Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Μεταπτυχιακό Πρόγραμμα Σπουδών *Στην Ασφάλεια Υπολογιστών και Δικτύων*

Μεταπτυχιακή Διατριβή

Δυναμικό Τείχος Προστασίας Χαμηλού Κόστους

Κωνσταντίνος Ευσταθίου

Επιβλέπων Καθηγητής
Δρ. Κωνσταντίνος Λιμνιώτης

Νοέμβριος 2023

Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Μεταπτυχιακό Πρόγραμμα Σπουδών *Στην Ασφάλεια Υπολογιστών και Δικτύων*

Μεταπτυχιακή Διατριβή

Δυναμικό Τείχος Προστασίας Χαμηλού Κόστους

Κωνσταντίνος Ευσταθίου

Επιβλέπων Καθηγητής
Δρ. Κωνσταντίνος Λιμνιώτης

Η παρούσα μεταπτυχιακή διατριβή υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων για απόκτηση μεταπτυχιακού τίτλου σπουδών στην Ασφάλεια Υπολογιστών και Δικτύων από τη Σχολή Θετικών και Εφαρμοσμένων Επιστημών του Ανοικτού Πανεπιστημίου Κύπρου.

Νοέμβριος 2023

ΛΕΥΚΗ ΣΕΛΙΔΑ

Περίληψη

Η κυβερνοασφάλεια είναι ένας τομέας όπου η προσαρμοστικότητα και η άμεση ανταπόκριση σε απειλές είναι αποφασιστικής σημασίας. Τα παραδοσιακά τείχη προστασίας (Firewalls) πλέον είναι αδύνατο να ανταπεξέλθουν στις ανάγκες και είναι σαφώς αποδυναμωμένα , γεγονός που οδηγεί στην ανάγκη εξέλιξης νέων λύσεων. Οι υπάρχουσες τεχνολογίες συγκεντρώνονται κυρίως στη ανίχνευση επιθέσεων σε όλα τα επίπεδα χωρίς να προχωρούν ιδιαίτερα στην αντιμετώπιση τους. Στις περιπτώσεις που υπάρχει η δυνατότητα αντίδρασης από κάποιο προϊόν, γίνεται μέσω μηχανισμού που διαθέτει το ίδιο το προϊόν π.χ τα IPs (intrution prevention systems). Επίσης, προϊόντα που έχουν τέτοιες δυνατότητες συνήθως έχουν υψηλό κόστος και είναι συμβατά με προϊόντα από τον ίδιο μόνο κατασκευαστή .

Στόχος αυτής της μεταπτυχιακής διατριβής είναι η αντιμετώπιση των πιο πάνω προκλήσεων μέσω ενός καινοτόμου συστήματος βασισμένο σε λειτουργικό Linux το οποίο να εγκατασταίνεται μπροστά από το τείχος προστασίας του οργανισμού, να έχει δυνατότητες αποκοπής μίας επίθεσης σε πραγματικό χρόνο, να έχει χαμηλό κόστος και να μπορεί να χρησιμοποιηθεί σαν μια γενική (generic) συσκευή συμβατή με οποιοδήποτε σύστημα έχει ικανότητα ανίχνευσης επιθέσεων όπως είναι τα Firewalls, τα Ids (Intrution detection systems), τα SIEMs (log management and analysis tools) κτλ.

Η συσκευή που δημιουργήσαμε βασίστηκε στις δυνατότητες του λειτουργικού συστήματος Centos Linux και του τείχους προστασίας IPtables που προσφέρει σε επίπεδο πυρήνα. Το σύστημα αναπτύχθηκε χρησιμοποιώντας διάφορες τεχνικές και εργαλεία που προϋπάρχουν στο εν λόγο σύστημα και έγινε ανάπτυξη ειδικού κώδικα API (application program Interface) για να πετύχουμε τη συμβατότητα με άλλα περιφερειακά συστήματα. Επίσης δημιουργήθηκε ειδικό service σε επίπεδο λειτουργικού για να πετύχουμε με ασφάλεια την επικοινωνία μεταξύ των συστημάτων. Η επικοινωνία μεταξύ των συστημάτων συμπεριλαμβανομένης και των αποτελεσμάτων σε επίπεδο κανόνων καθώς και η κατάσταση του συστήματος αποτυπώνονται σε πραγματικό χρόνο σε ένα ειδικά διαμορφωμένο Web Interface.

Summary

Cybersecurity is a field where adaptability and immediate response to threats are of crucial importance. Traditional security systems like Firewalls are no longer capable of meeting the demands and are clearly weakened, which leads to the need for the development of new solutions. Existing technologies primarily focus on detecting attacks at all levels without delving significantly into their mitigation. In cases where response capability exists within a product, it is usually accomplished through mechanisms inherent to the product itself, e.g., intrusion prevention systems (IPS). Additionally, products with such capabilities often come at a high cost and are usually compatible only with products from the same manufacturer.

The aim of this thesis is to address the aforementioned challenges through an innovative Linux system. This system is installed in front of the organization's firewalls and has real-time attack mitigation capabilities. It is cost-effective and can be used as a generic device compatible with any system with attack detection capabilities, such as firewalls, intrusion detection systems (IDS), SIEMs (log management and analysis tools), etc.

The device we created is based on the capabilities of the CentOS Linux operating system and the IPtables kernel level firewall. The system was developed using various techniques and tools existing within the system, and a special API (application program interface) code was developed to achieve compatibility with other peripheral systems. Special services were also created at the operating system level to ensure secure communication between systems. Communication between systems, including rule results and system status, is reflected in real-time in a specially configured web interface.

Ευχαριστίες

θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες προς αυτούς που με στήριξαν και με βοήθησαν κατά τη διάρκεια αυτής της σημαντικής πορείας.

Καταρχάς, θέλω να ευχαριστήσω τον επιβλέποντα καθηγητή μου, Δρ. Κωνσταντίνο Λιμνιώτη, για την εκπληκτική υποστήριξη, την εποικοδομητική καθοδήγηση και την εμπιστοσύνη που επέδειξε καθ' όλη τη διάρκεια αυτής της ερευνητικής διαδρομής. Οι συμβουλές σας προσέφεραν πολύτιμες κατευθύνσεις που συνέβαλαν σημαντικά στην επιτυχή ολοκλήρωση αυτής της εργασίας.

Επίσης, δεν θα μπορούσα να παραλείψω να εκφράσω την ευγνωμοσύνη μου προς την οικογένειά μου. Η στήριξη, η κατανόηση και η υπομονή που μου προσέφεραν καθ' όλη την πορεία της μεταπτυχιακής διατριβής μου αλλά και καθ' ολη τη διάρκεια της μεταπτυχιακής σπουδής μου ήταν ανεκτίμητες. Χωρίς την υποστήριξή τους, αυτή η επίτευξη δεν θα ήταν δυνατή.

Σας ευχαριστώ όλους για την υποστήριξη και τη συνεισφορά σας.

Κωνσταντίνος

Περιεχόμενα

Κεφάλαιο 1
Εισαγωγή

1.1 Εισαγωγή

Η κυβερνοασφάλεια αποτελεί ένα από τους πλέον κρίσιμους παράγοντες της σύγχρονης κοινωνίας καθώς η ψηφιακή εποχή που διανέμουμε έχει διαμορφώσει ένα κόσμο όπου η πληροφορία και η τεχνολογία διαδραματίζουν κυρίαρχο ρόλο στη ζωή μας. Οι πολλαπλές διασυνδέσεις στο Internet, τα social media και η έλλειψή ενημέρωσης σχετικά με τους κινδύνους που πηγάζουν από το Internet έχουν δημιουργήσει ένα τρωτό και επικίνδυνο περιβάλλον τόσο σε προσωπικό αλλά και σε εταιρικό επίπεδο .

Οι κυβερνοεπιθέσεις στις μέρες μας έχουν γίνει συχνό φαινόμενο. Οι συνέπειες πέραν από τις οικονομικές επιπτώσεις που προκαλούν επηρεάζουν και τους ανθρώπους σε προσωπικό επίπεδο. Μπορούν να οδηγήσουν σε απώλεια δεδομένων, διαρροές εμπιστευτικών πληροφοριών και ακόμη και σε φυσικές καταστροφές.

Σύμφωνα με στατιστικά από διάφορους οίκους αξιολόγησης το Cyber Risk είναι πλέον μέσα στα πρώτα ρίσκα λειτουργικού κινδύνου για την κάθε επιχείρηση .

1.1 Το τεχνολογικό σκηνικό της σύγχρονης εποχής μας

Οι υπάρχουσες τεχνολογίες στην κυβερνοασφάλεια αποτελούν ζωτικής σημασίας εργαλεία για την αντιμετώπιση των κυβερνοεπιθέσεων και προστασία των ψηφιακών περιβάλλοντων. Οι υπάρχουσες τεχνολογίες χρησιμοποιούνται από τους πλείστους οργανισμούς που είναι εκτεθειμένοι στο Internet . Ανάλογα με τις υπηρεσίες που προσφέρει ο οργανισμός και αναλόγως του προϋπολογισμού που έχει για την κυβερνοασφάλεια δημιουργείται και η ανάλογη υποδομή.

Σημαντικό είναι να επισημάνουμε ότι η καλή ασφάλεια δεν σημαίνει κατ' ανάγκη ψηλό προϋπολογισμό . Μια σωστή αξιολόγηση κινδύνου είναι πάντα η βάση για τη δημιουργία του πλαισίου κυβερνοασφάλειας για το κάθε οργανισμό ξεχωριστά.

Στις μέρες μας οι επιθέσεις σε συστήματα πληροφορικής έχουν γίνει ένα πάρα πολύ συχνό φαινόμενο με τις επιθέσεις στα συστήματα να είναι συνεχώς αυξανόμενες και πιο εξελιγμένες. Για την αντιμετώπιση τους χρησιμοποιούνται συνήθως πολύ εξειδικευμένα συστήματα (firewalls) τα οποία τις πλείστες φορές κοστίζουν από μερικές εκατοντάδες μέχρι και μερικές χιλιάδες ευρώ. Αναλογικά τρέχει και το κόστος συντήρησης τους , όπως το κόστος των αδειών χρήσης , το κόστος του εξειδικευμένου ανθρωπίνου δυναμικού που χρειάζεται για το administration τους, η υποστήριξή (support) από το προμηθευτή και πολλά άλλα.

Οι πιο σημαντικές και ευρέως γνωστές τεχνολογίες τις οποίες θα αναλύσουμε και περαιτέρω είναι τα Firewalls , IDS/IPS , SOCs και SIEMs.

1.2 Ερευνητικά Ερωτήματα

Οι πιο πάνω τεχνολογίες είναι αρκετά ώριμες στον εντοπισμό μίας επίθεσης και σε μερικές περιπτώσεις στην αποτροπή της χρησιμοποιώντας ίδιους πόρους. Επίσης, ως επί το πλείστο βασίζονται σε προ-επιλεγμένους κανόνες τα γνωστά μας Firewall rules χωρίς δυνατότητα κάποιου δυναμικού επαναπρογραμματισμού (dynamic reconfiguration) που να μπορεί να συνδυάζει τις δικές τους δυνατότητες εντοπισμού κάποιας επίθεσης αλλά και ανάλυση από περιφερειακά συστήματα τα οποία δεν είναι κατ' ανάγκη συστήματα ασφαλείας (security systems) όπως π.χ κάποιος email server ή κάποιο custom web application. Η προστασία που προσφέρουν, αν υποστηρίζεται , είναι συνήθως επιπέδου 3 (Layer 3 - Network)

Λαμβάνοντας υπόψη τα πιο πάνω καθώς και την ανάγκη που υπάρχει για πιο αποτελεσματικές, οικονομικές και συνάμα καινοτόμες τεχνολογίες στον τομέα της κυβερνοασφάλειας προσπαθήσαμε να δημιουργήσουμε μία καινοτόμο συσκευή η οποία θα μπορεί να χρησιμοποιηθεί και να προστατεύσει ταυτόχρονα όλα τα πιο πάνω συστήματα ασφαλείας αλλά και γενικά όλα τα συστήματα ενός οργανισμού. Η δική μας προσέγγιση βασίστηκε σε σύστημα Linux και στις δυνατότητες του για εύκολη εφαρμογή κανόνων ασφαλείας σε επίπεδο γέφυρας (network bridge) .

Επομένως, στη βάση των πιο πάνω θα ερευνήσουμε :

1) Τις τεχνολογίες που υπάρχουν και μπορούν να χρησιμοποιηθούν για την ανάπτυξη τείχους προστασίας σε επίπεδο γέφυρας (network bridge)

2) Κατά πόσο μια χαμηλών προδιαγραφών κοινή (generic) συσκευή θα μπορούσε να χρησιμοποιηθεί για αυτό το σκοπό ώστε να μπορεί να είναι οικονομικά προσιτή ακόμα και από μικρομεσαίους οργανισμούς

3) Πώς θα μπορεί μια τέτοια συσκευή να επικοινωνεί και να είναι συμβατή με περιφερειακά συστήματα όπως συστήματα ανάλυσης logs π.χ SIEMs.

1.3 Μεθοδολογία που ακολουθήθηκε

Η μελέτη μας στις υπάρχουσες τεχνολογίες μας βοήθησε να εντοπίσουμε ένα πιθανό κενό στο τρόπο αντιμετώπισής κυβερνοεπιθέσεων από τους διάφορους οργανισμούς. Είχαμε συγκεντρωθεί κυρίως στις δυνατότητες αξιοποίησης των δεδομένων ασφαλείας που παραλαμβάνονταν και επεξεργάζονταν από τα διάφορα συστήματα ασφαλείας. Είχαμε παρατηρήσει ότι τα πλείστα συστήματα επεξεργάζονταν πάρα πολύ καλά τα δεδομένα και έφταναν σε πολύ ορθά συμπεράσματα σχετικά με τους επιτιθέμενους αλλά πολύ λίγα ήταν εκείνα που προχωρούσαν στη αποκοπή μιας επίθεσης χρησιμοποιώντας κάποιο μηχανισμό ασφαλείας π.χ αναπρογραμματισμό κάποιου Firewall σε πραγματικό χρόνο.

