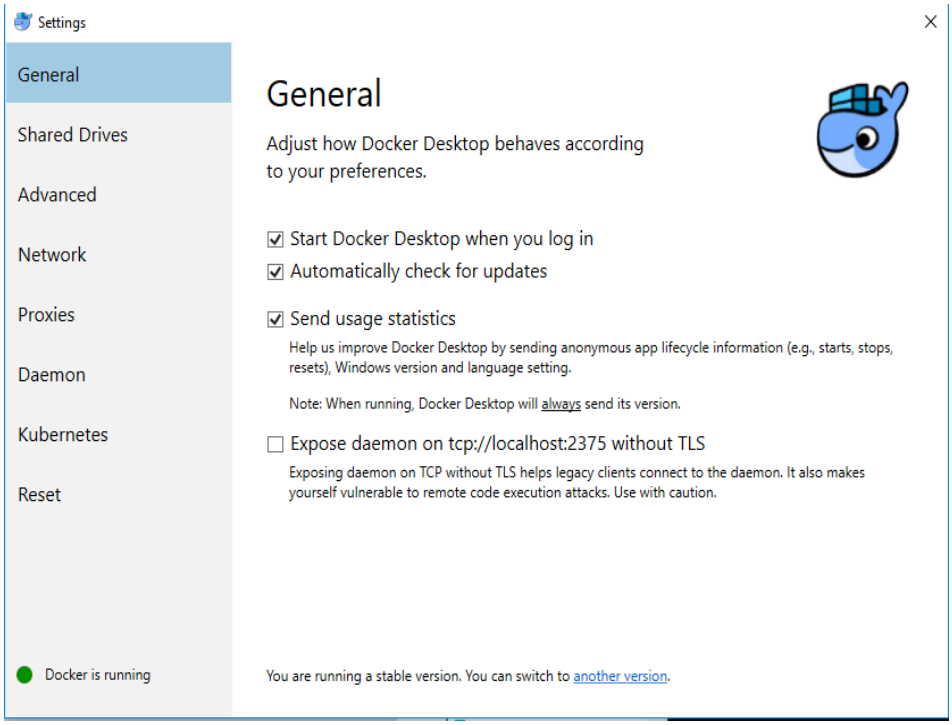
Εικόνα 16. Ενεργοποίηση Hyper-V μέσω του Control Panel των Windows 10.

3.4.3 Εγκατάσταση του Docker For Windows

Το αρχείο εγκατάστασης του Docker For Windows το βρίσκουμε στον παρακάτω σύνδεσμο:

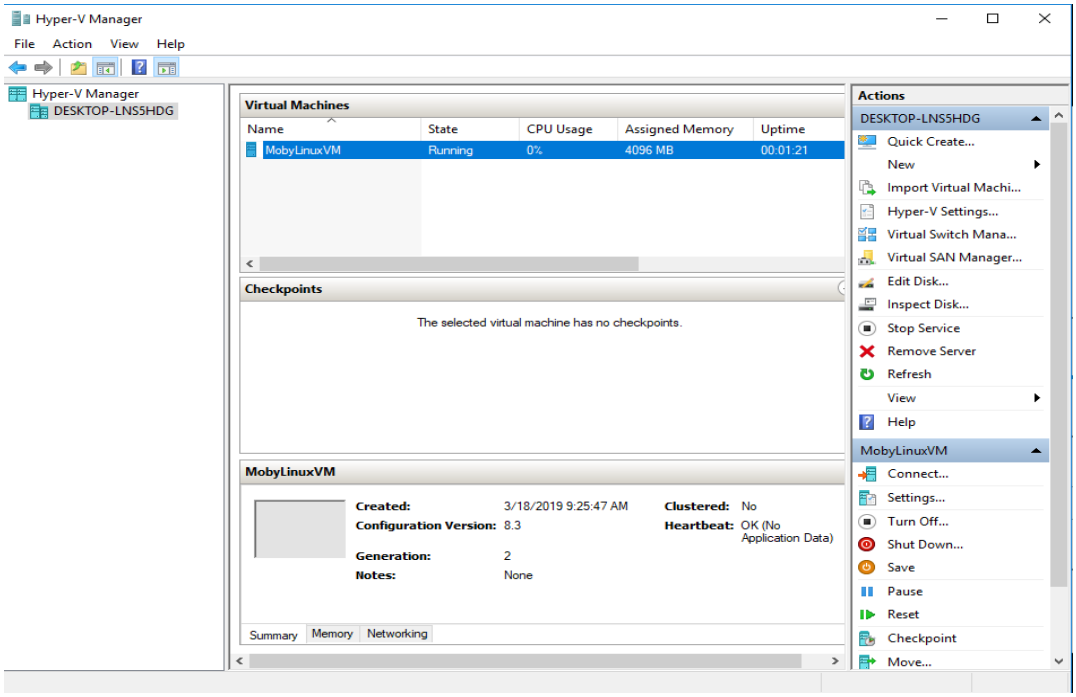
https://download.docker.com/win/stable/Docker%20Desktop%20Installer.exe

Εκτελούμε το αρχείο που κατεβάσαμε και προχωράμε στην εγκατάσταση της εφαρμογής. Στην Εικόνα 17 βλέπουμε το κεντρικό μενού της εφαρμογής και τις διάφορες επιλογές για παραμετροποίηση που μας δίνει.



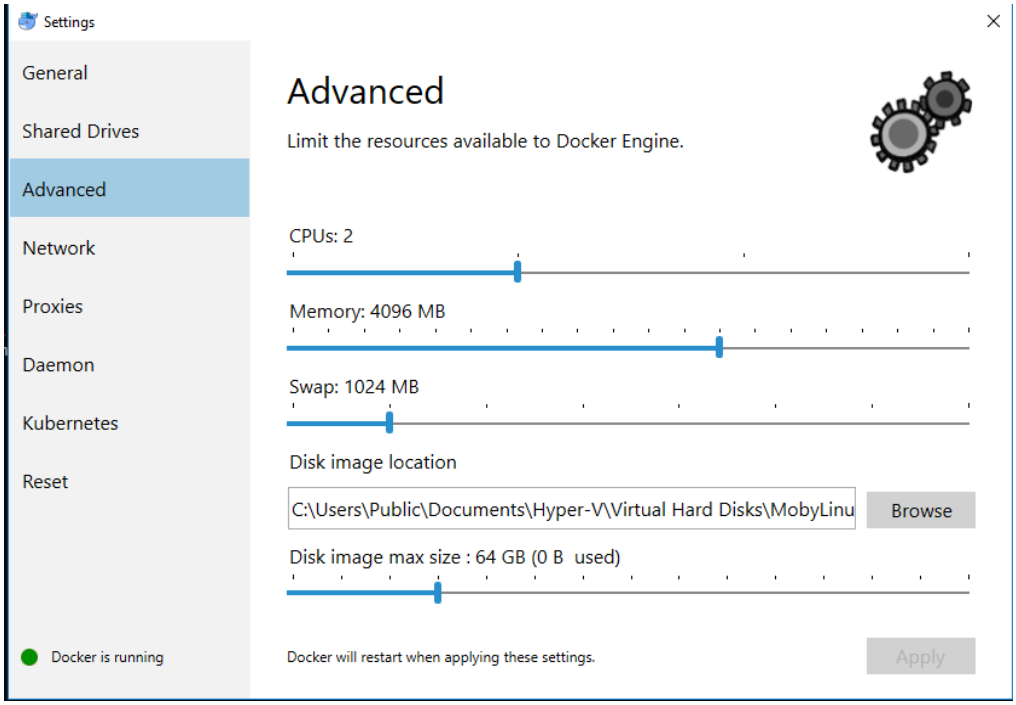
Εικόνα 17. Κεντρικό μενού της εφαρμογής Docker For Windows

Μετά το πέρας της εγκατάστασης μπορούμε να δούμε την εικονική μηχανή που έχει δημιουργηθεί. Έτσι, όπως βλέπουμε και στην Εικόνα 18, αν ανοίξουμε το Hyper-V Manager θα δούμε ότι υπάρχει μια εικονική μηχανή με όνομα MobyLinuxVM που είναι σε λειτουργία.



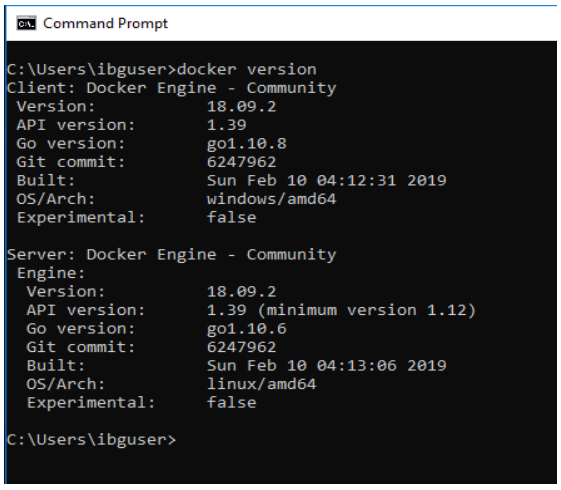
Εικόνα 18. Hyper-V Manager – Εικονική Μηχανή MobyLinuxM

Η διαχείριση της εικονικής μηχανής είναι πολύ απλή. Μπορεί να πραγματοποιηθεί είτε με τον παραδοσιακό τρόπο μέσα από το Hyper-V Manager που διαθέτει η Microsoft αλλά αυτό δεν είναι απαραίτητο. Όλες οι απαιτούμενες ρυθμίσεις που χρειάζονται να γίνουν είτε να επαναπρογραμματιστούν στο μέλλον δίνονται και μέσω των επιλογών του Docker For Windows. Έτσι ανά πάσα στιγμή μπορούμε να δώσουμε περισσότερους πόρους στην εικονική μηχανή, είτε αυτό είναι επεξεργαστική δύναμη, μνήμη RAM, αποθηκευτικό χώρο κλπ. Ακόμη μπορούμε να αλλάξουμε ρυθμίσεις δικτύου εάν αυτό απαιτηθεί. Η Εικόνα 19 μας δείχνει αυτές τις επιλογές και τον εύκολο τρόπο διαχείρισής τους.



Εικόνα 19. Ρυθμίσεις της Εικονικής μηχανής μέσω του Docker For Windows.

Μετά και το τέλος των ρυθμίσεων της εικονικής μηχανής μπορούμε να κατεβάσουμε, να δημιουργήσουμε και να εκτελέσουμε Linux Containers μέσα από το Command Prompt των Windows. Τέλος, στην Εικόνα 20, βλέπουμε την έκδοση του Client και του Server της μηχανής του Docker.