Λαμβάνοντας υπόψη τα πιο πάνω προχωρήσαμε στο σχεδιασμό ενός συστήματος το οποίο θα μπορεί να χρησιμοποιηθεί σε μεγάλο εύρος, να είναι οικονομικό, να μπορεί εύκολα να επικοινωνεί με άλλα περιφερειακά συστήματα και να έχει δυνατότητες αποκοπείς επιθέσεων. Λάβαμε επίσης υπόψη ότι το σύστημα θα έπρεπε να είναι εύκολο στην εγκατάσταση του μέσα σε ένα οργανισμό .

Για να ικανοποιήσουμε τις πιο πάνω απαιτήσεις προχωρήσαμε με σχεδιασμό συστήματος βασισμένο στο λειτουργικό Linux.

Επιλέξαμε να χρησιμοποιήσουμε λειτουργικό Centos Linux το οποίο εγκαταστήσαμε σε χαμηλών προδιαγραφών υπολογιστή με τέσσερις κάρτες δικτύου . Το Linux μας παρείχε όλα τα απαραίτητα εργαλεία για τη δημιουργία της συσκευής μας και επίσης μας παρείχε και τα εργαλεία για ανάπτυξη οτιδήποτε θα χρειαζόταν. Για την επικοινωνία με περιφερειακά συστήματα σχεδιάσαμε και αναπτύξαμε ένα γενικό API απλής χρήσης με σκοπό να πετύχουμε συμβατότητα με όσο γίνεται περισσότερα

συστήματα. Επίσης, σχεδιάσαμε και υλοποιήσαμε ένα βασικό WebInterface για να παρακολουθούμε την απόδοση του συστήματος σε πραγματικό χρόνο.

Ακολούθως, εγκαταστήσαμε το security σύστημα σε οικιακό περιβάλλον και προχωρήσαμε με ελέγχους απόδοσης του συστήματος χρησιμοποιώντας διάφορα scripts.

1.4 Δομή διατριβής

Στη παρούσα μεταπτυχιακή διατριβή θα παρουσιάσουμε μερικές από τις υπάρχουσες τεχνολογίες στο τομέα της κυβερνοασφάλειας. Θα γίνει βιβλιογραφική ανασκόπηση σε σημαντικά σημεία από τη διαθέσιμη βιβλιογραφία και θα αναλυθούν οι υπάρχουσες τεχνολογίες.

Ακολούθως θα παρουσιάσουμε τη δική μας καινοτόμο πρόταση η οποία θεωρούμε ότι θα αποτελέσει ένα εναλλακτικό και πολύ αποτελεσματικό τρόπο αντιμετώπισης κυβερνοεπιθέσεων. Η δική μας πρόταση θα προσπαθήσει να παρουσιάσει ένα χαμηλών προδιαγραφών σύστημα ικανό να «φορτωθεί» με χιλιάδες κανόνες τύπου Firewall (firewall rules) από διάφορα περιφερειακά, μέσω API, για να προστατέψει κάποιο οργανισμό.

Στη συνέχεια θα εστιάσουμε σε τεχνική υλοποίηση της λύσης, η οποία θα περιλαμβάνει:

- λεπτομερή περιγραφή όλων των βημάτων που ακολουθήθηκαν για το setup του λειτουργικού,
- την παραμετροποίηση που έγινε στο λειτουργικό
- την δημιουργία του API και των αναγκαίων γεφυρών επικοινωνίας

Επίσης, θα παρουσιάσουμε σενάρια εμπνευσμένα από πραγματικές συνθήκες και θα δείξουμε στατιστικά απόδοσης του συστήματος κάτω από αυτές τις συνθήκες.

Κεφάλαιο 2

Ανασκόπηση Βιβλιογραφίας

2.1 Ανασκόπηση Βιβλιογραφίας

Η υλοποίηση του συστήματος αναμένεται να έχει αρκετό integration με πληθώρα διαφορετικών τεχνολογιών και εργαλείων από διάφορες τεχνολογίες ανοιχτού κώδικα (open source). Τέτοιες τεχνολογίες χρησιμοποιήθηκαν και μελετήθηκαν από πολλούς ερευνητές τα τελευταία χρόνια με απώτερο σκοπό τη βελτίωση τους και συνάμα την αύξηση στην ασφάλεια που προσφέρουν .

Οι σχετικές έρευνες που εντοπίσαμε μέσω Google Scholar και της βιβλιοθήκης του Ανοιχτού Πανεπιστημίου Κύπρου αναλύουν και συνάμα αναδεικνύουν τις δυνατότητες των εργαλείων αυτών σε αρκετά λεπτομερή βαθμό. π.χ η έρευνα του 2016 από τους Diekmann, Michaelis, Haslbeck, και Carle σχετικά με την 'Verified iptables firewall analysis' [1] προσφέρει λεπτομερή ανάλυση της αποτελεσματικότητας του iptables σε περιβάλλοντα που διατίθενται σε πραγματικό χρόνο. Αυτή η μελέτη επιβεβαιώνει όχι μόνο την ικανότητα του iptables να αποτρέπει επιθέσεις αλλά και να ανταποκρίνεται σε διάφορες μορφές επιθέσεων με επιτυχία.

Η ενσωμάτωση της έρευνας αυτής με την ανάλυση των Oktivasari, Zain, [2] και συνεργατών για την αντιμετώπιση επιθέσεων Slowloris αποδεικνύει τη σημαντική συνοχή στα αποτελέσματα που προκύπτουν από τη χρήση του iptables στην προστασία ενός διακομιστή ιστού (Webserver). Η ικανότητα του iptables να εντοπίζει και να μπλοκάρει επιθέσεις όπως η επίθεση Slowloris παρουσιάζεται ως αξιόπιστη, ενισχύοντας έτσι την αξιοπιστία του ως εργαλείο ασφαλείας.

Επίσης, οι έρευνες με τίτλο 'Log Management comprehensive architecture in Security Operation Center (SOC)' [3] και ‘The design and implement of the centralized log gathering and analysis system’ [4] προσφέρουν μια σημαντική προσέγγιση στη χρήση μοντέλων ανάλυσης αρχείων καταγραφής για την υποστήριξη της διαδικασίας έρευνας σε περιβάλλοντα διακεκριμένης κυβερνοασφάλειας. Η ανάλυση αυτή αποσκοπεί στη δημιουργία μοντέλων συσχέτισης δεδομένων για την ανίχνευση, αξιολόγηση και ανάλυση πληροφοριών από διάφορες πηγές καταγραφής. Οι έρευνες προτείνουν μια

συστηματική μέθοδο για την ανάπτυξη μοντέλων που είναι σε θέση να συσχετίζουν δεδομένα από διάφορα συστήματα και πηγές καταγραφής, προκειμένου να παρέχουν ένα ολοκληρωμένο εργαλείο για τη διερεύνηση και ανάλυση περιστατικών μέσω συσχέτισης (correlation). Ακόμα πιο λεπτομερή είναι η έρευνα 'Security Log Management: Identifying Patterns in the Chaos' [5] η οποία ασχολείται με τεχνικές ανάλυσης χρησιμοποιώντας διάφορα λογισμικά ανοιχτού κώδικα.

Όπως διαφαίνεται, οι πιο πάνω έρευνες συγκεντρώνονται ως επί το πλείστον στις δυνατότητες των εργαλείων σε ατομικό επίπεδο ή στον εντοπισμό μίας επίθεσης σε θεωρητικό επίπεδο.

Ο συνδυασμός των πιο πάνω μαζί με περαιτέρω αυτοματοποίηση μέσω το σημείο της αποκοπής μιας επίθεσης θα αποτελέσει και το βασικό στόχο της Μεταπτυχιακής μας Διατριβής.

2.2 Υπάρχουσες Τεχνολογίες

Η κυβερνοασφάλεια είναι ένας τομέας που αποτελείται από πολλές και πολύπλοκες τεχνολογίες για τη προστασία των συστημάτων, των δικτύων και των δεδομένων ενός οργανισμού. Η ανάγκη για αποτελεσματική προστασία οδήγησε στη δημιουργία πολλών διαφορετικών και πολλές φορές ομοίων τεχνολογιών οι οποίες εφαρμόζονται με βάση τις ανάγκες του κάθε οργανισμού.

Firewalls (Τείχη Προστασίας)

Τα firewalls είναι πρώτη γραμμή προστασίας για τα συστήματα ενός οργανισμού. Κυρίως σκοπός τους είναι να ελέγχουν την κίνηση από και προς ένα δίκτυο, βασιζόμενοι σε προκαθορισμένους κανόνες. Στην κύρια λειτουργία τούς εξασφαλίζουν ότι μόνο εξουσιοδοτημένη, επιθυμητή και ασφαλής κίνηση περνά ανάμεσα στα δίκτυα, ενισχύοντας έτσι την ασφάλεια του συστήματος και των δεδομένων.

IDS/IPS (Intrution Detection/Prevention Systems)

Τα IDS είναι συστήματα για εντοπισμό ανεπιθύμητων επιθέσεων ή επικίνδυνων δραστηριοτήτων στο δίκτυο. Βασίζονται κυρίως σε "signatures" , δηλαδή patterns τα οποία προσπαθούν να ταυτίσουν στην κίνηση του δικτίου ώστε να αναγνωρίσουν μια κυβερνοεπίθεση. Ανάλογα με τη ρύθμιση τους (IPs) και τη τοποθεσία τους σε ένα δίκτυο μπορεί να δράσουν και να αποκόψουν μια επίθεση σε πραγματικό χρόνο.

SOC (Security Operation Center)

Τα SOC είναι εξειδικευμένα κέντρα παρακολούθησης. Στα SOC συνήθως εργάζονται εξειδικευμένοι επαγγελματίες στον τομέα της κυβερνοασφάλειας, οι οποίοι επιβλέπουν συστηματικά τις δραστηριότητες του δικτύου. Με τη χρήση προηγμένων εργαλείων και τεχνολογιών, ανιχνεύουν πιθανές απειλές, αναλύουν συμπεράσματα και αντιδρούν άμεσα σε πιθανά προβλήματα ασφαλείας.

SIEM (Security Information and Event Management)

Τα SIEMs είναι συστήματα που συλλέγουν, αναλύουν πληροφορίες ασφαλείας από διάφορες πηγές, προκειμένου να ανιχνεύσουν και να αντιμετωπίσουν κάποια απειλή ασφαλείας. Οι πληροφορίες που αναλύονται από τα SIEM μπορούν να χρησιμοποιηθούν για την ανίχνευση προβλημάτων ασφαλείας, την έγκαιρη αποκάλυψη επιθέσεων, τη διαχείριση περιστατικών και τη λήψη μέτρων για την αντιμετώπισή τους.

Κεφάλαιο 3
Περιγραφή Μεθοδολογίας

3.1 Γενικά

Πολλοί οργανισμοί τα τελευταία χρόνια έχουν υιοθετήσει λύσεις SIEM με σκοπό να έχουν καλύτερη οπτική του περιβάλλοντος τους όσο αφορά την κυβερνοασφάλεια. Στις πλείστες περιπτώσεις η δυνατότητα των συστημάτων αυτών σταματά στο σημείο της ανάλυσης και της αναγνώρισης μιας επίθεσης και δεν υπάρχει ο μηχανισμός αποκοπείς της. Η έρευνα αυτή θα προσπαθήσει να δημιουργήσει μια χαμηλού κόστους αλλά συνάμα σταθερή και αποτελεσματική συσκευή, εύκολη στη λειτουργία και εγκατάσταση της, που θα μπορεί να αξιοποιεί τα αποτελέσματα από όλα τα SIEMς.

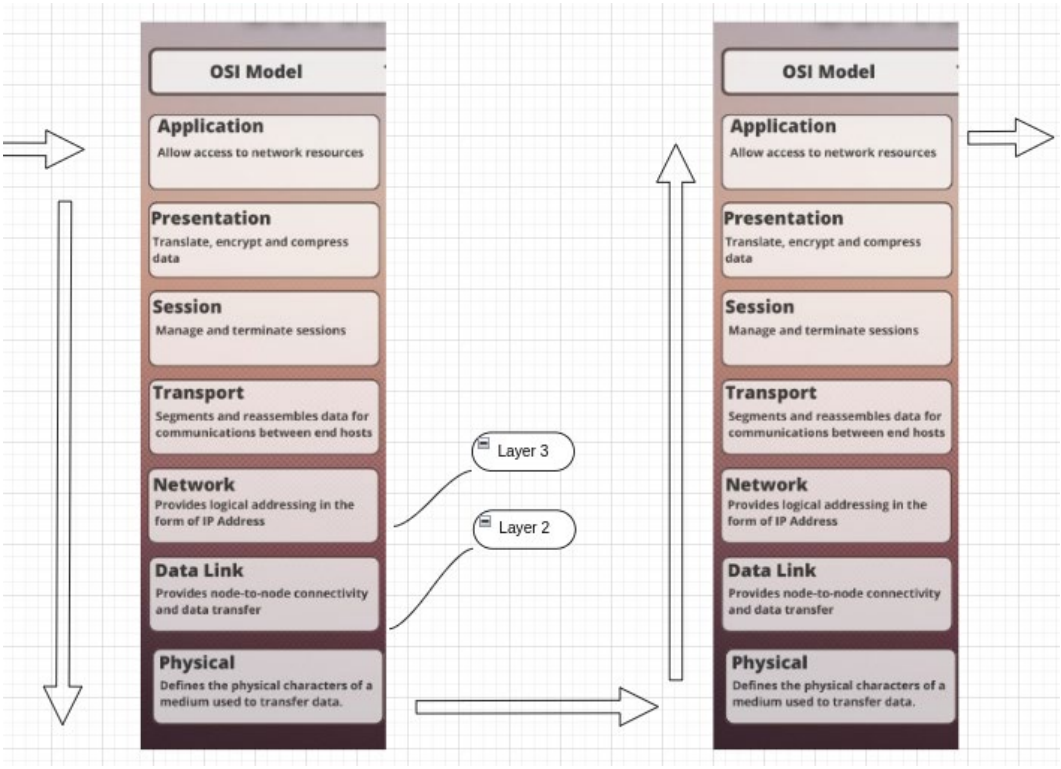
Η συσκευή αυτή θα εκμεταλλευτεί και θα συνδυάσει τις δυνατότητες που προσφέρει μια γέφυρα δικτύου (network bridge) μαζί με το τείχος προστασίας (iptables) του λειτουργικού Linux. Η σύνδεση με τα διάφορα περιφερειακά θα γίνεται μέσω API.

3.1.1 Το μοντέλο OSI

Το OSI (Open Systems Interconnection) είναι ένα θεωρητικό πλαίσιο που καθορίζει λειτουργίες δικτύων σε επτά επίπεδα, ώστε να επιτύχουμε επικοινωνία μεταξύ διαφορετικών υπολογιστικών συστημάτων. Η ροή της επικοινωνίας αποτυπώνεται στην Εικόνα 1 – ISO Model . Η επικοινωνία ξεκινά συνήθως από κάποιο input από κάποιο χρήστη ή εφαρμογή στο “Επίπεδο Εφαρμογής (Application) “. Ακολούθως, μεταφέρεται στα υπόλοιπα επίπεδα προς τα κάτω μέχρι στο “Φυσικό Επίπεδο (Physical)” όπου μέσω πλέον των συσκευών δικτύου καταλήγει στην απομακρυσμένη (remote) εφαρμογή ακολουθώντας την αντίθετη πορεία.

Τα επίπεδα, περιληπτικά έχουν τα εξής χαρακτηριστικά :

- Φυσικό Επίπεδο (Physical): Μεταφορά δεδομένων μέσω φυσικών μέσων όπως καλώδια, ασύρματες συχνότητες.
- Επίπεδο Σύνδεσης Δεδομένων (Data Link): Διαχείριση σφαλμάτων μετάδοσης, αναγνώριση συσκευών στο δίκτυο με MAC διευθύνσεις.
- Επίπεδο Δικτύου (Network): Δρομολόγηση πακέτων δεδομένων, μετατροπή διευθύνσεων IP.
- Επίπεδο Μεταφοράς (Transport): Εγκαθίδρυση σύνδεσης, διαχείριση ροής δεδομένων μέσω TCP/UDP.
- Επίπεδο Σύστασης (Session): Διαχείριση σφαλμάτων, ροής δεδομένων, συμπερίληψη σήμανσης στα πακέτα δεδομένων.
- Επίπεδο Παρουσίας (Presentation): Μετατροπή μορφής δεδομένων, κωδικοποίηση, συμπερίληψη σημάτων συστήματος.
- Επίπεδο Εφαρμογής (Application): Τοπικές εφαρμογές, πρόσβαση στις διαδικτυακές υπηρεσίες, όπως περιήγηση στον ιστό, ηλεκτρονικό ταχυδρομείο κ.λπ.



Εικόνα 1 . OSI Model

3.1.2 Διάγραμμα εγκατάστασης συσκευής

Μία σημαντική δυνατότητα που μας παρέχει το Linux είναι η εύκολη δημιουργία

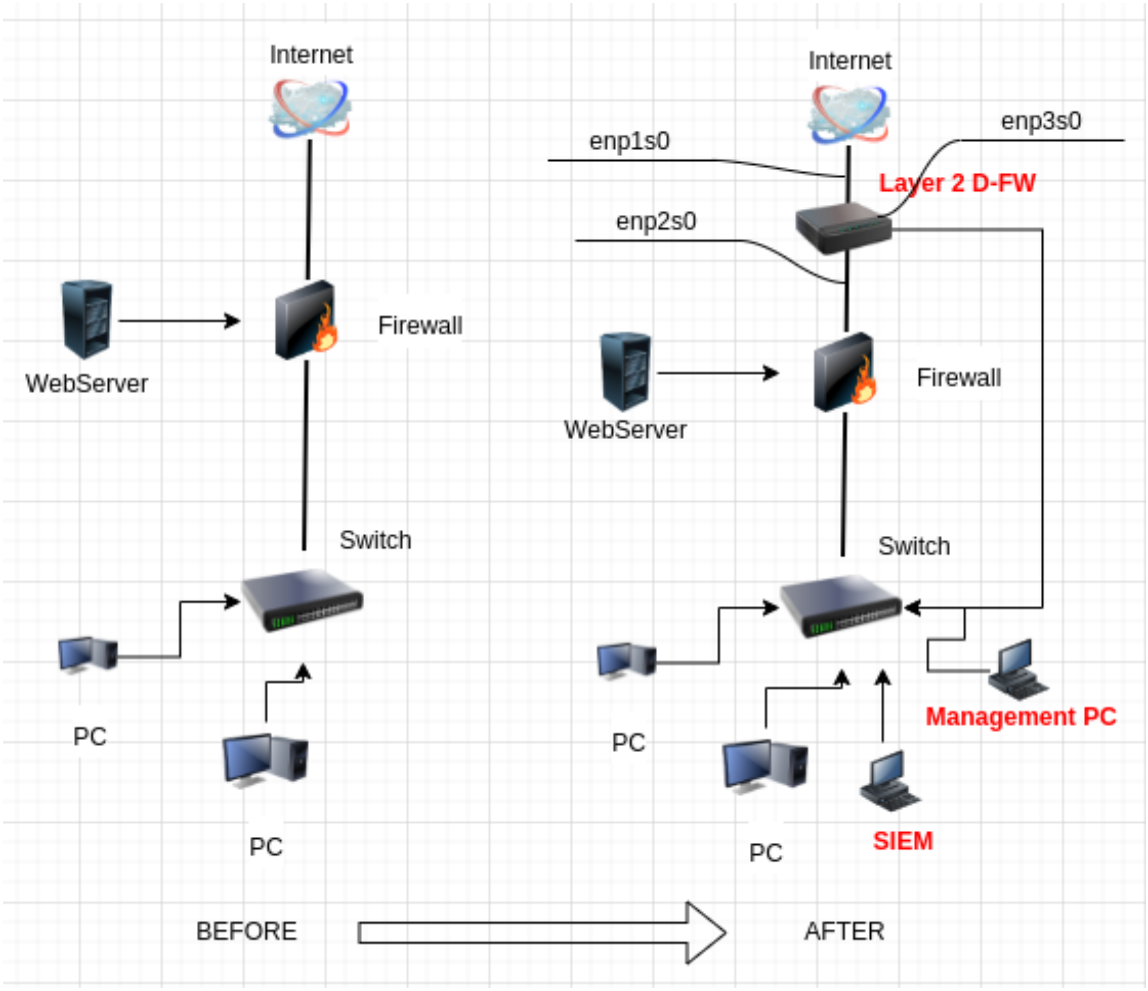
γεφυρών (bridges) μεταξύ των καρτών δικτύου συνδυάζοντας δύο ή περισσότερες από αυτές. Στην περίπτωση των γεφυρών μπορούμε να περάσουμε κίνηση (traffic) μέσω από δύο κάρτες εκ των οποίων η μία θα είναι εισερχόμενη (inbound interface) και η άλλη η εξερχόμενη (outbound interface) και λόγο του γεγονότος ότι δεν επηρεάζεται τη δρομολόγηση τότε πρακτικά μπορείς να δημιουργήσεις μία συσκευή η οποία θα μπορεί να διακόπτει ένα φυσικό καλώδιο στα δύο χωρίς να επηρεάζεται καμία δικτυακή λειτουργία. Στην δική μας συσκευή χρησιμοποιήσαμε δύο κάρτες (enp1s0, enp2s0) για να φτιάξουμε τη γέφυρα μας.

Για την ασφάλεια των γεφυρών χρησιμοποιούνται διάφοροι τρόποι προστασίας από επιθέσεις όπως ασφαλίζοντας τη θύρα (port) μέσω ειδικής διαρρύθμισης στην οποία δηλώνεται χειροκίνητα το σύστημα που έχει πρόσβαση ή χρησιμοποιώντας κανόνες τύπου Firewall οι οποίοι εφαρμόζονται στο επίπεδο της κάρτας δικτύου π.χ. Source,Destination,Action όπου τα Source & Destination δεν είναι διευθύνσεις IP, αλλά φυσικές κάρτες π.χ –inbound-interface enp1s0 .

Λαμβάνοντας υπόψη το σχεδιασμό μιας γέφυρας αλλά και τις δυνατότητες που παρέχει από πλευράς Firewall , προχωρήσαμε και στο σχεδιασμό της δικής μας συσκευής η οποία :

- θα μπορεί να εγκατασταθεί σε οποιονδήποτε δίκτυο χωρίς σχεδόν καθόλου δικτυακές αλλαγές
- δεν θα μπορεί να υποστεί σχεδόν καμία επίθεση σε επίπεδο δικτύου (Επιθέσεις σε IP διευθύνσεις)
- Θα μπορεί να ελέγχει κίνηση σε επίπεδο κάρτας δικτύου , δυνατότητα την οποία παρέχει το Linux Firewall (iptables) που χρησιμοποιήσαμε.

Επίσης για τον έλεγχο της συσκευής χρειάστηκε να εγκαταστήσουμε και τρίτη κάρτα δικτύου (enp2s0) η οποία χρησιμοποιείτε για σκοπούς management της συσκευής. Η κάρτα αυτή θα είναι συνδεδεμένη στο εσωτερικό switch του οργανισμού . Η Εικόνα 2 - Διάγραμμα εγκατάστασης συσκευής – αποτυπώνει διαγραμματικά τον τρόπο εγκατάστασης της συσκευής .



Εικόνα 2 – Διάγραμμα εγκατάστασης συσκευής

Με βάση το πιο πάνω για την ανάπτυξη της προτεινόμενης συσκευής χρησιμοποιήσαμε ανοιχτού κώδικα λογισμικό (Centos Linux ver 7) το οποίο εγκαταστήσαμε σε χαμηλών προδιαγραφών υπολογιστή. Για τη δημιουργία του Firewall χρησιμοποιήσαμε IPTABLES, για το API χρησιμοποιήσαμε PHP με Apache Web Server. Επίσης, δημιουργήσαμε και ένα απλό Web Interface με το οποίο θα παρακολουθούμε τα calls στο API μας και τα Logs που θα δημιουργούνται στο Firewall. Το Web Interface θα έχει backend database το MariaDB.

Σημαντικό είναι το γεγονός ότι η συσκευή θα μπορεί να εγκαθίσταται μπροστά από το Firewall του οργανισμού και θα προστατεύει ακόμα και το ίδιο το Firewall από πιθανές επιθέσεις. Ενδεικτικά αναφέρεται σαν παράδειγμα μία συνεχόμενη επίθεση στο ίδιο το Firewall ή σε κάποιο σύστημα που προστατεύεται από το Firewall και βρίσκεται σε κάποιο ζώνη DMZ. Αφού γίνει αντιληπτή η επίθεση τότε θα μπορούμε να την σταματήσουμε οριστικά και να αποφορτίσουμε τόσο το Firewall όσο και το σύστημα

που δέχεται την επίθεση κάνοντας reset (----reject) το connection στη δική μας συσκευή.

3.2 Ανάπτυξη συσκευής

Για την ανάπτυξη της συσκευής χρησιμοποιήσαμε software & hardware τα οποία είναι κοινά και διαθέσιμα είτε δωρεάν (open source) είτε πολύ φθηνά π.χ γενικών προδιαγραφών (generic) hardware.

3.2.1 Λογισμικά που χρησιμοποιήθηκαν

Όλα τα λογισμικά που χρησιμοποιήσαμε είναι ανοιχτού κώδικα και διέπονται από άδεια ανοιχτής χρήσης (Open Source License). Το κύριο λογισμικό είναι το Centos Linux , που είναι και το λειτουργικό σύστημα που χρησιμοποιήθηκε. Τα λοιπά λογισμικά συμπεριλαμβάνονται στο Centos Linux.

Centos Linux : Το CentOS Linux είναι μια διανομή λειτουργικού συστήματος βασισμένη στον κώδικα πηγής του Red Hat Enterprise Linux (RHEL). Έχει σχεδιαστεί για να παρέχει μια σταθερή, αξιόπιστη και δωρεάν λύση για επιχειρήσεις και χρήστες που αναζητούν ένα λειτουργικό σύστημα βασισμένο στο RHEL χωρίς το κόστος υποστήριξης που συνήθως συνοδεύει την εμπορική έκδοση του Red Hat.

Τα κύρια χαρακτηριστικά του CentOS Linux περιλαμβάνουν:

I. Σταθερότητα: Το CentOS προσφέρει μια σταθερή πλατφόρμα με μακροχρόνια υποστήριξη, καθιστώντας το κατάλληλο για συστήματα παραγωγής.
II. Ασφάλεια: Οι ενημερώσεις ασφαλείας παρέχονται τακτικά
III. Κοινότητα: Μια πολύ ενεργή κοινότητα χρηστών και προγραμματιστών συνεισφέρει στην ανάπτυξη και τη συντήρηση του.

Το λογισμικό μπορεί να γίνει download από το www.centos.org σαν ISO image

Iptables : Είναι ένα εργαλείο που χρησιμοποιείται σε συστήματα Linux για τον έλεγχο και τον διαχειρισμό των κανόνων firewall σε επίπεδο πακέτων δεδομένων. Οι κανόνες iptables χρησιμοποιούνται για την προστασία και τον έλεγχο της δικτύωσης σε ένα Linux σύστημα, επιτρέποντας ή αποκλείοντας πακέτα δεδομένων με βάση διάφορα κριτήρια όπως η προέλευση (source), ο προορισμός (destination), ο τύπος (type) του πακέτου και πολλά άλλα.