Εικόνα 20. Έκδοση της μηχανής του Docker

3.4 Docker for Windows σε Windows Server 2019

3.4.1 Προδιαγραφές υλικού και λογισμικού

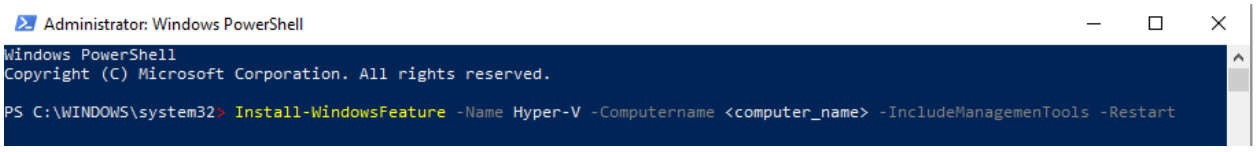
Για να μπορέσει να εγκατασταθεί και να λειτουργήσει ο Docker For Windows πρέπει να τηρούνται οι παρακάτω προϋποθέσεις

- Έκδοση λειτουργικού συστήματος Windows Server 2019
- Επεξεργαστή 64bit αρχιτεκτονικής και υποστήριξη SLAT τεχνολογίας
- Ενεργοποιημένα τα χαρακτηριστικά Hyper-V και Containers
- BIOS - Ενεργοποιημένη δυνατότητα για εικονικοποίηση

3.4.2 Ενεργοποίηση του Microsoft Hyper-V

Η ενεργοποίηση του ρόλου Hyper-V στα Windows Server 2019 γίνεται και πάλι με δύο τρόπους. Ο πρώτος είναι με την εκτέλεση κατάλληλης εντολής σε PowerShell, όπως φαίνεται στην Εικόνα 21.

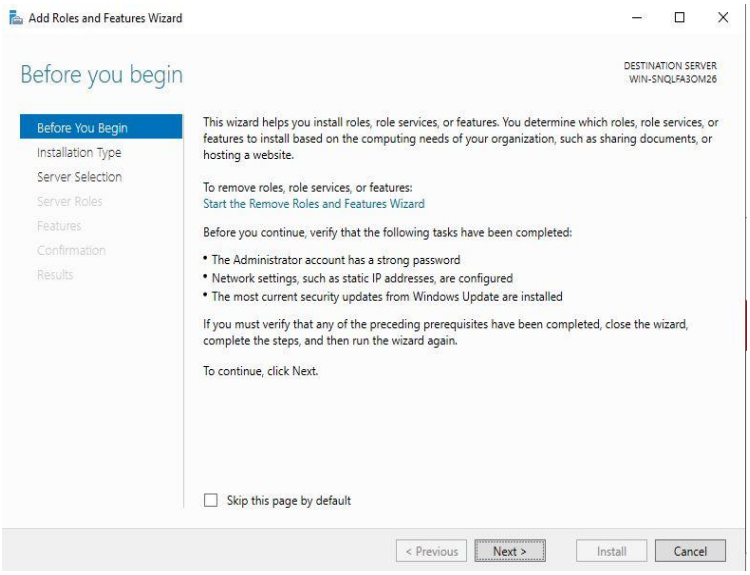
Install-WindowsFeature –Name Hyper-V –ComputerName <computer_name> –IncludeManagementTools –Restart



Εικόνα 21. Ενεργοποίηση Hypervisor

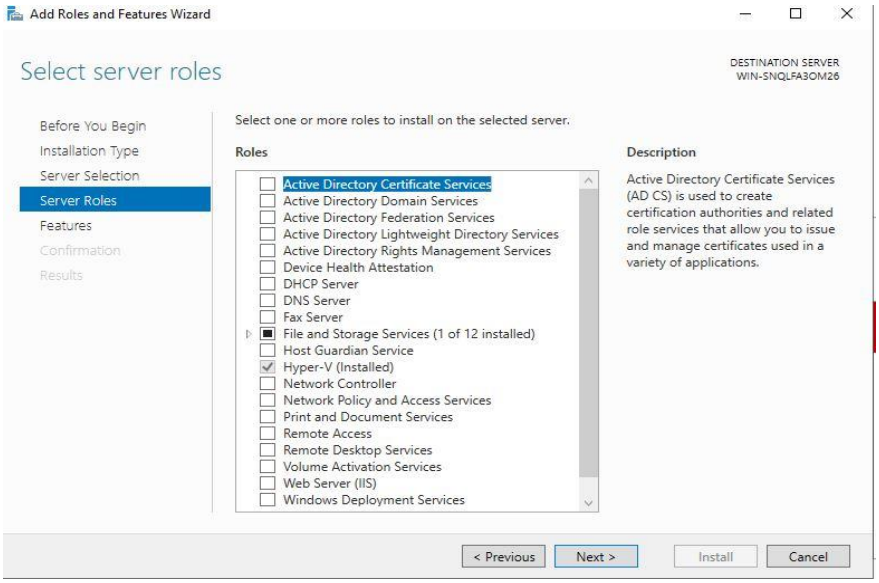
Ο δεύτερος και πιο συνηθισμένος τρόπος είναι μέσα από το μενού διαχείρισης των Windows Server, το **Server Manager**.

Από το μενού του Server manager πατάμε **Add Roles and Features** και μας εμφανίζει το μενού της Εικόνας 22.



Εικόνα 22. Add Roles and Features Wizard των Microsoft Windows Server

Στο επόμενο βήμα βρίσκουμε και επιλέγουμε το Hyper-V. Σε αυτό το βήμα ο οδηγός θα βρει και θα προσθέσει ότι άλλο είναι απαραίτητο για να γίνει η εγκατάσταση του Hyper-V. Στην Εικόνα 22 βλέπουμε ότι στο δικό μας σύστημα έχει ήδη πραγματοποιηθεί εγκατάσταση του Hyper-V μέσω του PowerShell.



Εικόνα 22. Μενού επιλογής του Ρόλου Hyper-V

3.5 Linux Container on Ubuntu

Κατά τη διάρκεια της συγκριτικής μας μελέτης έγινε επιλογή μιας διανομής Ubuntu για να γίνει η εγκατάσταση και η εκτέλεση του Docker. Η επιλογή αυτή έγινε με βάση το κριτήριο της πιο δημοφιλούς δημοφιλούς διανομής αυτή τη χρονική περίοδο.

Προαπαιτούμενο για να μπορέσουμε να εγκαταστήσουμε και να εκτελέσουμε τον Docker είναι να έχουμε μια απο τις παρακάτω εκδόσεις Ubuntu.

- Ubuntu Focal 20.04 (LTS)
- Ubuntu Eoan 19.10
- Ubuntu Bionic 18.04 (LTS)
- Ubuntu Xenial 16.04 (LTS)

Ακόμη η μηχανή του Docker υποστηρίζει αρχιτεκτονικές x86_64, amd64, armhf και arm64.

3.5.1 Εγκατάσταση του Docker

Για την εγκατάσταση εκτελούμε τις παρακάτω εντολές.

Για να κάνουμε χρήση του repository σε HTTPS κάνουμε update το πακέτο apt.

$ sudo apt-get update

**$ sudo apt-get install **

**apt-transport-https **

**ca-certificates **

**curl **

**gnupg-agent **

software-properties-common

Έπειτα προσθέτουμε το αυθεντικό GPG κλειδί του Docker με τις παρακάτω εντολές.

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

Ενημερώνουμε το repository στην έκδοση stable.

**$ sudo add-apt-repository **

**“deb [arch=amd64] https://download.docker.com/linux/ubuntu **

**$(lsb_release -cs) **

stable”

Εγκατάσταση του docker engine

$ sudo apt-get update

$ sudo apt-get install docker-ce docker-ce-cli containerd.io

Έλεγχος ορθής λειτουργίας του docker

$ sudo docker run hello-world

Στην επόμενη ενότητα γίνεται αναφορά στην εγκατάσταση του εργαλείου που χρησιμοποιήσαμε για τη δημιουργία και εκτέλεση συγκριτικών δοκιμών και αυτό είναι το Phoronix Test Suite.

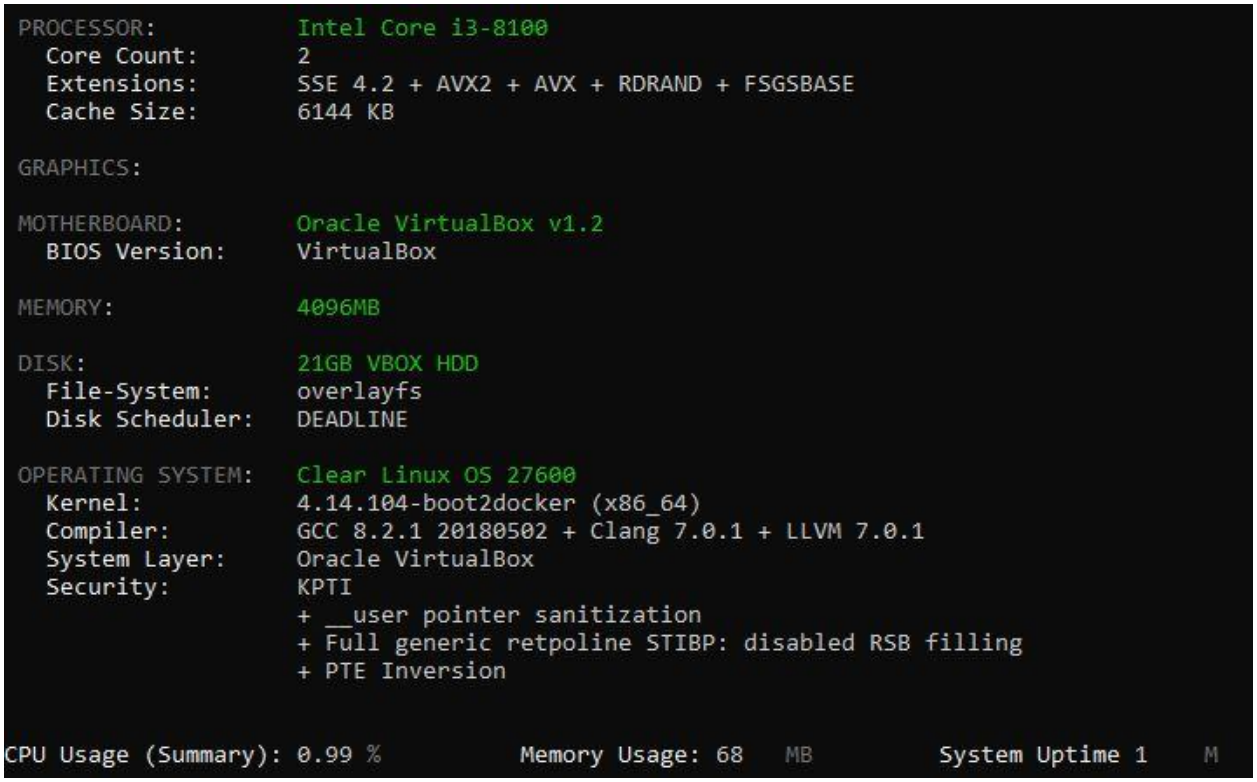
3.6 Εγκατάσταση του Phoronix Test Suite Container

Η εγκατάσταση του Phoronix Test Suite είναι κοινή για όλα τα πρότυπα που δημιουργήσαμε.