Οι κανόνες iptables μπορούν να χρησιμοποιηθούν για διάφορους σκοπούς, συμπεριλαμβανομένων των εξής:

i. Προστασία του συστήματος από επιθέσεις: Οι κανόνες iptables μπορούν να χρησιμοποιηθούν για να αποκλείσουν εισερχόμενες συνδέσεις από ανεπιθύμητες πηγές ή για να περιορίσουν την πρόσβαση σε συγκεκριμένες υπηρεσίες.

ii. Δρομολόγηση πακέτων (routing): Οι κανόνες iptables μπορούν να χρησιμοποιηθούν για τον έλεγχο και δρομολόγηση πακέτων σε ένα δίκτυο.

iii. Μετατροπή IP (NAT): Μπορούν να χρησιμοποιηθούν για τη μετατροπή της διεύθυνσης IP των πακέτων κατά τη διέλευσή τους μέσα από το σύστημα.

Σημαντικό για τη δική μας συσκευή είναι το γεγονός ότι η δρομολόγηση πακέτων (routing) και η μετατροπή IP (NATing) δεν είναι εφικτό να χρησιμοποιηθούν (δεν χρειάζονται) επειδή η συσκευή μας βασίζεται σε γέφυρα δικτύου χωρίς IP διεύθυνση . Οι κανόνες που χρησιμοποιήθηκαν βασίζονται στη δυνατότητα του συγκεκριμένου λογισμικού να αναλύσει κίνηση τόσο σε επίπεδο IP (SRC → DST) αλλά επίσης με βάση την κάρτα εισδοχής (inbound interface) κάποιου πακέτου.

Iptables (IPSET) : Το IPSET module [6] στο iptables είναι ένα plugin που μας επιτρέπει να διαχειριζόμαστε αποτελεσματικά μεγάλα σύνολα διευθύνσεων IP, δικτύων ή θυρών. Είναι ιδιαίτερα χρήσιμο για τη βελτίωση της απόδοσης του iptables όταν ασχολείται με πολλούς κανόνες και διευθύνσεις IP.

Ορισμένα από τα βασικά χαρακτηριστικά του περιλαμβάνουν :

- Διαχείριση Συνόλων: Το IPSET παρέχει ένα πλαίσιο για τη δημιουργία και τη διαχείριση συνόλων διευθύνσεων IP, δικτύων και θυρών. Αυτά τα σύνολα μπορούν να ενημερώνονται δυναμικά, κάτι που διευκολύνει τη διατήρηση blacklists σε κανόνες.
- Βελτιωμένη Απόδοση: Το IPSET είναι σχεδιασμένο για τη διαχείριση μεγάλων συνόλων δεδομένων πιο αποτελεσματικά από τους παραδοσιακούς κανόνες

iptables. Χρησιμοποιεί δομές δεδομένων όπως πίνακες κατακερματισμού (hash tables) και χάρτες bitmap για να επιτύχει γρηγορότερες αναζητήσεις και ταίριασμα κανόνων.

- Ενσωμάτωση με το iptables: Το IPSET ενσωματώνεται απροβλημάτιστα με το τείχος προστασίας iptables, επιτρέποντάς μας να αναφέρουμε σύνολα IPSET στους κανόνες του τείχους προστασίας. Αυτό μπορεί να απλοποιήσει τη διαχείριση των κανόνων και να μειώσει την πολυπλοκότητα της διαμόρφωσης του iptables.

Apache: Ο "Apache" είναι ένας από τους πιο δημοφιλείς open-source web servers παγκοσμίως στη παρουσίαση και διαχείριση ιστοσελίδων στο πλαίσιο του World Wide Web (www).

Ορισμένα βασικά χαρακτηριστικά του Apache HTTP Server περιλαμβάνουν:

- Δυνατότητα Εξυπηρέτησης Ιστοσελίδων: Ο Apache είναι σε θέση να εξυπηρετεί ιστοσελίδες σε ποικίλες γλώσσες προγραμματισμού, όπως HTML, PHP, Python, και πολλές άλλες.

- Δυνατότητα Προσαρμογής: Ο Apache επιτρέπει την προσαρμογή μέσω της ρύθμισης διάφορων modules και διαμορφωτικών αρχείων.

- Ασφάλεια: Οι δυνατότητες ασφαλείας του Apache συμπεριλαμβάνουν τη δυνατότητα διαχείρισης των δικαιωμάτων πρόσβασης (μέσω .htaccess) σε αρχεία και καταλόγους και την υποστήριξη για SSL/TLS για ασφαλή μετάδοση δεδομένων.

- Διαχείριση Εφαρμογών: Ο Apache μπορεί να χρησιμοποιηθεί για τη διαχείριση εφαρμογών web, όπως τα διαδικτυακά καταστήματα, τα blogs, και τα συστήματα διαχείρισης περιεχομένου (CMS).

Για τη συσκευή μας χρησιμοποιήσαμε API (Application Program Interface) γραμμένα σε PHP τα οποία έτρεξαν κάτω από τον Apache Web Server.

PHP : Είναι μια δημοφιλής γλώσσα προγραμματισμού ανοιχτού κώδικα που συχνά

χρησιμοποιείται για την ανάπτυξη δυναμικών ιστοσελίδων και εφαρμογών web. Τα αρχικά PHP αναφέρονται στην φράση "PHP: Hypertext Preprocessor," υποδεικνύοντας την κύρια χρήση της γλώσσας για την επεξεργασία δεδομένων που προορίζονται να εμφανιστούν σε μια ιστοσελίδα HTML πριν από την αποστολή τους στον περιηγητή του χρήστη.

Ορισμένα χαρακτηριστικά και χρήσεις της PHP περιλαμβάνουν:

- Δυναμικές Ιστοσελίδες: Η PHP επιτρέπει τη δημιουργία ιστοσελίδων που παράγουν δυναμικό περιεχόμενο, όπου οι πληροφορίες μπορούν να ανακτώνται από βάσεις δεδομένων, αρχεία, ή άλλες πηγές και να παρουσιάζονται στον χρήστη σε πραγματικό χρόνο.

- Επεξεργασία Δεδομένων: Η PHP διαθέτει πλούσιες λειτουργίες για την επεξεργασία δεδομένων, συμπεριλαμβανομένης της ανάλυσης XML, τον χειρισμό αρχείων, και την αλληλεπίδραση με βάσεις δεδομένων.

- Επικοινωνία με τη Βάση Δεδομένων: Η PHP μπορεί να συνδεθεί με διάφορες βάσεις δεδομένων, όπως η MariaDB (MySQL), PostgreSQL, και SQLite, για την αποθήκευση και ανάκτηση πληροφοριών.

- Ανάπτυξη Εφαρμογών: Εκτός από τον ρόλο της στον προγραμματισμό ιστοσελίδων, η PHP μπορεί να χρησιμοποιηθεί για την ανάπτυξη γενικών εφαρμογών, από γραμμικές εντολές μέχρι γραφικές εφαρμογές.

Η PHP είναι εύκολα προσβάσιμη και διαδεδομένη, με μια μεγάλη κοινότητα προγραμματιστών που παρέχει υποστήριξη, βιβλιοθήκες, και πληροφορίες για την ανάπτυξη με τη χρήση αυτής της γλώσσας.

MariaDB Database : MariaDB είναι ένα συστήματα διαχείρισης βάσεων δεδομένων (DBMS) που βασίζεται στον κώδικα πηγής του MySQL. Πρόκειται για μια ανοιχτού κώδικα εναλλακτική λύση στο MySQL, δημιουργημένη όταν η MySQL AB εξαγοράστηκε από την Oracle Corporation και υπήρξαν ανησυχίες για το μέλλον της ανοιχτής κοινότητας του MySQL.

Τα βασικά χαρακτηριστικά της MariaDB περιλαμβάνουν:

- Συμβατότητα με MySQL: Η MariaDB είναι σχεδιασμένη για να είναι συμβατή με τις εφαρμογές και τις εντολές που αναπτύχθηκαν για το MySQL, καθιστώντας την εύκολη για τους χρήστες που ήδη χρησιμοποιούν MySQL να μεταβούν σε MariaDB χωρίς προβλήματα.

- Βελτιώσεις και Επεκτάσεις: Η MariaDB περιλαμβάνει πολλές βελτιώσεις και επεκτάσεις σε σχέση με τον MySQL, συμπεριλαμβανομένων νέων μηχανισμών αποθήκευσης και εργαλείων.

- Απόδοση: Η MariaDB είναι γνωστή για την αυξημένη απόδοσή της, η οποία μπορεί να είναι χρήσιμη για εφαρμογές που απαιτούν υψηλή απόδοση στην επεξεργασία των δεδομένων.

- Ασφάλεια: Η MariaDB προσφέρει πολλές επιπρόσθετες λειτουργίες ασφαλείας και εργαλεία για τον έλεγχο της πρόσβασης στη βάση δεδομένων και την προστασία των δεδομένων.

3.2.2 Hardware που χρησιμοποιήθηκε

Ένας από τους αρχικούς μας στόχους ήταν η συσκευή που θα αναπτύσσαμε να βασιζόταν πάνω σε χαμηλών προδιαγραφών hardware. Ως εκ τούτου χρησιμοποιήθηκε ένα fanless mini PC (Εικ. 3 – Fanless PC) με 4-κάρτες δικτύου. Το PC ήταν ένα Sophos Appliance XG105 μοντέλο 2018 το οποίο κάναμε format και εγκαταστήσαμε το Centos Linux. Θεωρητικά, μπορεί να χρησιμοποιηθεί οτιδήποτε PC ή Server με τις πιο κάτω ελάχιστες προδιαγραφές :

Processor Type	Intel Celeron
Graphics Card Chipset	Embed
RAM (memory)	2GB
HDD	64GB SSD
Processor Model	Intel(R) Atom(TM) Processor E3930 @ 1.30GHz

RAM Type	DDR4
Network Cards	4* 1GB RJ45

Εικόνα 3 - Fanless PC (Sophos FW Appliance)

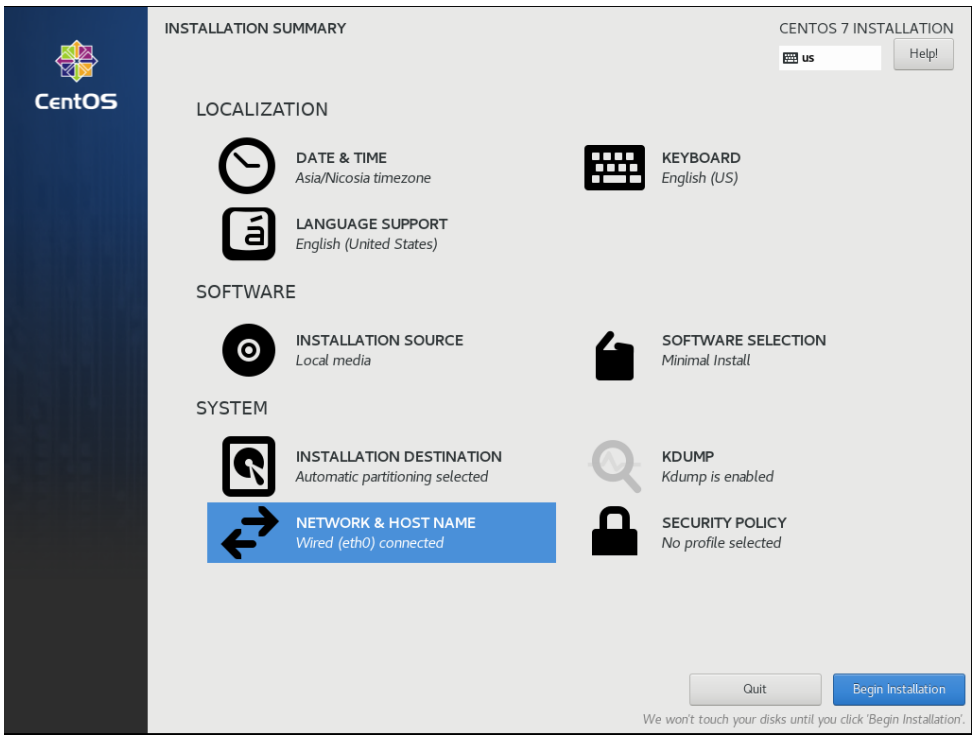
3.3 Setup του συστήματος

3.3.1 Εγκατάσταση Λειτουργικου

Το λειτουργικό σύστημα που επιλέχτηκε για τη συσκευή μας ήταν το Centos Linux ver. 7.9. Το Centos προσφέρεται δωρεάν σαν ISO image στο https://www.centos.org . Το ISO image , αφού το κατεβάσαμε, τρέξαμε την εντολή ‘dd’ για να δημιουργήσουμε ένα bootble USB που χρησιμοποιήσαμε για την εγκατάσταση του.

dd if=Centos_image.iso of=/dev/usb

Ακολούθως, επανεκκινήσαμε το Appliance και επιλέξαμε σαν bootable media το USB που δημιουργήσαμε.



Εικόνα 4 – Εγκατάσταση Centos

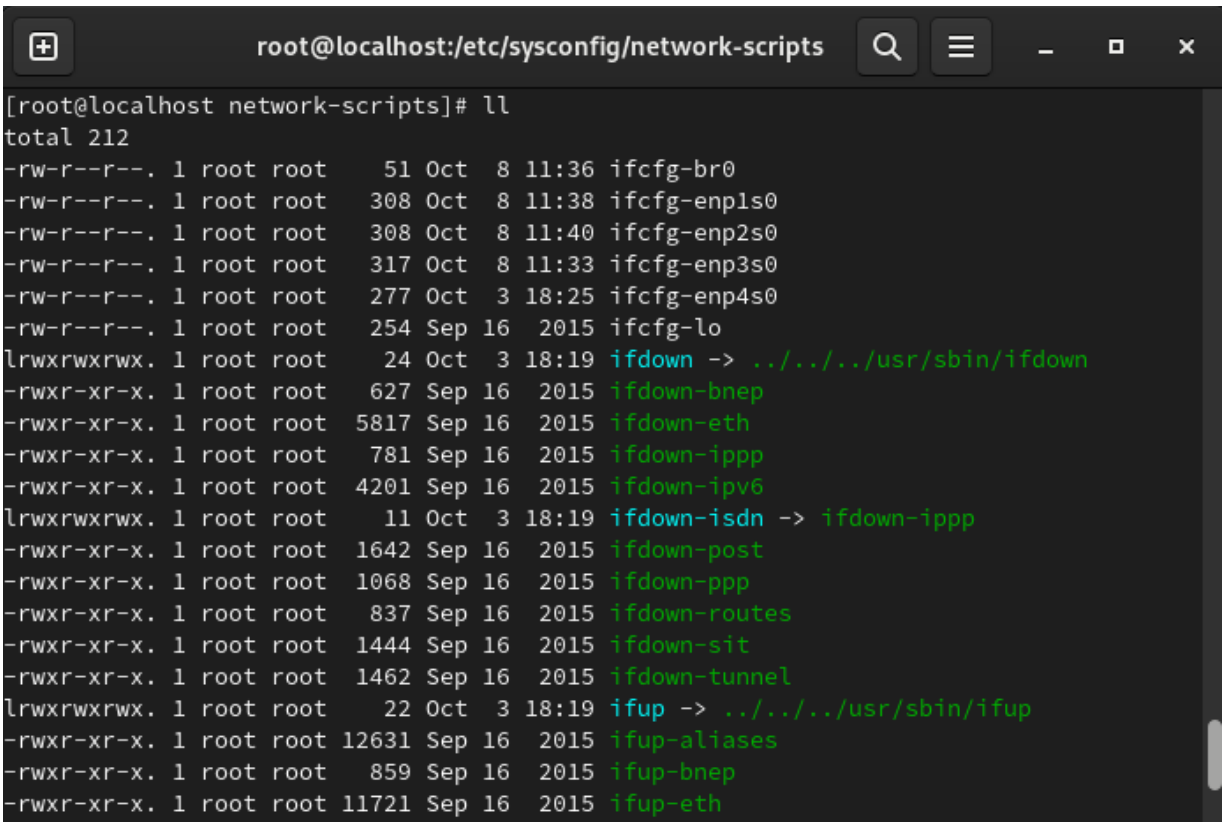
Στην διαδικασία εγκατάστασης δεν κάναμε καμιά αλλαγή, αποδεχτήκαμε τα default settings (Εικ. 4 – Εγκατάσταση Centos) . Με το πέρας της εγκατάστασης, προχωρήσαμε από τη κονσόλα (console) του συστήματος για όλες τις υπόλοιπες παραμέτρους όπως περιγράφονται πιο κάτω.

3.3.2 Διαμόρφωση δικτύου (network configuration)

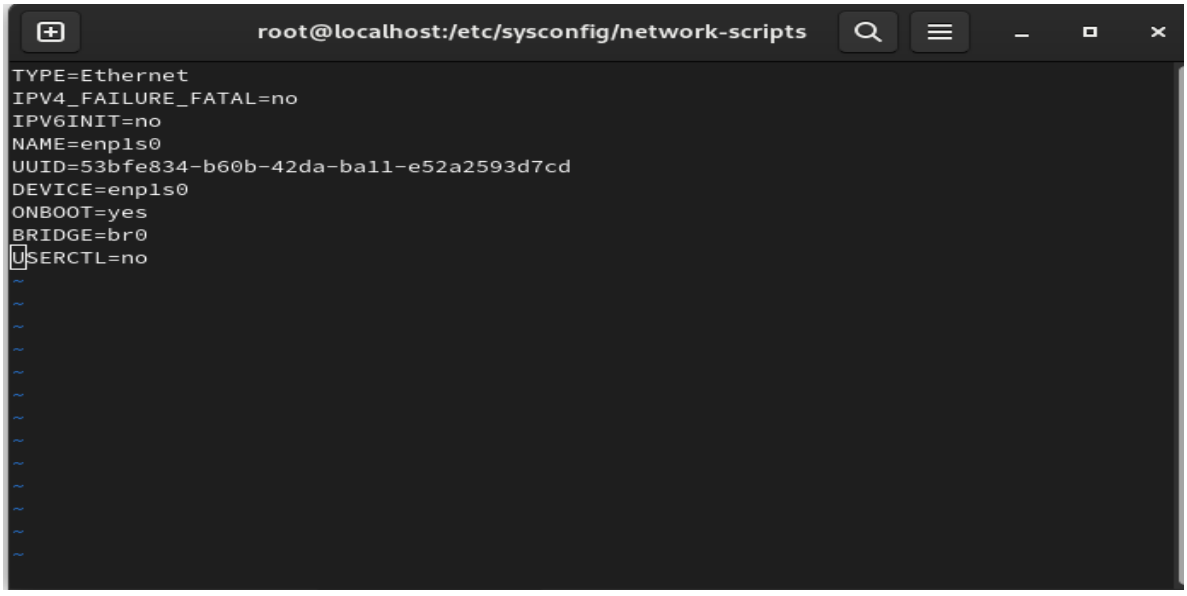
> Δημιουργία γέφυρας (network bridge)

Για τη δημιουργία της γέφυρας χρειάστηκε χειροκίνητα να αλλάξουμε τα βασικά αρχεία που ορίζουν το δίκτυο στο Centos Linux. Η γέφυρα (Εικ. 8 – Bridge Config) που δημιουργήσαμε αποτελείται από δύο κάρτες δικτύου, την enp1s0 (Εικ. 6 – Κάρτα δικτύου enp1s0) και την enp2s0 (Εικ. 7 – Κάρτα enp2s0). Η γέφυρα είναι μία επιπλέον κάρτα δικτύου (br0) η οποία θα αποτελέσει και την κύρια αναφορά μας στην εφαρμογή των κανόνων firewall.

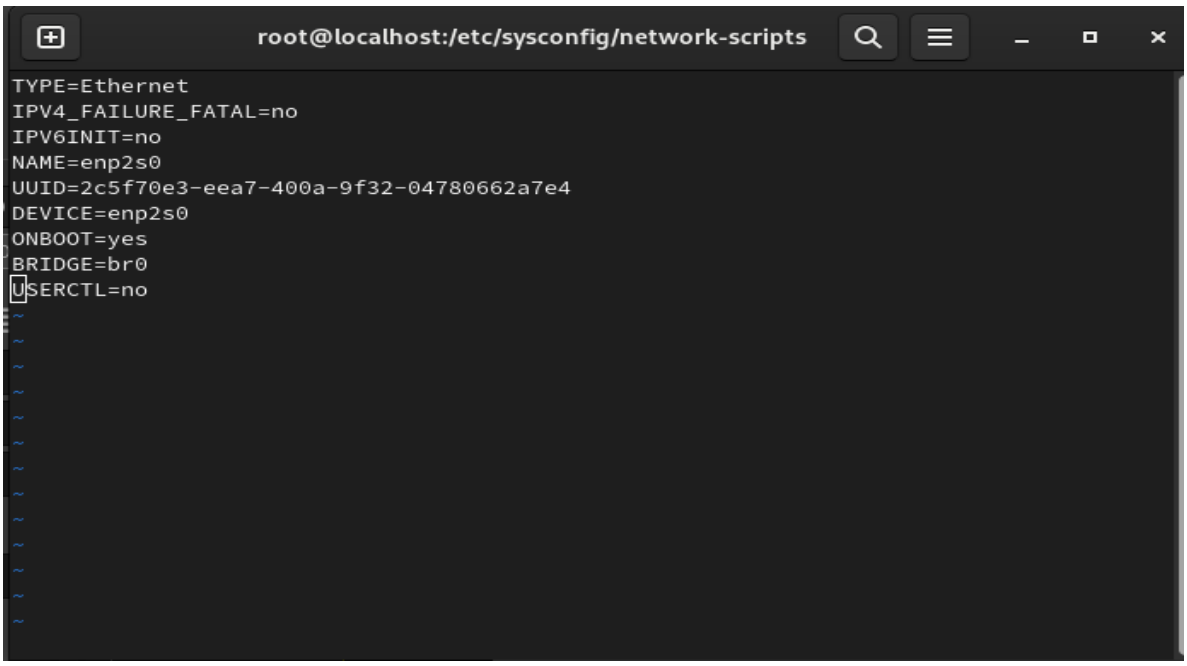
Τα αρχεία που ορίζουν το δίκτυο στο σύστημα Centos (Εικ. 5 – Αρχεία δικτύου σε σύστημα Centos Linux), βρίσκονται στο πιο κάτω path : **/etc/sysconfig/network-scripts**



Εικόνα 5 – Αρχεία δικτύου στο Centos Linux

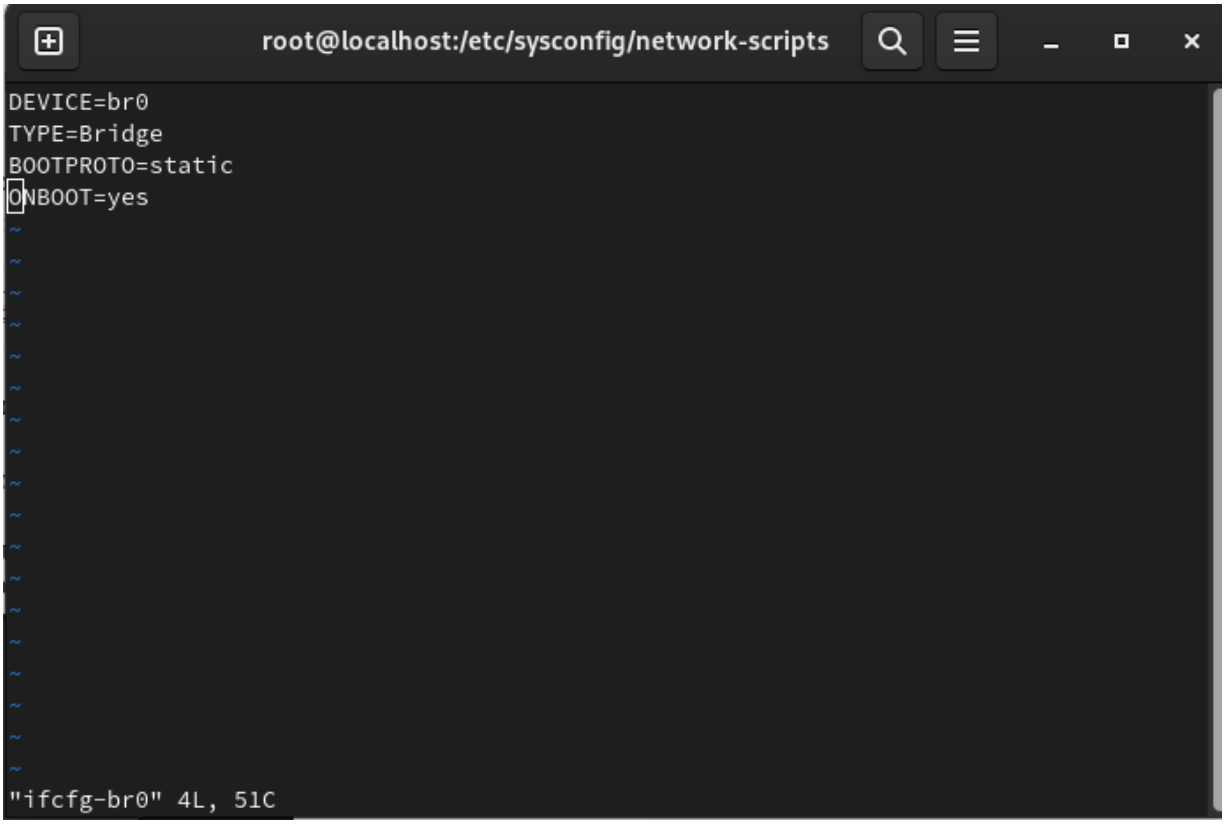


Εικόνα 6 - Κάρτα δικτύου enp1s0



Εικόνα 7 – Κάρτα δικτύου enp2s0

Σημαντική είναι η παράμετρος **BRIDGE=br0** με την οποία ορίζεται η γέφυρα στην οποία ανήκει η συγκεκριμένη κάρτα δικτύου.