Η εγκατάσταση του είναι πολύ απλή επειδή ο container βρίσκεται στο repository του Docker. Εκτελούμε την παρακάτω εντολή και περιμένουμε μέχρι να τελειώσει η διαδικασία:

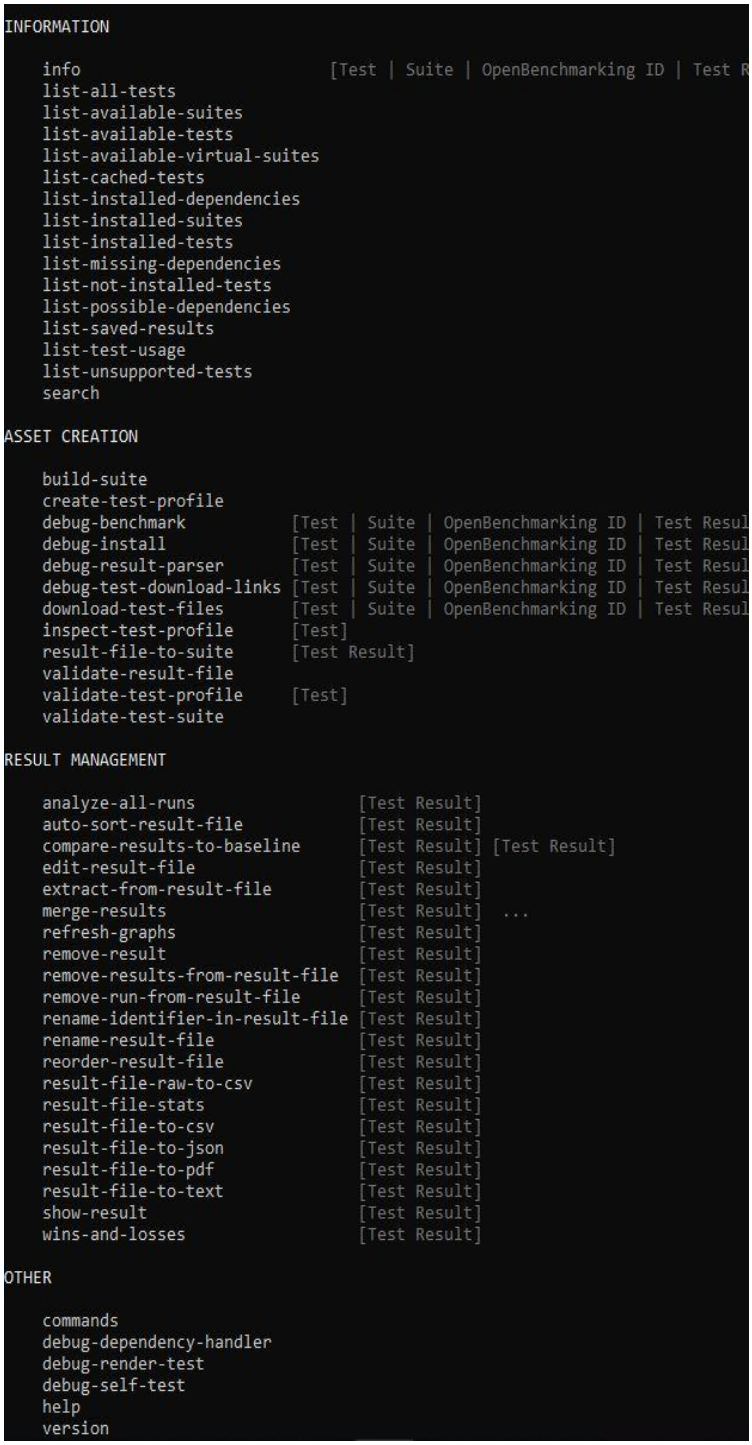
#docker run -it phoronix/pts

Μετά το τέλος της διαδικασίας η εφαρμογή ξεκινάει και μας δείχνει τα χαρακτηριστικά του συστήματος στο οποίο εκτελείται, όπως φαίνεται στην Εικόνα 23.



Εικόνα 23. Κατά την είσοδο στην εφαρμογή μας παρουσιάζονται τα χαρακτηριστικά του συστήματος στο οποίο εκτελείται.

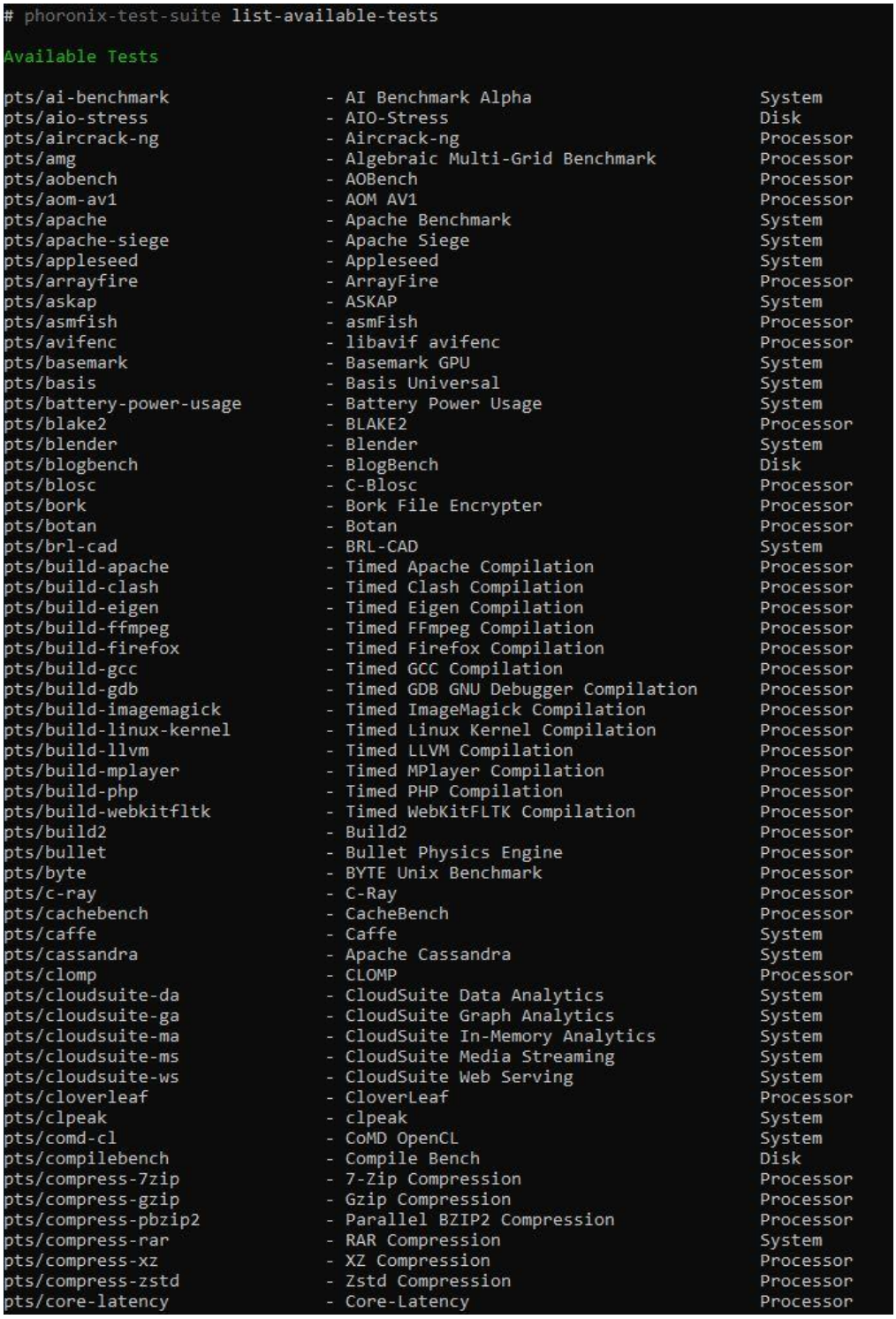
Η σουίτα διαθέτει ένα πολύ μεγάλο αριθμό από εντολές που μπορεί να εκτελέσει ο χρήστης ώστε να παραμετροποιήσει όπως ακριβώς επιθυμεί το σύστημα του πριν την εκτέλεση δοκιμών. Στην Εικόνα 24 βλέπουμε μερικές από τις εντολές που μπορούμε να εκτελέσουμε.



Εικόνα 24. Μερικές από τις Διαθέσιμες εντολές της σουίτας

Εκτελούμε την παρακάτω εντολή, ώστε να εμφανίσει τις διαθέσιμες συγκριτικές δοκιμές (Εικόνα 25).

 # phoronix-test-suite list-available-tests



Εικόνα 25. Μερικές από τις διαθέσιμες συγκριτικές δοκιμές.

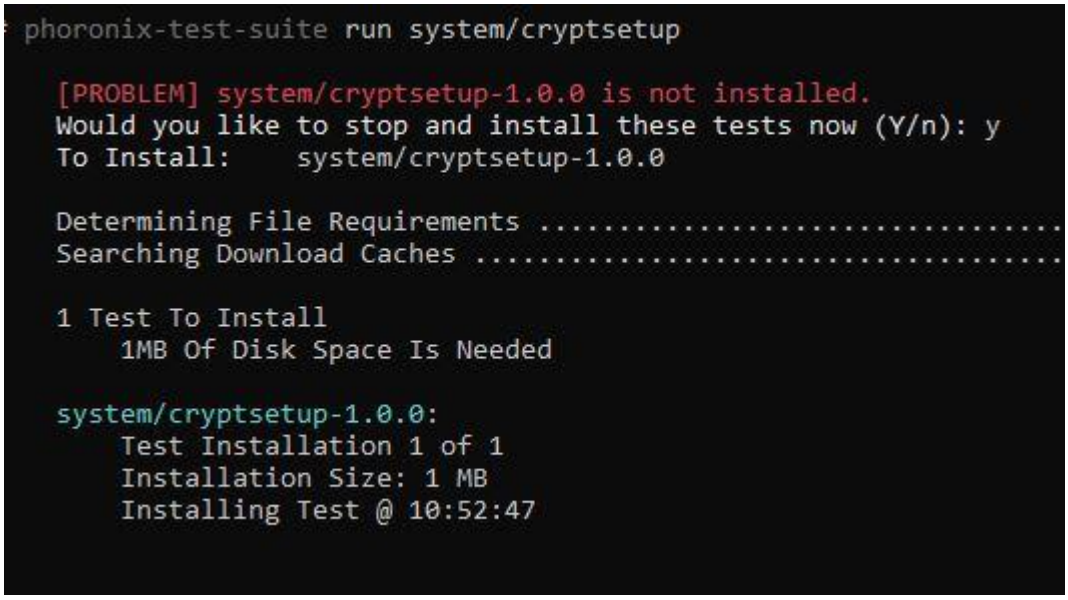
3.6.1 Εγκατάσταση και εκτέλεση συγκριτικής δοκιμής

Η εγκατάσταση και εκτέλεσης μιας συγκριτικής δοκιμής είναι πολύ εύκολη. Με την παρακάτω εντολή θα εκτελέσουμε μια δοκιμή για κρυπτονομίσματα. Όπως είναι

φυσικό, εάν έχει γίνει ξανά εκτέλεση της συγκεκριμένης δοκιμής δεν χρειάζεται να ξαναγίνει εγκατάσταση.

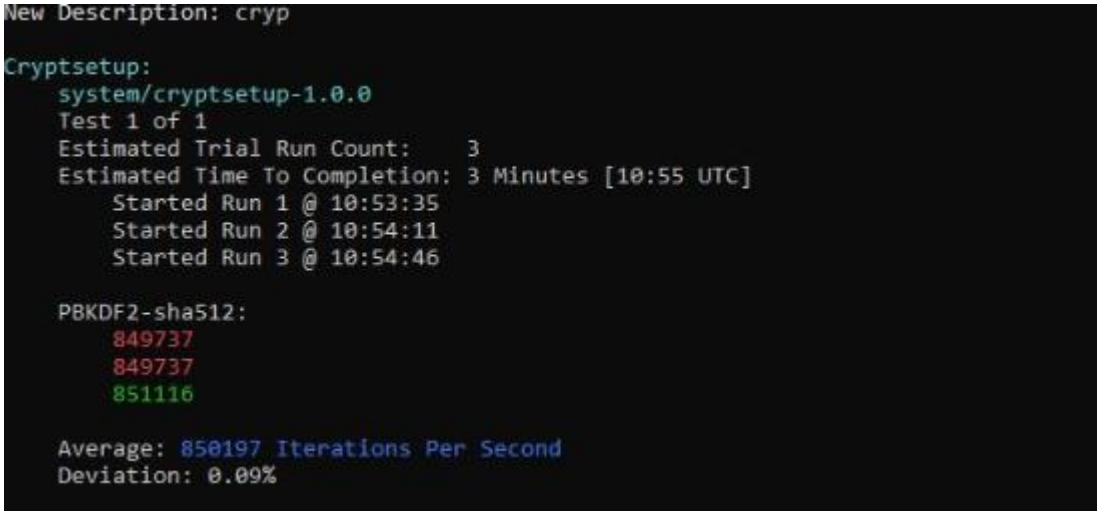
#phoronix-test-suite run system/cryptsetup

Όπως βλέπουμε στην εικόνα 26, με την παραπάνω εντολή γίνεται η εγκατάσταση της συγκριτικής δοκιμής. Ο χρόνος εγκατάστασης εξαρτάται από το μέγεθος της κάθε δοκιμής.



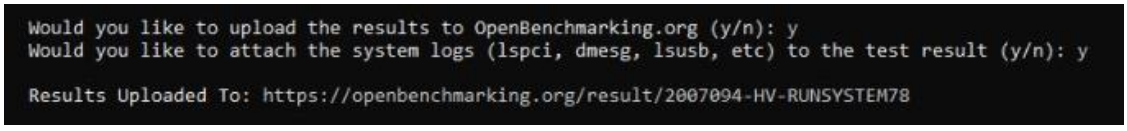
Εικόνα 26. Εγκατάσταση Συγκριτικής δοκιμής.

Μετά το τέλος της δοκιμής μας εμφανίζει τα αποτελέσματα και την τυπική απόκλιση. Κάθε δοκιμή μπορεί να αποτελείται από διαφορετικά στάδια. Στην Εικόνα 27 βλέπουμε ότι η συγκριτική δοκιμή cryptsetup έχει τρία στάδια και ο συνολικός χρόνος εκτέλεσης ήταν περίπου 2 λεπτά.



Εικόνα 27. Εκτέλεση της συγκριτικής δοκιμής CryptSetup

Έπειτα, έχουμε τη δυνατότητα να κάνουμε άμεση σύγκριση των αποτελεσμάτων με διάφορα συστήματα άλλων χρηστών που έχουν ανεβάσει τα δικά τους αποτελέσματα στην ιστοσελίδα του Openbenchmark.org. Αρκεί να πληκτρολογήσουμε yes στην ερώτηση, αν επιθυμούμε να γίνει μεταφόρτωση των αποτελεσμάτων μας στην ιστοσελίδα του Openbenchmark.org. Εφόσον πληκτρολογήσουμε yes δημιουργείται ένας σύνδεσμος(Εικόνα 28) στον οποίο υπάρχουν τα αποτελέσματά μας.

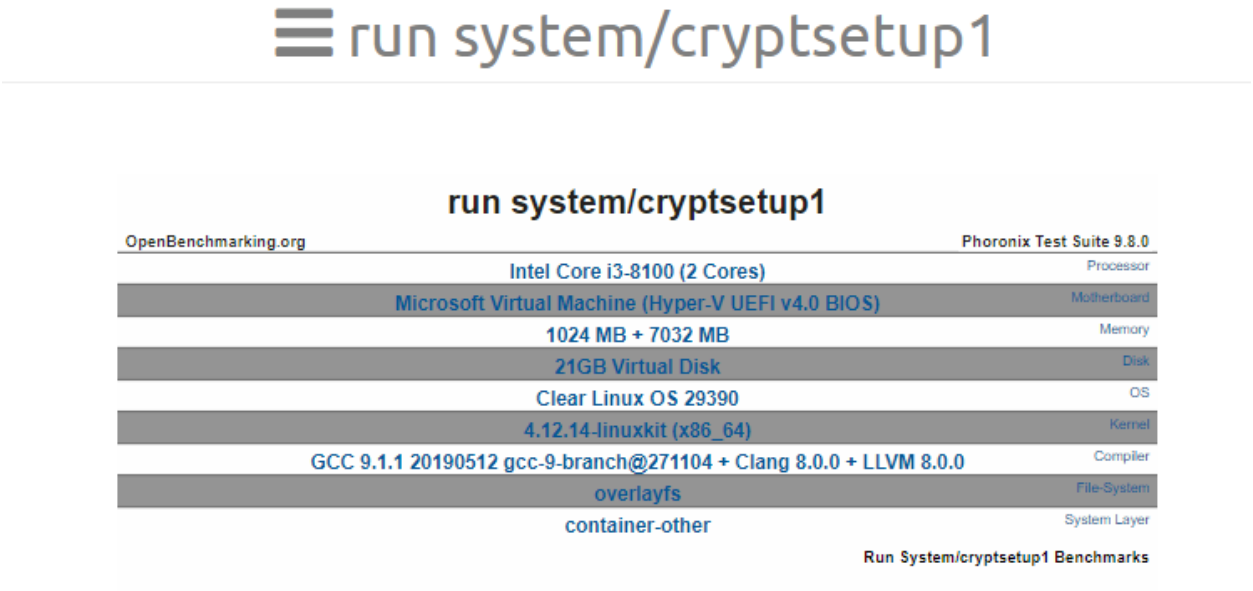


Εικόνα 28. Δημιουργία του συνδέσμου στο Openbenchmark.org με τα αποτελέσματά μας

Ο σύνδεσμος που δημιούργησε η εφαρμογή,

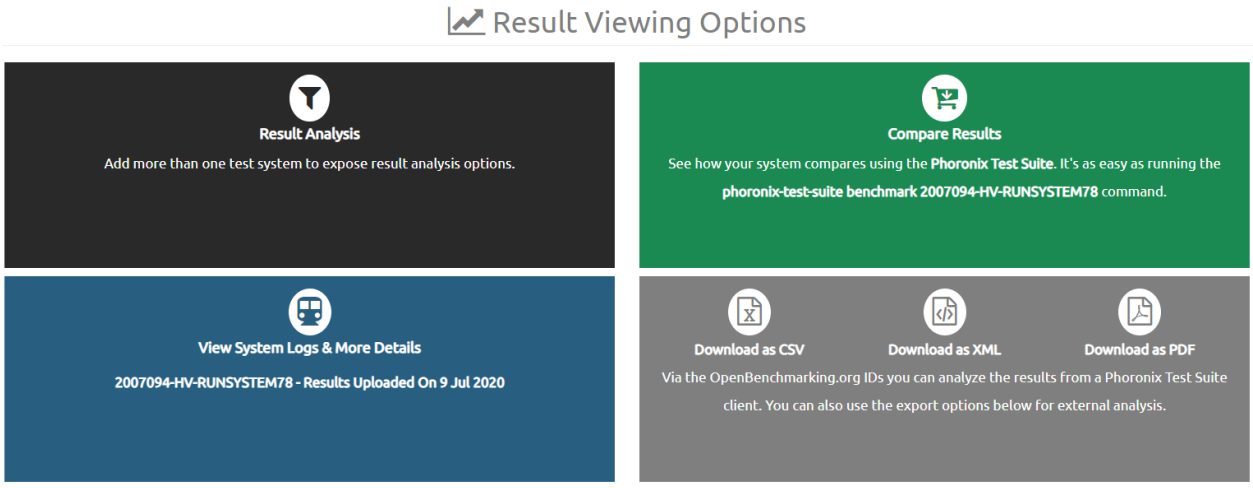
https://openbenchmarking.org/result/2007094-HV-RUNSYSTEM78

Μας δίνει όλες τις πληροφορίες που χρειαζόμαστε για τη συγκριτική μελέτη που εκτελέσαμε. Έτσι, όπως μπορούμε να δούμε και στην Εικόνα 29, υπάρχουν οι πληροφορίες του συστήματος που εκτελέστηκε η δοκιμή με πληροφορίες όπως το υλικό αλλά και το λογισμικό.



Εικόνα 29. Ο τρόπος εμφάνισης των χαρακτηριστικών του υλικού και του λογισμικού μέσω της ιστοσελίδας Openbenchmark.org.