Εικόνα 8 – Bridge Config

Με τις πιο πάνω αλλαγές έχουμε πετύχει τη δημιουργία της γέφυρα (bridge) η οποία

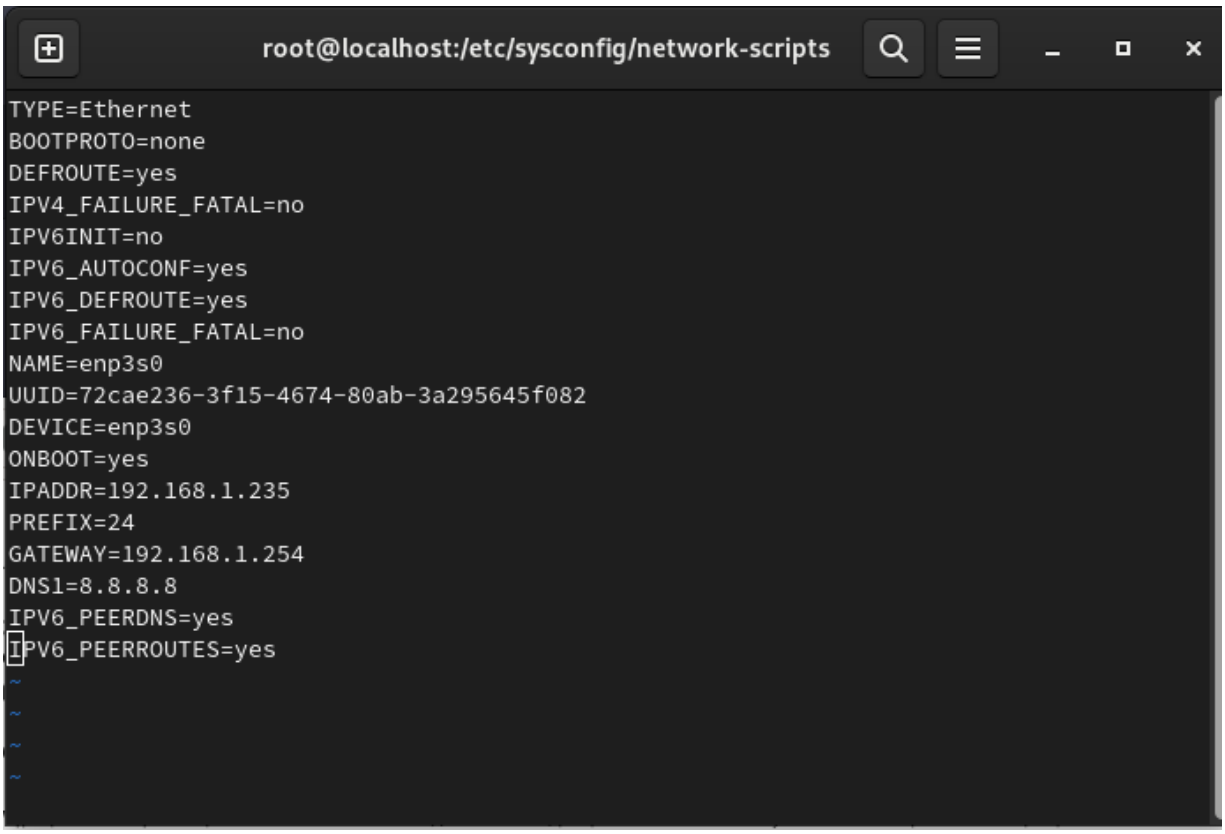
αποτελείται από δύο (2) άλλες φυσικές κάρτες δικτύου. Στο παρόν στάδιο μέσω της γέφυρας (Εικ. 8 – Bridge Config), νοουμένου ότι ενώσουμε τα καλώδια όπως αποτυπώνεται στο αρχικό μας σχεδιάγραμμα (Εικ. 2 – Διάγραμμα εγκατάστασης συσκευής) η κίνηση δικτύου θα μπορεί να περνά χωρίς κανένα εμπόδιο.

> Ρύθμιση κάρτας δικτύου Management

Η συσκευή μας χρειάζεται μία έχτρα κάρτα δικτύου η οποία θα χρησιμοποιείται για σκοπούς administration και επικοινωνίας με τα περιφερικά συστήματα (SIEMs) . Η κάρτα αυτή είναι συνδεδεμένη με το εσωτερικό δίκτυο του εκάστοτε οργανισμού και φέρει IP διεύθυνση του εσωτερικού δικτύου.

Σημαντικό είναι το γεγονός ότι η συγκεκριμένη κάρτα δεν είναι προσβάσιμη από το Internet παρόλο που βρίσκεται στην συσκευή μας η οποία είναι ουσιαστικά ενωμένη με το Internet. Αυτή την απομόνωση μας την εξασφαλίζει η διάταξη της γέφυρα (bridge configuration). Επίσης, και από πλευράς δικτύου δεν είναι δυνατή η επικοινωνία από το Intenet λόγο του ότι η διεύθυνση IP θα είναι εσωτερική (Private) .

Η Εικ. 9 – Management Interface Configuration – αποτυπώνει τις παραμέτρους που έχει η συγκεκριμένη κάρτα (ifcfg-enp3s0).



Εικόνα 9 – Management Interface Configuration

Ενεργοποίηση δικτύου

Η ενεργοποίηση του δικτύου ώστε οι πιο πάνω αλλαγές να περάσουν στο σύστημα σαν μόνιμες γίνεται με την με τις πιο κάτω εντολές :

systemctl restart network
systemctl enable network

3.3.3 Εγκατάσταση λοιπών λογισμικών με τη χρήση του YUM

Η εγκατάσταση λογισμικού στο Centos Linux γίνεται με την εντολή ‘yum’ . Το λογισμικό ‘yum’ είναι και ο προβλεπόμενος τρόπος εγκατάστασης και αναβάθμισης ενός συστήματος Centos Linux. Οι κύριες λειτουργίες της εντολής "yum" περιλαμβάνουν την εγκατάσταση, την αναβάθμιση, την αφαίρεση και τον έλεγχο των πακέτων.

Για όλες τις εγκαταστάσεις πακέτων στο λειτουργικό χρησιμοποιήθηκε η εντολή:

#yum install package-name

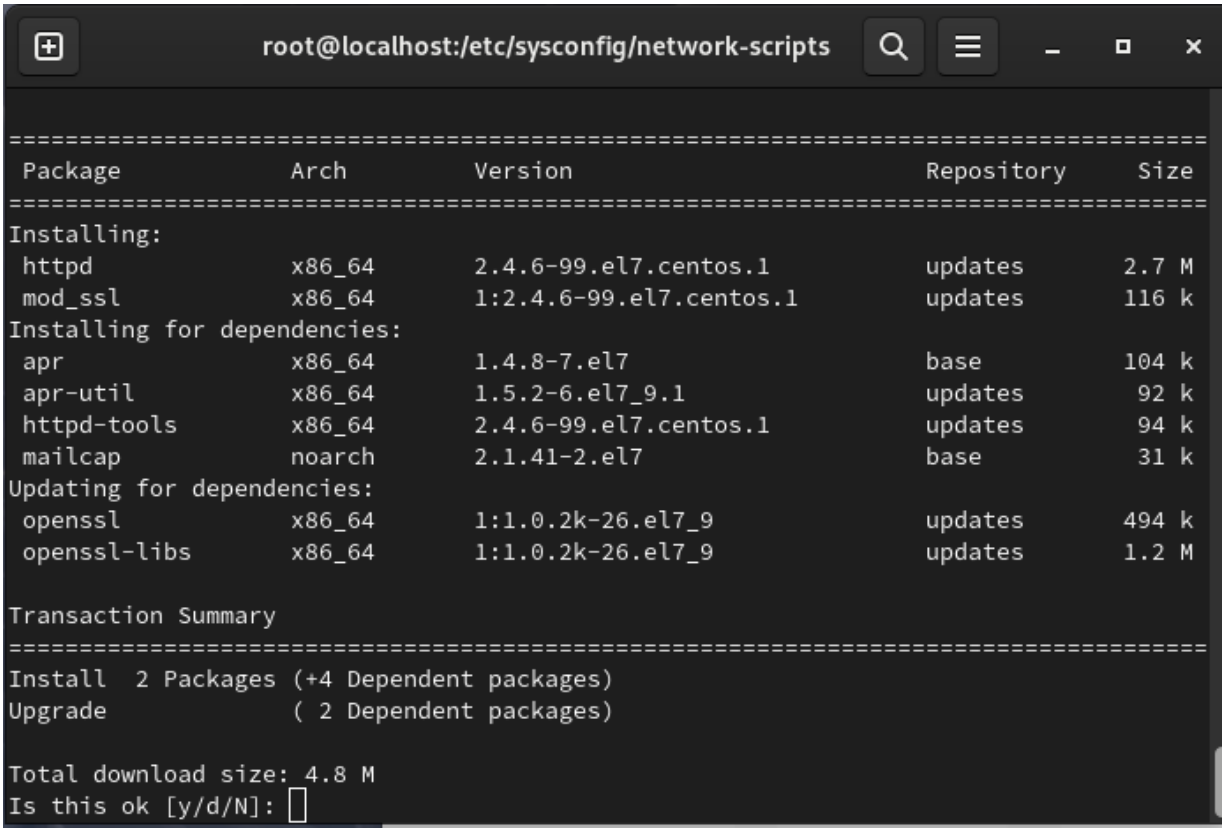
> Εγκατάσταση Apache Web Server (Εικ. 10 – Setup of Apache)

Το όνομα του πακέτου εγκατάστασης για τον Apache είναι το ‘httpd’ . Για να έχουμε τη δυνατότητα κρυπτογράφησης ‘https’ στα API requests μας απαιτείται και η εγκατάσταση του mod_ssl module για τον Apache.

Συνοπτικά, η πιο κάτω εντολή θα εγκαταστήσει τον Apache μαζί με όλα τα λοιπά πακέτα (dependancies) για να έχουμε ένα ολοκληρωμένο Web Server που να υποστηρίζει και κρυπτογράφηση (https).

Σαν ‘root’ χρήστης τρέχουμε την εντολή :

#yum install httpd mod_ssl -y



Εικόνα 10 – Setup of Apache Web Server

Ακολούθως, εκτελούμε την πιο κάτω εντολή για ενεργοποίησης του Web Server

service httpd start

και την εντολή

systemctl enable httpd

η οποία θα ενεργοποιήσει το μηχανισμό του Linux ο οποίος είναι υπεύθυνος να ξεκινά το service με τη φόρτωση του συστήματος (on boot)

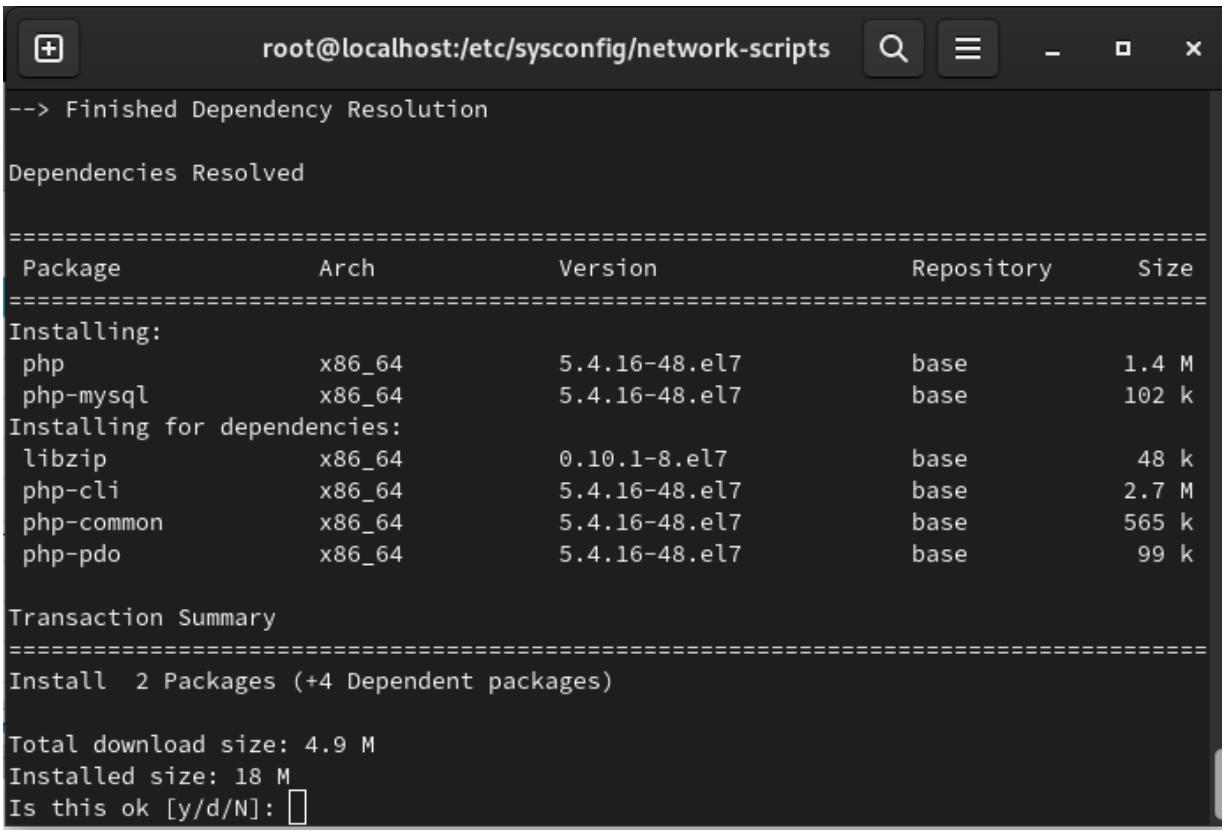
> Εγκατάσταση PHP, php-mysql

Η εγκατάσταση της γλώσσας προγραμματισμού PHP γίνεται μέσω του πακέτου ‘php’ . Στη διατριβή μας χρησιμοποιήθηκε και το module ‘php-mysql’ διότι έχουμε δημιουργήσει και μια βασική Web εφαρμογή (basic interface) για σκοπούς monitoring της συσκευής όσο αφορά τη λειτουργία που αφορά το API και το δυναμικό τείχος προστασίας (Dynamic Firewall) που θέλουμε να παρακολουθούμε. Μέσω της Web εφαρμογής προβάλουμε σε πραγματικό χρόνο τη λειτουργία του API και τη λειτουργία του Firewall.

Η εντολή (Εικ. 11 – Setup of PHP) :

yum install php php-mysql

θα εγκαταστήσει τη γλώσσα προγραμματισμού PHP και το module που χρειάζεται ώστε να επικοινωνούμε με τη βάση δεδομένων μαζί με όλα τα απαιτούμενα πακέτα για τη σωστή λειτουργία του συστήματος .