Ακόμη, όπως βλέπουμε και στην Εικόνα 30, υπάρχει πληθώρα επιλογών για εξαγωγή των αποτελεσμάτων σε xml, csv ή pdf μορφή, αλλά και η δυνατότητα να εκτελεστεί ακριβώς η ίδια συγκριτική δοκιμή σε άλλα συστήματα της επιλογής μας. Η επιλογή αυτή θεωρείται πολύ σημαντική διότι έτσι μπορούμε να κάνουμε μια προσομοίωση της ίδιας δοκιμής, χωρίς να μας ανησυχεί η ακεραιότητα των αποτελεσμάτων.

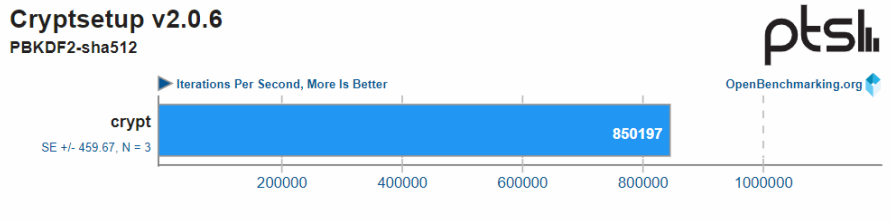


Εικόνα 30. Διάφορες επιλογές για την επεξεργασία των αποτελεσμάτων

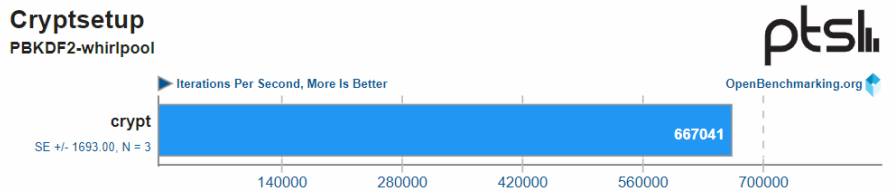
Η μορφή παρουσίασης των αποτελεσμάτων είναι ξεκάθαρη και βοηθάει στην άμεση και εύκολη σύγκριση διαφορετικών συστημάτων (Εικόνα 31).

Overview

Cryptsetup



Cryptsetup



Εικόνα 31. Παρουσίαση των αποτελεσμάτων

Κεφάλαιο 4 - Πειραματική Ανάλυση

4.1 Μεθοδολογία

Βασικός στόχος της διπλωματικής εργασίας, είναι η συγκριτική μελέτη των επιλογών για δημιουργία, εκτέλεση και διαχείριση Linux Containers σε περιβάλλον Microsoft Windows.

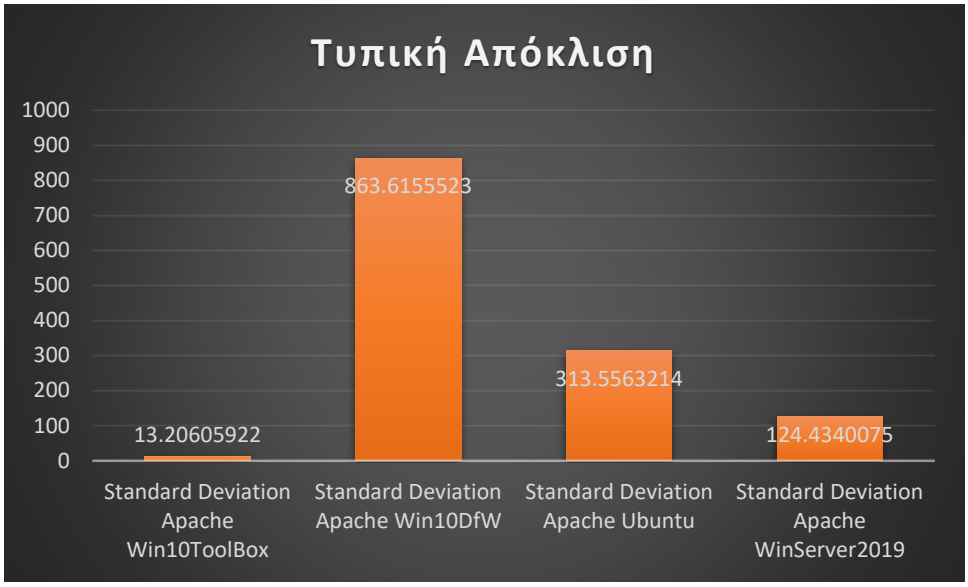
Αρχικά, για την εξαγωγή αποτελεσμάτων, επιλέχθηκαν έξι διαφορετικές πειραματικές δοκιμές. Οι πειραματικές δοκιμές επιλέχθηκαν με βάση την δημοφιλία τους κατά τη διάρκεια συγκριτικών δοκιμών διαφορετικών συστημάτων. Στη συνέχεια, κάθε ένα πείραμα εκτελείται δέκα συνεχόμενες φορές στο κάθε ένα πρότυπο που έχουμε δημιουργήσει ώστε να εξασφαλιστεί η αξιοπιστία των αποτελεσμάτων. Για λόγους αξιοπιστίας η σουίτα δοκιμών που έχουμε επιλέξει, το Phoronix Benchmark Suite μας δίνει τη δυνατότητα να κάνουμε προσομοίωση της πρώτης δοκιμής σε όλα. Έτσι έχουμε ακριβώς τα ίδια δεδομένα σε κάθε μια εκτέλεση. Και στις τρεις εικονικές μηχανές που δημιουργήσαμε έχουμε δώσει τους ίδιους πόρους δηλαδή, δύο επεξεργαστές, 4GB μνήμη και 64GB σκληρό δίσκο. Τέλος η εκτέλεση του Phoronix Test Suite γίνεται με τους ίδιους πόρους όπως και οι εικονικές μηχανές ώστε να έχουμε μια αξιόπιστη σύγκριση των αποτελεσμάτων μας. Παρακάτω γίνεται μια αναφορά σε όλες τις συγκριτικές μελέτες που επιλέχθηκαν να εκτελεστούν.

4.2 Δοκιμή 1: Apache Benchmark

Το Apache Benchmark αποτελεί ένα εργαλείο για συγκριτική μελέτη ενός διακομιστή Apache Hypertext Transfer Protocol (HTTP). Είναι σχεδιασμένο έτσι ώστε να δίνει την αποτύπωση της απόδοσης μιας συγκεκριμένης εγκατάστασης διακομιστή Apache. Αυτό το επιτυγχάνει αποτυπώνοντας τον αριθμό των αιτήσεων ανά δευτερόλεπτο που μπορεί να εξυπηρετήσει. Στο συγκεκριμένο πείραμα ζητάμε την μέτρηση του χρόνου για την διαχείριση ενός εκατομμυρίου αιτήσεων με τα εκατό από αυτά να εκτελούνται ταυτόχρονα.

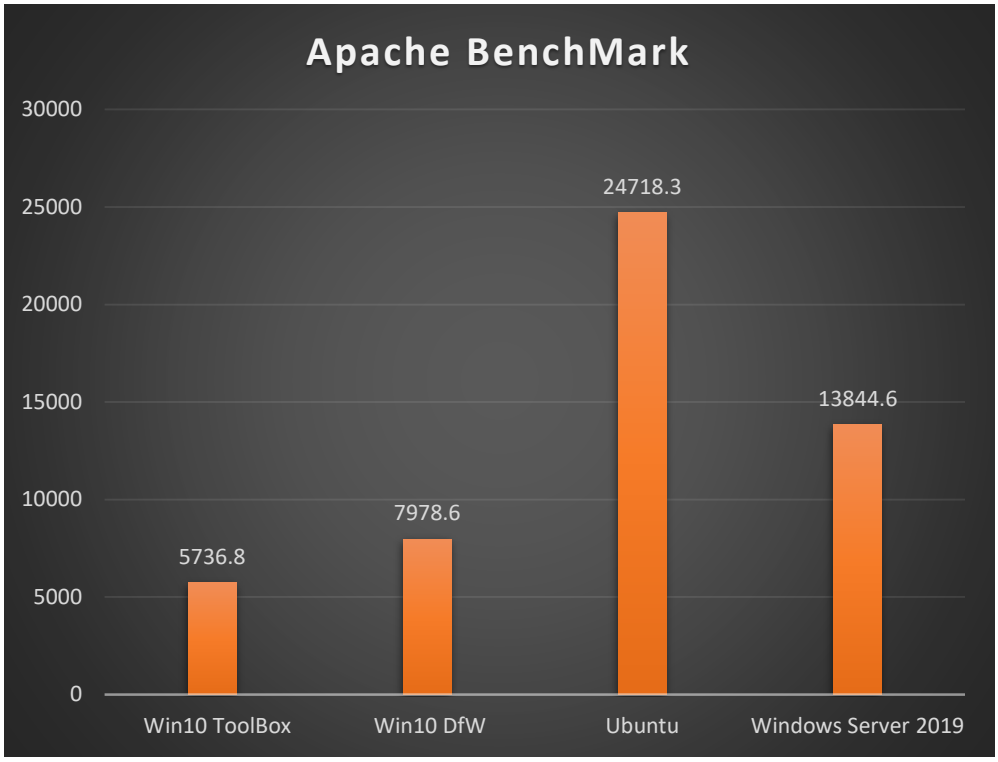
Στη δική μας πειραματική προσέγγιση έγινε χρήση της έκδοσης Apache Benchmark v2.4.29 η οποία ήταν η τελευταία διαθέσιμη κατά την συγγραφή της παρούσας διπλωματικής εργασίας. Στην περίπτωση του Docker Toolbox όπως βλέπουμε και στην

Εικόνα 32. η τυπική απόκλιση έμεινε στο χαμηλότερο επίπεδο, με τιμή 13,2 ενώ την μεγαλύτερη τιμή έχουμε στην περίπτωση του Docker for Windows σε Windows 10 Professional, 863,6. Στα Ubuntu η τυπική απόκλιση ήταν 313,5 και τέλος στα Windows Server 124.4.



Εικόνα 32. Τυπική Απόκλιση αποτελεσμάτων Apache Benchmark

Όπως παρατηρούμε και στην Εικόνα 33 είναι ξεκάθαρο σε ποιο από τα συστήματά μας έγινε η μεγαλύτερη εξυπηρέτηση αιτήσεων από τον Apache Server και αυτό είναι τα Ubuntu με μέση τιμή 24718,3 αιτήματα ανά δευτερόλεπτο. Στη συνέχεια ακολουθούν τα Windows Server με 13844,6, τα Windows 10 Pro με 7978,6 και τέλος τα Windows 10 με το Docker ToolBox με 5736,8 αιτήματα ανά δευτερόλεπτο.



Εικόνα 33. Αποτελέσματα Apache BenchMark

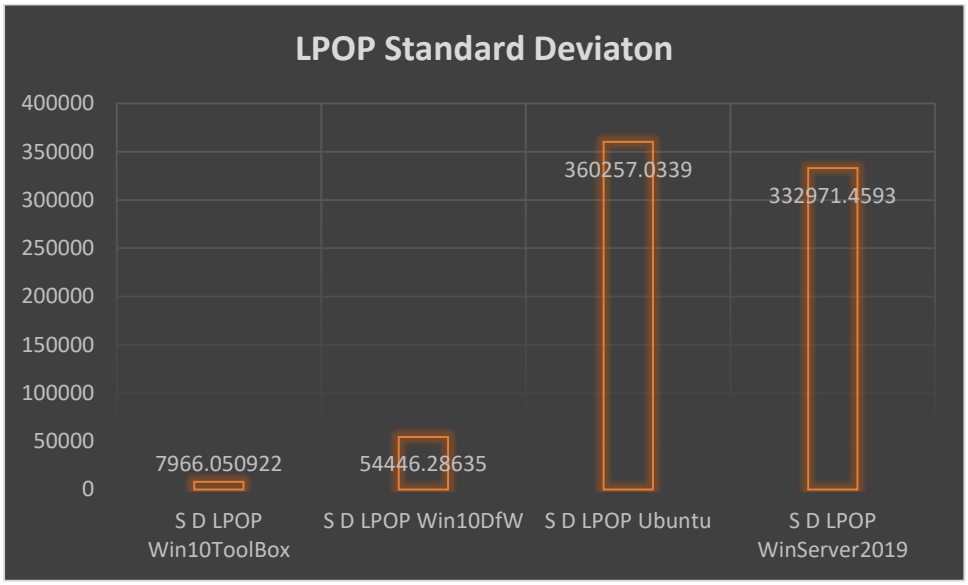
4.3 Δοκιμή 2: Redis Benchmark

Το Redis αποτελεί μια δομή δεδομένων ανοιχτού κώδικα κάτω από την ομπρέλα του BSD, οι οποία χρησιμοποιείται ως βάση δεδομένων, ή σαν μνήμη cache. Υποστηρίζει πολλές διαφορετικές δομές δεδομένων όπως πίνακες κατακερματισμού, λίστες, hyperLogLogs, spatial indexes και άλλα. Αποτελεί την πιο διαδεδομένη key-value βάση δεδομένων γιατί μπορεί ταυτόχρονα να θεωρηθεί μνήμη αποθήκευσης, αλλά και cache. Είναι υλοποιημένη με τέτοιο τρόπο ώστε τα δεδομένα να είναι πάντα σε κατάσταση μόνιμης επεξεργασίας στην κύρια μνήμη ενός υπολογιστή.

Μέσω της σουίτας του Phoronix θα εκτελέσουμε μια δοκιμή απόδοσης σε μια δομή δεδομένων Redis. Το Redis Benchmark, προσομοιώνει την εκτέλεση εντολών που θα γίνουν από έναν αριθμό N πελατών οι οποίοι την ίδια στιγμή θα στέλνουν ένα συνολικό αριθμό M ερωτημάτων. Στην παρούσα συγκριτική μελέτη επειδή εξετάζουμε μια δομή δεδομένων έχουμε πέντε ξεχωριστές τιμές.

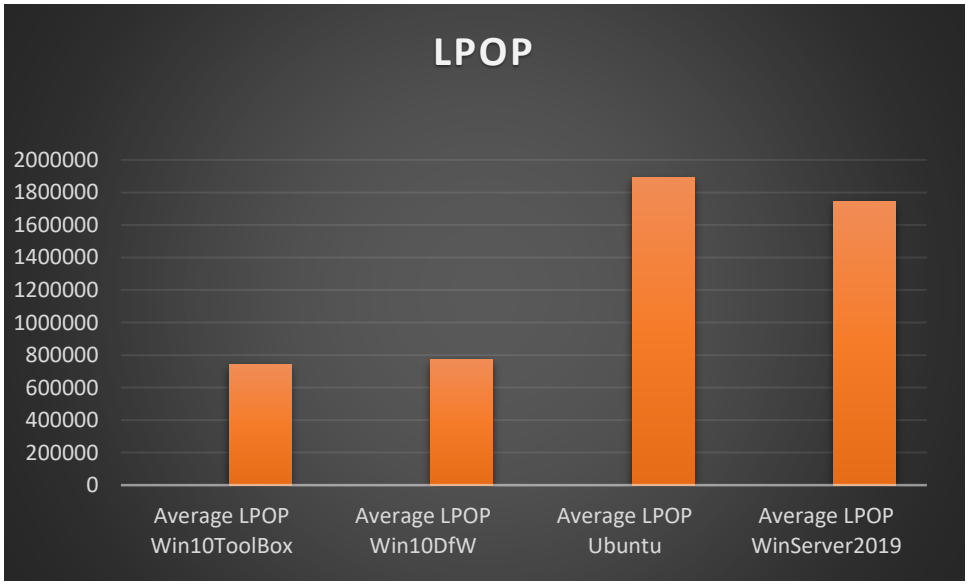
Η πρώτη αναφέρεται στο LPOP το οποίο διαγράφει και επιστρέφει την πρώτη γραμμή της λίστας. Στην Εικόνα 34 βλέπουμε την τυπική απόκλιση στο LPOP και αξίζει να αναφέρουμε ότι ενώ στο πρότυπο του Docker ToolBox υπάρχει ομοιομορφία στις τιμές,

στις άλλες 3 περιπτώσεις κατά την πρώτη εκτέλεση το αποτέλεσμα μας είναι πολύ καλύτερη σε σχέση με τις υπόλοιπες εννιά φορές.



Εικόνα 34. Τυπική Απόκλιση LPOP

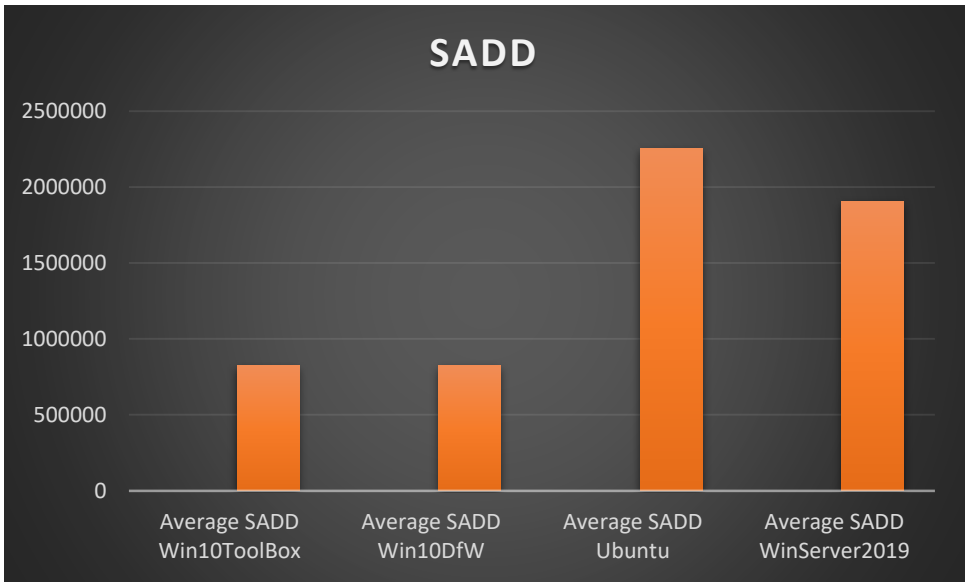
Στην Εικόνα 35 παρατηρούμε ότι έχουμε τα καλύτερα αποτελέσματα και πάλι στα Ubuntu, αλλά με μικρή διαφορά σε σχέση με τα Windows Server. Αντίστοιχα είναι κοντά και οι τιμές στα δύο συστήματα με Windows 10.



Εικόνα 35. Αποτελέσματα LPOP

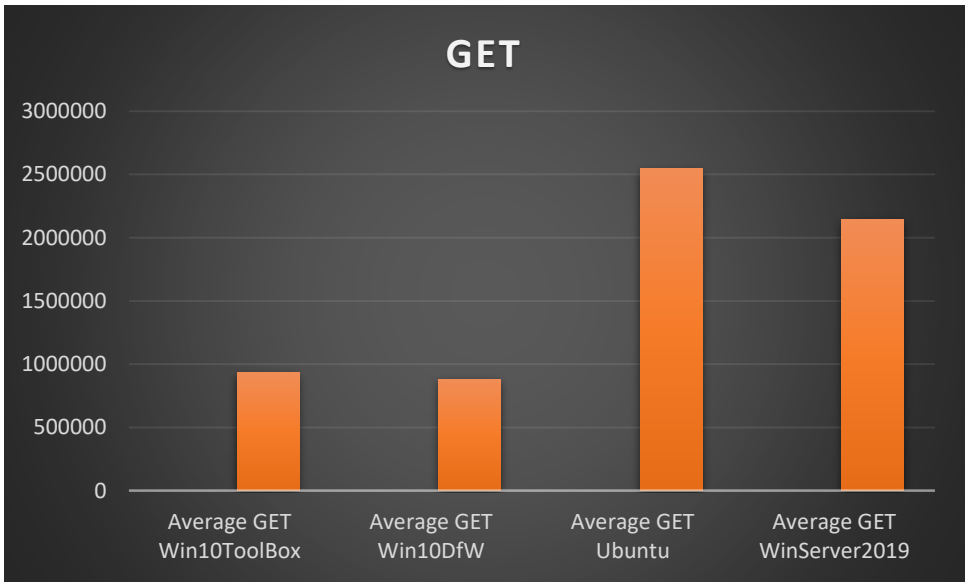
Επόμενη μέτρηση στη συγκριτική δοκιμή μας είναι στην εντολή SADD η οποία προσθέτει δεδομένα στη δομή δεδομένων. Υψηλότερη τιμή έχουμε στα Ubuntu με

μέση τιμή 2250726,4 και την χαμηλότερη στο Docker For Windows 10 η οποία είναι σχεδόν ίδια με το Windows ToolBox.