Εικόνα 11 – Setup of PHP

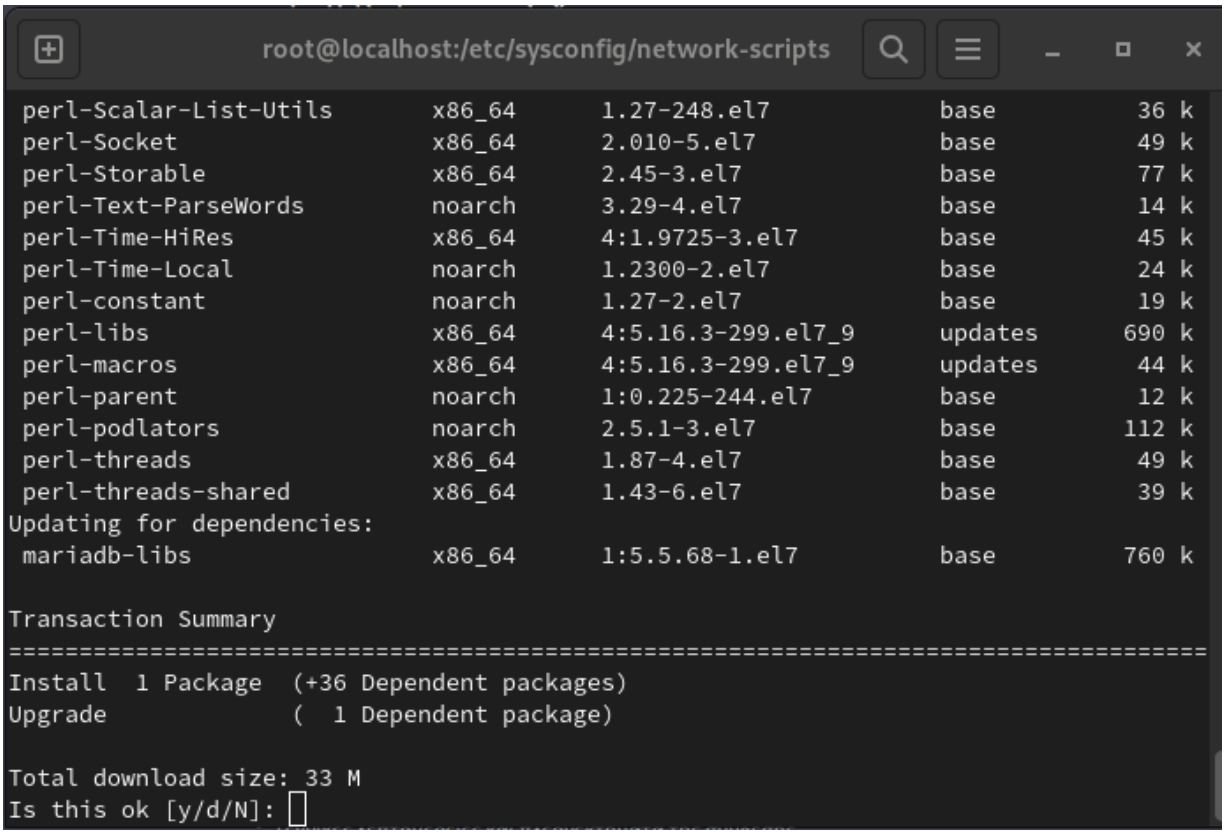
> Εγκατάσταση MariaDB Database server

Το πακέτο εγκατάστασης της βάση δεδομένων που έχει χρησιμοποιηθεί είναι το mariadb-server

Χρησιμοποιώντας την εντολή (Εικ. 12 – Setup of MariaDB) :

#yum install maraidb-server

θα εγκατασταθεί το πακέτο μαζί και οποιαδήποτε άλλα πακέτα χρειάζονται για την λειτουργία του.



Εικόνα 12 – Setup of MariaDB

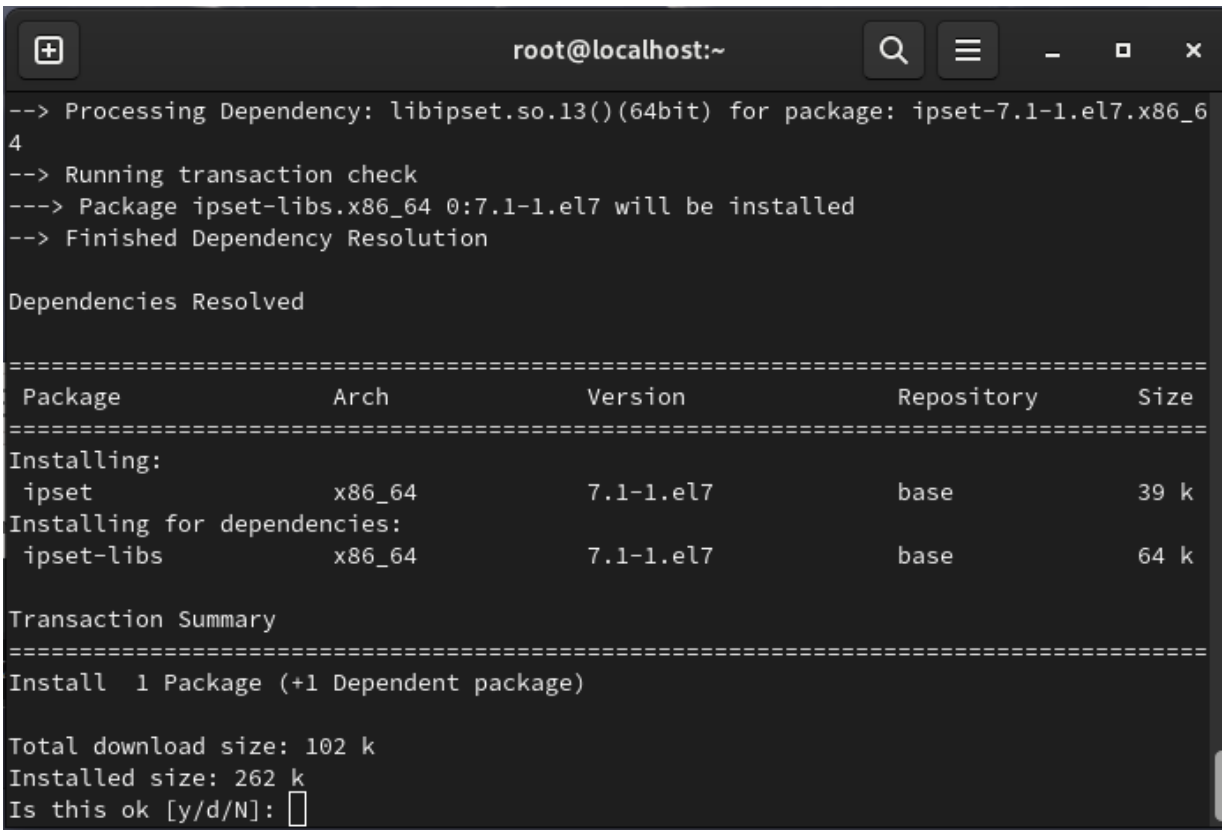
Για την ενεργοποίηση του χρησιμοποιήθηκαν οι πιο κάτω εντολές:

systemctl enable mariadb
#systemctl start mariadb

> Εγκατάσταση IPSET

Για την εγκατάσταση του IPSET module του πακέτου Iptables χρησιμοποιήθηκε επίσης η εντολή ‘yum’ . Το πακέτο αυτό δεν χρειάζεται ενεργοποίηση διότι φορτώνεται αυτόματα από το Iptables μόλις ζητηθεί από το σύστημα .
Η εγκατάσταση γίνεται με την εντολή (Εικ. 13 – Setup of IPSET):

yum install ipset



Εικόνα 13 – Setup of IPSET

Με τα πιο πάνω:

- ✓ έχει ολοκληρωθεί η βάση όσο αφορά τη δημιουργία της προτεινόμενης συσκευής σε επίπεδο λογισμικού και hardware,
- ✓ έχει ολοκληρωθεί η μελέτη για τη λειτουργία του συστήματος,
- ✓ έχουμε δημιουργήσει την απαραίτητη γέφυρα,
- ✓ έχουμε εγκαταστήσει όλα τα λογισμικά που προαπαιτούνται για να προχωρήσουμε με την ανάπτυξη του δικού μας λογισμικού (API & Web Interface) .

3.4 Ενεργοποίηση και Λειτουργία μέσω API

Γενικά

Η γενική λειτουργία των API (Application Program Interfaces) είναι να επιτρέπουν σε διάφορα λογισμικά, συσκευές ή υπηρεσίες να επικοινωνούν μεταξύ τους και να

ανταλλάσσουν δεδομένα. Τα API λειτουργούν σαν γέφυρες μεταξύ διαφορετικών συστημάτων και επιτρέπουν την εκτέλεση λειτουργιών ή την ανάκτηση πληροφοριών από το ένα σύστημα μέσω του άλλου.

Για τη διατριβή μας το API που δημιουργήσαμε είναι η γέφυρα μεταξύ των SIEMs ή οποιουδήποτε συστήματος ικανού να αναλύσει σε επίπεδο δικτύου μια επίθεση και του Linux firewall .

Προβλήματα που καλεστήκαμε να λύσουμε

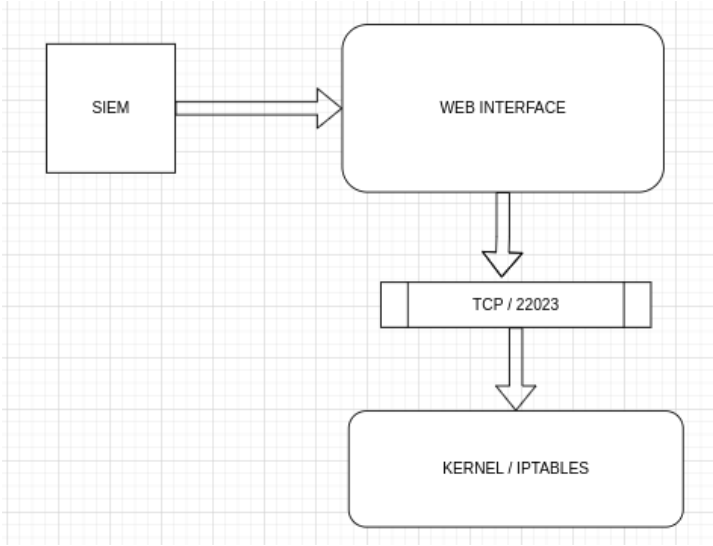
Ένα από τα βασικότερα προβλήματα που καλεστήκαμε να λύσουμε αφορούσε την επικοινωνία μεταξύ του API και του Linux Firewall. Ο κυρίως λόγος είχε να κάνει με τα δικαιώματα (permissions) που έχει ένα service όπως ο WebServer σε σχέση με το λειτουργικό σύστημα, δηλαδή, ένα web service δεν μπορεί να αλλάξει οτιδήποτε ελέγχετε από το λειτουργικό στη βάση του π.χ από το kernel όπως στην προκειμένη περίπτωση το Firewall που σκοπεύαμε να επαναπρογραμματίζουμε on-the-fly .

Για να λύσουμε το πιο πάνω πρόβλημα προχωρήσαμε με τη δημιουργία ενός TCP Service. Το service θα χρησιμοποιηθεί σαν ενδιάμεσο μέσω επικοινωνίας. Τα services γενικά έχουν την ιδιαιτερότητα να μπορούν να δεχτούν εντολές από προγράμματα που τρέχουν σε επίπεδο χρήστη και να τις εκτελέσουν σε επίπεδο root (linux super user) .

Δημιουργία του Service

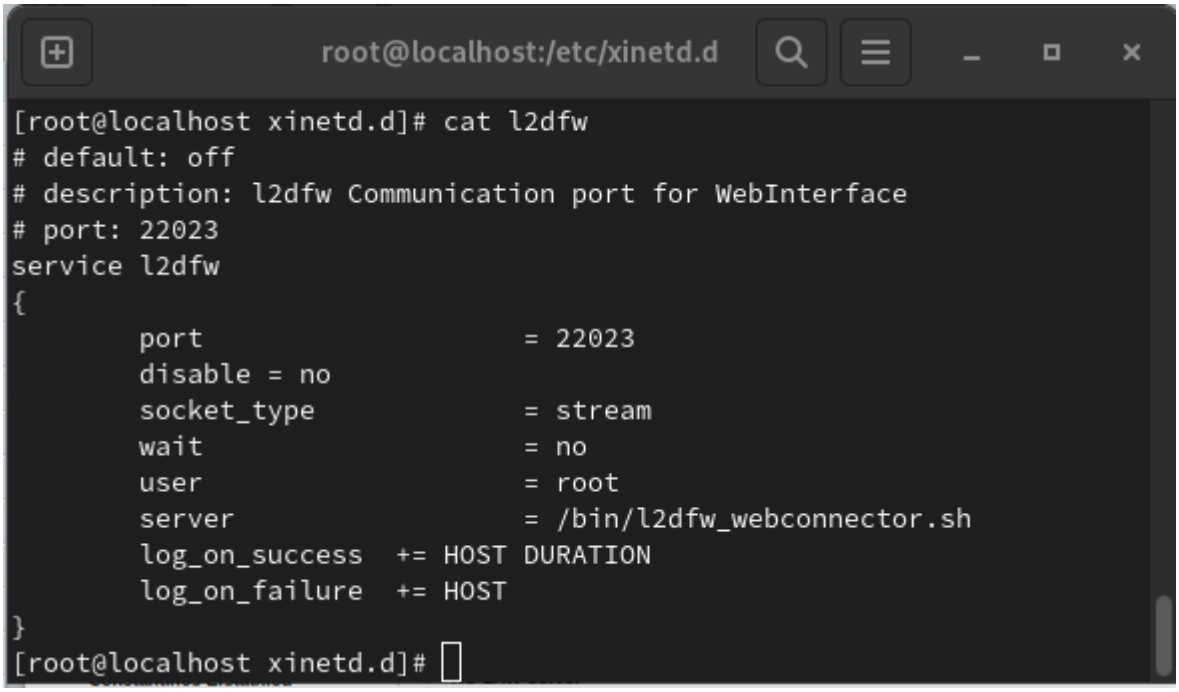
Για τη δημιουργία του δικού μας service χρησιμοποιήσαμε το xinetd (super-server) το οποίο έχει τη δυνατότητα να χειρίζεται άλλα services σε ένα λειτουργικό Linux. Σχεδιάστηκε για τη διαχείριση και επίβλεψη εφαρμογών δικτύου , επιτρέποντας την εκκίνηση τους κατά απαίτηση (on-request) .