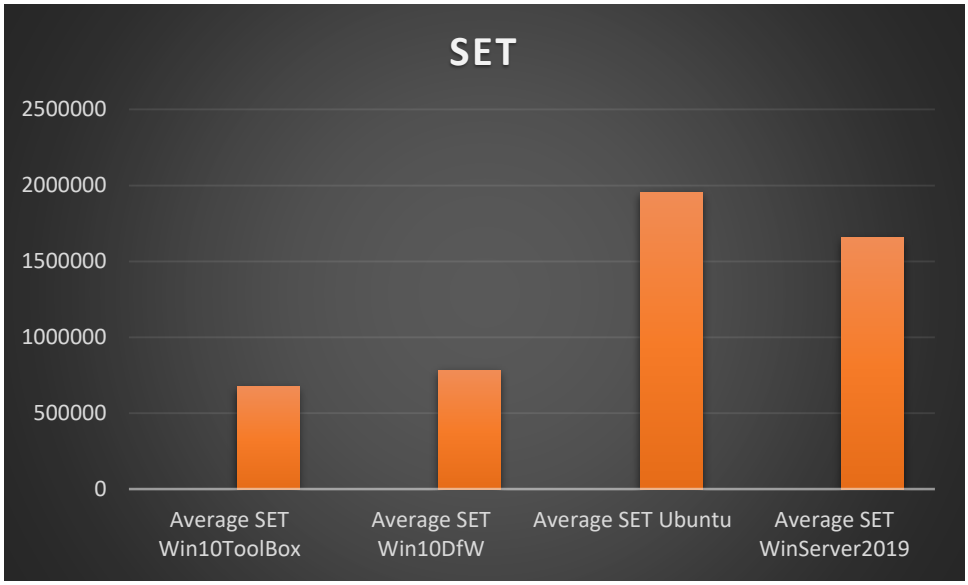


Εικόνα 36. Αποτελέσματα SADD

Αντίστοιχα τα αποτελέσματα είναι ίδια και για τις υπόλοιπες εντολές GET και SET όπως φαίνεται και στις Εικόνες 37 και 38.



Εικόνα 37. Αποτελέσματα GET

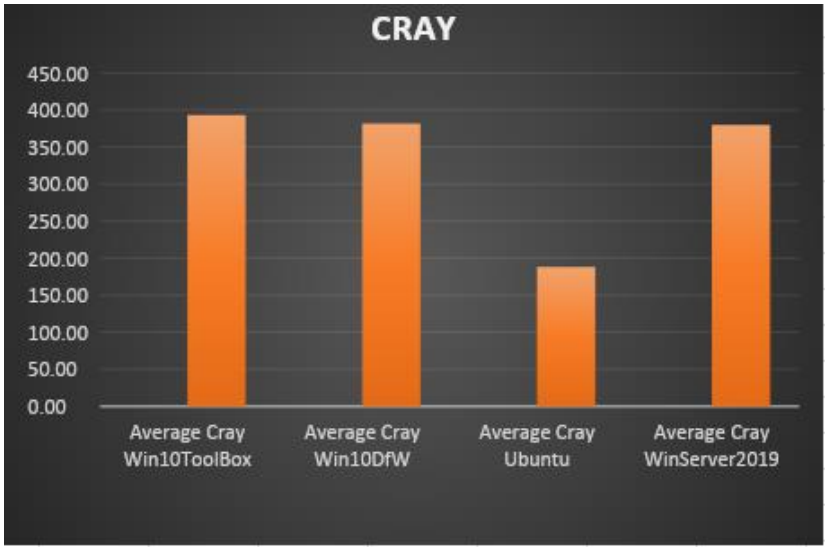


Εικόνα 38. Αποτελέσματα GET

4.4 Δοκιμή 3: C-ray

Το C-ray benchmark αποτελεί ένας απλό ray tracer που σχεδιάστηκε για να κάνει πειραματικές δοκιμές στο floating point του επεξεργαστή. Αποτελεί την πιο απλή μορφή μιας δοκιμής που δεν χρειάζεται καμία αλληλεπίδραση με την κύρια μνήμη ενός υπολογιστή παρά μόνο με την μνήμη επιπέδου 1 του επεξεργαστή. Η συγκεκριμένη δοκιμή που εκτελούμε στο δικό μας περιβάλλον είναι multi-threaded, η οποία θα δημιουργήσει μια εικόνα 1600 X 1200 μετά από εκτέλεση 8 rays ανά pixel για anti-aliasing.

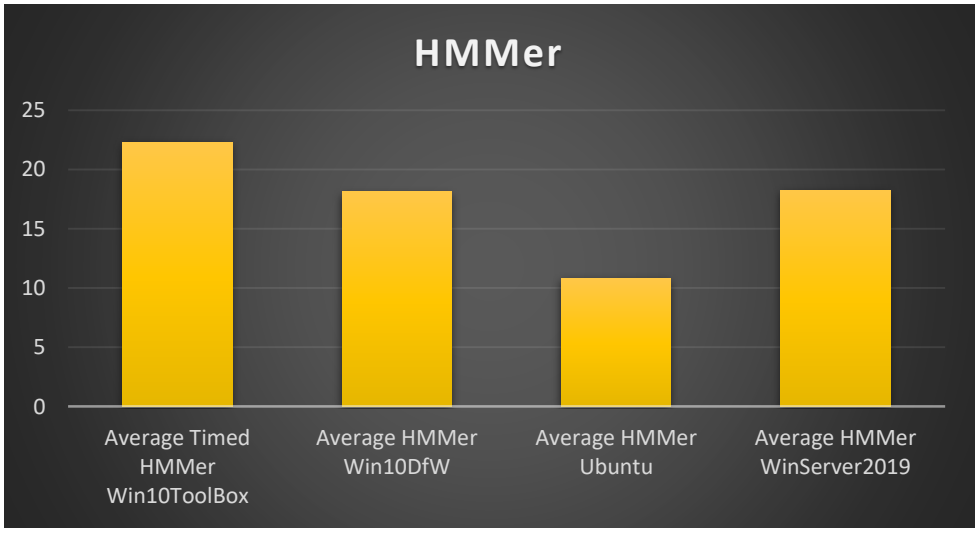
Όπως παρατηρούμε και στην Εικόνα 39 για άλλη μια φορά τα Ubuntu έχουν την καλύτερη τιμή με 188 δευτερόλεπτα ενώ τα άλλα τρία συστήματα βρίσκονται σε πολύ κοντινές τιμές. Είναι άξιο σχολιασμού ότι βλέπουμε για πρώτη φορά το Docker Toolbox να έχει καλύτερη τιμή από τα άλλα δύο συστήματα που τρέχουν σε Hyper-V. Η τυπική απόκλιση και στα τέσσερα πρότυπά μας ήταν στο μηδέν.



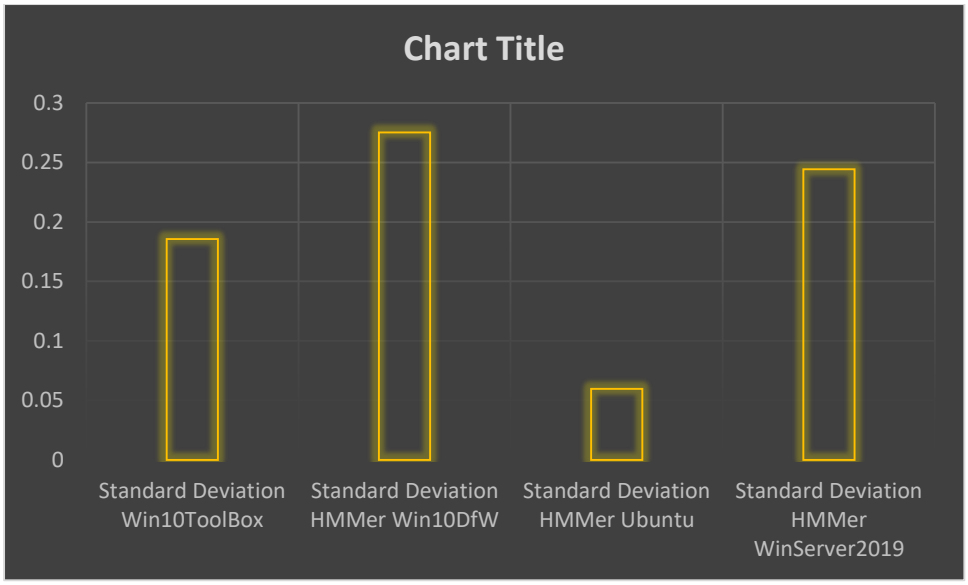
Εικόνα 39. Αποτελέσματα CRAY

4.5 Δοκιμή 4: Timed HMMer Search

Η τέταρτη συγκριτική αξιολόγηση, πραγματοποιεί αναζήτηση στην βάση δεδομένων PFAM που αποτελεί μια τεράστια συλλογή από οικογένειες πρωτεϊνών για κοινά προφίλ Hidden Markov μοντέλα. Η συγκεκριμένη πειραματική διαδικασία αναζητά στη βάση τη δομή της πρωτεΐνης Drosophila Sevenless. Έτσι, όπως βλέπουμε στα αποτελέσματα(Εικόνα 39), η πιο γρήγορη αναζήτηση γίνεται και πάλι με τον Container που εκτελείται σε περιβάλλον Linux, ενώ η τυπική απόκλιση όπως βλέπουμε στην Εικόνα 40, είναι πολύ μικρή για όλα τα συστήματα.



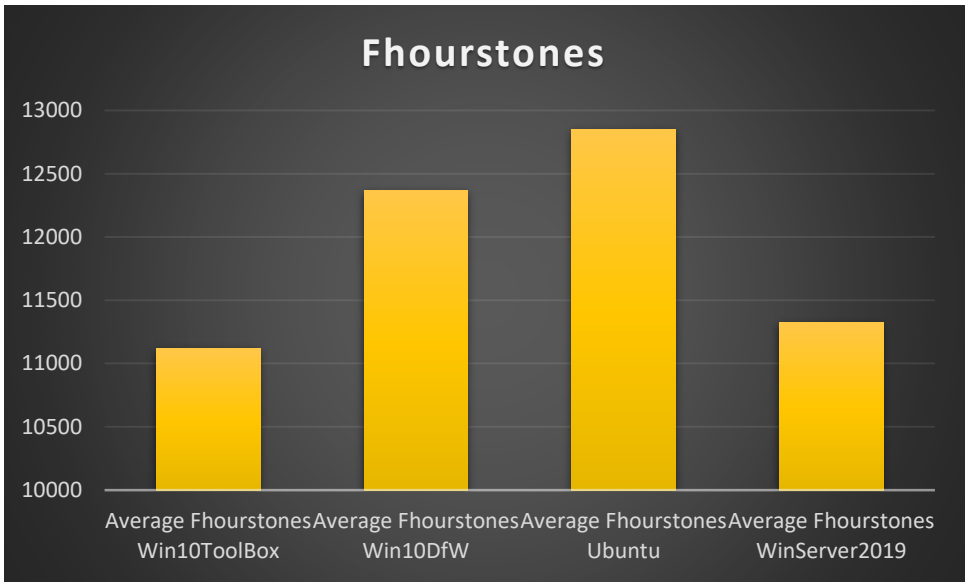
Εικόνα 39. Αποτελέσματα HMMer Search



Εικόνα 40. Τυπική Απόκλιση HMMer Search

4.6 Δοκιμή 5: Fhourstones Benchmark

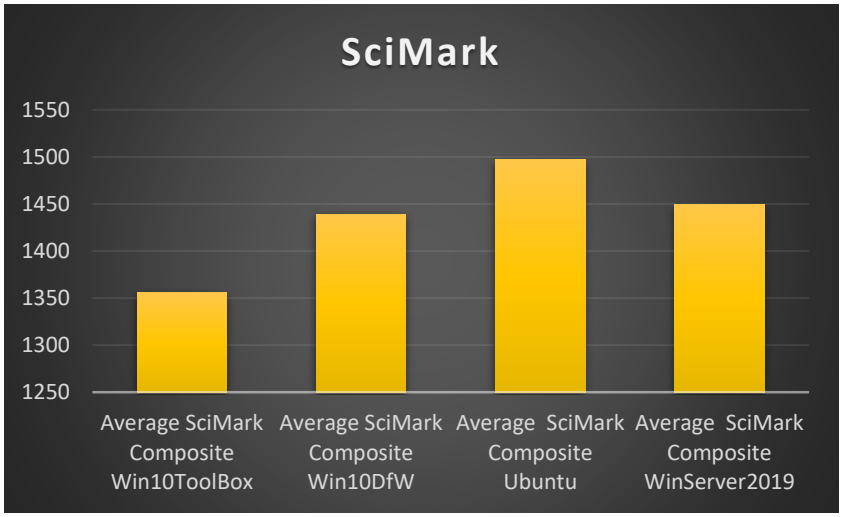
Το Fhourstones βρίσκει την επόμενη θέση στο παιχνίδι Connect-4 σε έναν πίνακα 7X6. Η μονάδα μέτρησης του είναι ο αριθμός των θέσεων που αναζητά ανά δευτερόλεπτο. Στα αποτελέσματά μας, είναι η πρώτη φορά που βλέπουμε στη 2η θέση να είναι το Windows 10 Docker For Windows.



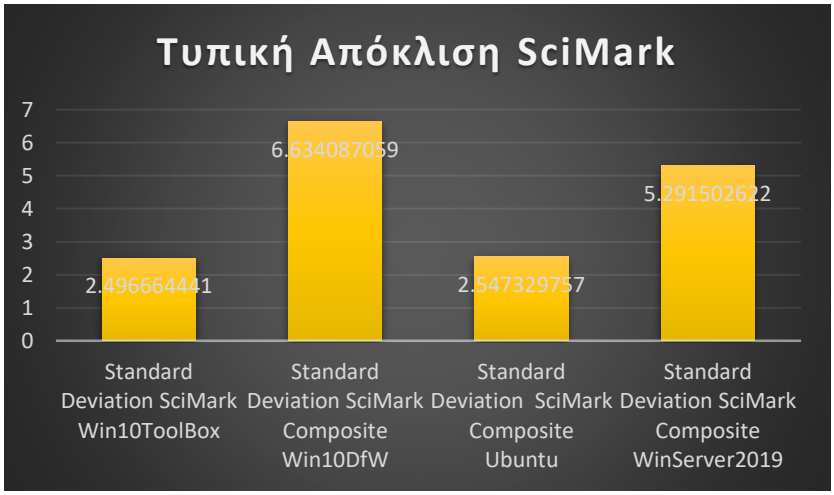
Εικόνα 41. Αποτελέσματα Fhourstones

4.8 Δοκιμή 6: SciMark

Το SciMark αποτελεί μια συγκριτική δοκιμή σε γλώσσα προγραμματισμού Java για επιστημονικούς και αριθμητικούς υπολογισμούς. Εκτελεί διάφορους υπολογισμούς και παρουσιάζει αποτέλεσμα σε Mflops. Όπως βλέπουμε στο διάγραμμα των αποτελεσμάτων (Εικόνα 42) η μεγαλύτερη τιμή εμφανίζεται στα Ubuntu, 1497,4 Mflops ενώ η μικρότερη στο Docker ToolBox με τιμή 1356,3.
Τέλος, η τυπική απόκλιση όπως φαίνεται στην Εικόνα 42, βρίσκεται σε πολύ χαμηλά επίπεδα.



Εικόνα 42. Αποτελέσματα SciMark



Εικόνα 43. Τυπική απόκλιση SciMark

Κεφάλαιο 5 Συμπεράσματα και Μελλοντικές Προεκτάσεις

Μέσα από τη συγκριτική μελέτη μας, καταφέραμε να καταλήξουμε σε κάποια συμπεράσματα. Τα αποτελέσματα μας δείχνουν 3 βασικά πράγματα.

Το πρώτο και σημαντικότερο είναι ότι υπάρχει εξέλιξη στις δυνατότητες για δημιουργία, εκτέλεση αλλά και διαχείριση Linux Containers μέσα από περιβάλλον των Microsoft Windows. Σε όλα τα πειράματά μας, είδαμε την εξέλιξη από το αρχικό Docker ToolBox το οποίο έκανε χρήση μιας άλλης εφαρμογής για να λειτουργήσει η εικονική μηχανή (το VirtualBox), στην εανσωμάτωση της τεχνολογίας του Hyper-V. Η χρήση του Hyper-V έδειξε ότι γίνεται καλύτερη και ουσιαστικότερη διαχείριση στις εικονικές μηχανές. Όπως ήταν αναμενόμενο τα Microsoft Windows Server με την καλύτερη διαχείριση σε εικονικές μηχανές που διαθέτει, αποτύπωσαν και στα αποτελέσματα αυτή την υπεροχή.

Ένα τρίτο συμπέρασμα που μπορούμε να βγάλουμε, είναι ότι ενώ δείξαμε την ιστορική εξέλιξη των Linux Containers σε περιβάλλον Microsoft, στα πειράματά μας φαίνεται ξεκάθαρα ότι όταν μας ενδιαφέρει η λειτουργικότητα και η απόδοση ενός συστήματος, όπως για παράδειγμα στον Apache web Server, η απόδοση στο αρχικό τους περιβάλλον είναι πολύ καλύτερη.

Μια πολύ ενδιαφέρουσα μελλοντική προέκταση που δεν μπόρεσε να μελετηθεί στην συγκεκριμένη διπλωματική εργασία, είναι η συνέχεια των συγκεκριμένων συγκριτικών δοκιμών σε περιβάλλον Microsoft Windows Subsystem for Linux 2 (WSL2). Κατά την συγγραφή της παρούσας διατριβής, έγινε προσπάθεια εκτέλεσης των συγκριτικών δοκιμών σε Microsoft WSL1. Στην πρώτη έκδοση του Windows Subsystem fro Linux, ο πυρήνας Linux που τρέχει απευθείας στο Λειτουργικό των Windows χωρίς την μεσολάβηση μιας εικονικής μηχανής είχε αρκετές ελλείψεις και δεν μπορούσε να εκτελέσει Linux Containers. Τον Μάιο του 2020 η Microsoft ανακοίνωσε τη 2η έκδοση του WSL η οποία υποστηρίζει Linux Containers.

Βιβλιογραφικές Αναφορές

[1] Sagar Ajay Rahalkar(2016), “Virtualization and Cloud Basics”, India

[2] Hyungro Lee, “Virtualization Basics: Understanding Techniques and Fundamentals”

[3] Roberto Morabito, Jimmy Kjällman, and Miika Komu (2015) “Hypervisors vs. Lightweight Virtualization: a Performance Comparison” 2015 IEEE International Conference on Cloud Engineering

[4] Radhwan Y Ameen, Asmaa Y. Hamo(2013) “Survey of Server Virtualization”)International Journal of Computer Science and Information Security, Vol.11, No.3, 2013

[5] Wellie Chao (2006) “The Pros and Cons of Virtual Machines in the Datacenter”

[6] Sung-Jae Jung, Yu-Mi Bae and Wooyoung Soh(2011) “Web Performance Analysis of Open Source Server Virtualization Techniques ” International Journal of Multimedia and Ubiquitous Engineering Vol. 6, No. 4, October, 2011

[7] Apache License v2.0, Docker Engine overview https://docs.docker.com/engine/

[8] Michael Bosse, Kubernetes vs Docker – What is the difference https://www.nakivo.com/blog/docker-vs-kubernetes/(13 Μαίου 2019)

[9] Mark Labrecque, Pros and Cons of Docker, https://affinitybridge.com/blog/pros-and-cons-docker(8 Οκτωβρίου 2018)

[10] Aram koukia, Why Docker? Pros and Cons, https://koukia.ca/why-docker-pros-and-cons-949d104478c5 (23 Νοεμβρίου 2017)

[11]Microsoft, Linux Containers on Window 10 https://docs.microsoft.com/en-us/virtualization/windowscontainers/deploy-containers/linux-containers(17 Σεπτεμβρίου 2019)

[12] Ubuntu, Run linux containers on Windows, https://ubuntu.com/tutorials/windows-ubuntu-hyperv-containers#1-overview (2020)

[13] Rolf Neugebauer, Preview: Linux Containers on Windows https://www.docker.com/blog/preview-linux-containers-on-windows/(13 Σεπτεμβρίου 2017)

[14] Thomas Maurer, Install Hyper-v on windows Server using Powershell https://www.thomasmaurer.ch/2017/08/install-hyper-v-on-windows-server-using-powershell/(1 Αυγούστου 2017)

[15]Microsoft, Install the Hyper-V role on Windows Server https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/get-started/install-the-hyper-v-role-on-windows-server(2 Αυγούστου 2016)

[16] Chandan Kumar, How to perform Redis Benchmark? https://geekflare.com/redis-benchmark-tools/(23 Ιουνίου 2019)

[17] The Apache Software Foundation, Apache HTTP Server benchmarking tool, ()https://httpd.apache.org/docs/2.4/programs/ab.html

[18]Redis Abs- Home of Redis, An introduction to Redis data types and abstractions https://redis.io/topics/data-types-intro

[19] Roldan Pozo, Bruce Miller, About Scimark 2.0 https://math.nist.gov/scimark2/(2004)

[20] Phoronix Media https://www.phoronix-test-suite.com/documentation/phoronix-test-suite.html

[21] Wes Felter, Alexandre Ferreira, Ram Rajamony, Juan Rubio (2014) “An Updated Performance Comparison of Virtual Machines and Linux Containers” RC25482 (AUS1407-001) July 21, 2014 Computer Science