Στη δική μας περίπτωση, η δική μας εφαρμογή μέσω του WebServer (apache) θα καλεί το service μας (@ localhost) το οποίο με τη σειρά του θα εκτελεί την ανάλογη εντολή σε επίπεδο super-user ώστε να γίνεται η αλλαγή στο Firewall (επίπεδο πυρήνα). (Εικ. 14 – Data Flow)



Εικόνα 14 – Data Flow

Το service που δημιουργήσαμε (Εικ. 15 – TCP Service) ακούει στη πόρτα (port) 22023 TCP. Στη περίπτωση που κάποια εφαρμογή ενωθεί και στείλει πληροφορίες στη συγκεκριμένη πόρτα, τότε θα εκτελεστεί το script που ορίζεται σαν server (/bin/l2dfw_webconnector.sh) . Το script αυτό μπορεί θα περιλαμβάνει τις ανάλογες εντολές που πρέπει να εκτελεστούν σε επίπεδο super user.
Η επιλογή του αριθμού 22023 είναι τυχαία.



Εικόνα 15 – TCP Service

To Web Interface

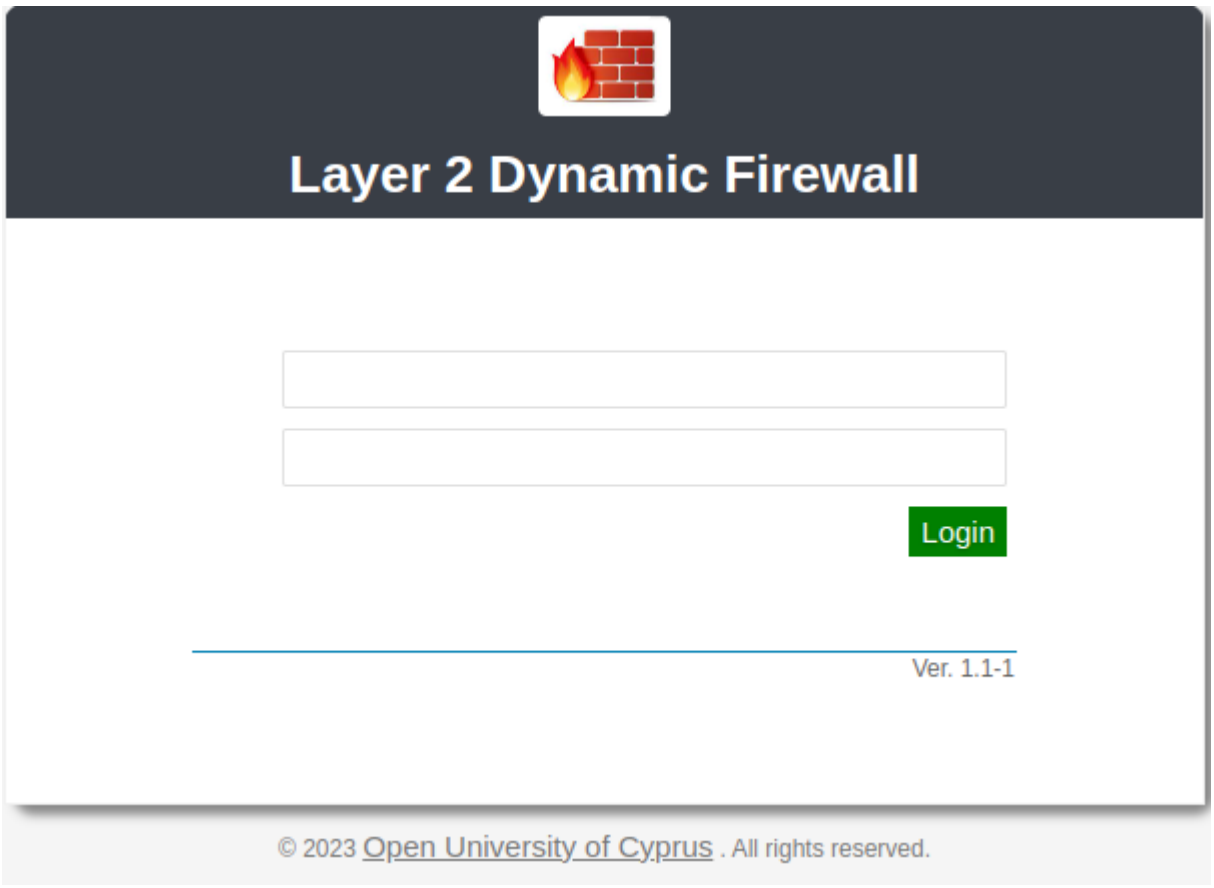
Για το WebInterface χρησιμοποιήθηκε η γλώσσα προγραμματισμού PHP. Έχουμε δημιουργήσει, πέραν του login screen, μια κεντρική σελίδα στην οποία παρουσιάζουμε σε πραγματικό χρόνο το status του συστήματος, και το API που δέχεται remote connections από τα διάφορα περιφερειακά.

Το Authentication τόσο του Remote API όσο και του χρήστη που θέλει να δει το status του συστήματος γίνεται μέσω MariaDB βάσης δεδομένων.

Login Screen

Το Login Screen (Εικ. 16 – Login Screen) υποστηρίζει username/password τα οποία έχουμε φορτώσει στη βάση δεδομένων μας. Έχουμε επίσης χρησιμοποιήσει τεχνικές αποφυγής SQL Injection ή παρόμοιων επιθέσεων στο μηχανισμό Authentication μας όπως πιο κάτω:

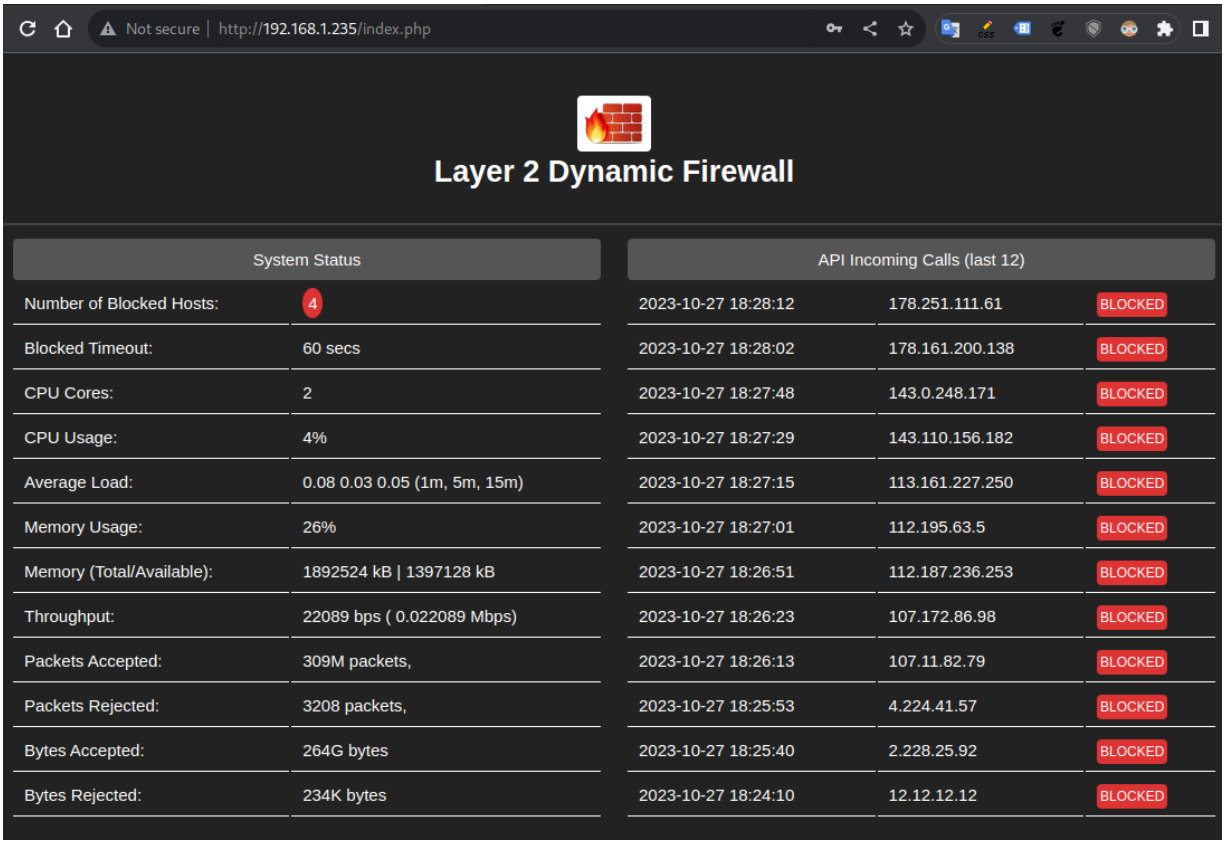
```
$login=$_POST['login_user'];
$syskey=$_POST['login_password'];
$syskey=md5($syskey);
//SQL INJECTION PROTECTION
$login = mysql_real_escape_string($login);
$syskey = mysql_real_escape_string($syskey);
```



Εικόνα 16 – Login Screen

Κεντρική Σελίδα

Η κεντρική σελίδα (Εικ. 17 – Main Screen) αποτελείται από δύο μέρη , στα αριστερά το ‘System Status’ το παρουσιάζει σε πραγματικό χρόνο το status του συστήματος από πλευράς πόρων (resources) και δεξιά το ‘API Incoming Calls’ στο οποίο φαίνονται επίσης σε πραγματικό χρόνο οι τελευταίες εντολές από περιφερειακά συστήματα στο API μας.



Εικόνα 17 – Main Screen

Βάση Δεδομένων

Η βάση δεδομένων που χρησιμοποιήθηκε είναι η MariaDB . Η MariaDB είναι relational database με εξαιρετική απόδοση ιδανική για απλά Web Applications. Είναι σχετικά εύκολη στη χρήση και προσφέρετε δωρεάν.

Στη βάση δεδομένων (Εικ. 18 – Βάση Δεδομένων) δημιουργήσαμε δύο ‘DB Tables’, ένα που χρησιμοποιούμε για το Authentication χρήστη και του API και ένα για να φυλάγουμε πληροφορίες (Logs) σχετικά με τις εντολές που δέχεται το API .

Στο πρώτο table με την ονομασία ‘auth’ για το χρήστη admin έχουμε το password σε md5 hash μορφή μαζί με ένα τυχαίο ‘ vcode’ το οποίο είναι το ‘key’ για το συγκεκριμένο χρήστη που μπορεί να χρησιμοποιεί σαν authentication στο API.

```
MariaDB [(none)]> use l2dfw
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [l2dfw]> show tables;
+-----------------+
| Tables_in_l2dfw |
+-----------------+
| auth            |
| l2dfw_logs      |
+-----------------+
2 rows in set (0.00 sec)

MariaDB [l2dfw]> select * from auth;
+----+----------+----------------------------------+------------+---------+-------------------+
| id | username | password                         | systemname | active  | vcode             |
+----+----------+----------------------------------+------------+---------+-------------------+
|  1 | admin    | 147245d2aee44f57134ce2285d541511 | System1    | checked | 5RjHmeprAfSzLDyr7 |
+----+----------+----------------------------------+------------+---------+-------------------+
1 row in set (0.00 sec)

MariaDB [l2dfw]> select * from l2dfw_logs limit 3;
+----+-------+---------+---------------------+-------------+--------------+----------+-----+-------+-------+-------+--------+-------------------------+
| id | sysid | logtype | datetime            | interfacein | interfaceout | src      | dst | proto | sport | dport | action | loginfo                 |
+----+-------+---------+---------------------+-------------+--------------+----------+-----+-------+-------+-------+--------+-------------------------+
|  1 |     0 | INFO    | 2023-10-11 15:18:57 | NULL        | NULL         | 10.1.1.1 |     | NULL  | NULL  | NULL  | BLOCK  | Host [10.1.1.1] blocked |
|  2 |     0 | INFO    | 2023-10-11 15:19:01 | NULL        | NULL         | 10.1.1.1 |     | NULL  | NULL  | NULL  | BLOCK  | Host [10.1.1.1] blocked |
|  3 |     0 | INFO    | 2023-10-11 15:20:03 | NULL        | NULL         | 10.1.1.1 |     | NULL  | NULL  | NULL  | BLOCK  | Host [10.1.1.1] blocked |
+----+-------+---------+---------------------+-------------+--------------+----------+-----+-------+-------+-------+--------+-------------------------+
```

Εικόνα 18 – Βάση Δεδομένων

Το δευτερο ‘DB table’ με την ονομασία l2dfw_logs χρησιμοποιείται για την αποθήκευση logs . Το συγκεκριμένο table περιλαμβάνει πληροφορίες σχετικά με τις εντολές που δέχεται το API από τα διάφορα περιφερειακά συστήματα SIEMs. Η διαρρύθμιση του έγινε με σκοπό να μπορεί να φυλάξει μελλοντικά και άλλες πληροφορίες όπως π.χ τα Firewalls Logs.

API (Application Program Interface)

Το API που δημιουργήσαμε είναι ένα σχετικά απλό λογισμικό που θα είναι ο συνδετικός κρίκος μεταξύ των περιφερειακών συστημάτων και του δικού μας συστήματος. Όπως και το υπόλοιπο λογισμικό , είναι γραμμένο σε PHP. Για να το χρησιμοποιήσουμε καλούμε το σύστημα μας στο https://<ip address>/api/index.php

Δέχεται σαν input (POST) παραμέτρους τις ακόλουθες:

- **vcode=** : Αυτό είναι το κλειδί (authentication key) που είναι δεμένο σε κάθε χρήστη. Χρησιμοποιείτε για να γίνει η αυθεντικοποίηση του συστήματος που ενώνεται στο API. Η τιμή (value) που δέχεται είναι alpanumeric 16 χαρακτήρων.
- **action=** : Το action που καλείται το API να εκτελέσει. Οι τιμές που δέχεται είναι οι

ακόλουθες:

- BLOCK : (action=BLOCK) δηλαδή καλούμε το API να εκτελέσει ενέργεια αποκοπής κάποιας IP διεύθυνσης
- UNBLOCK : χρησιμοποιείται για να κάνουμε UNBLOCK κάποια IP διεύθυνση.
- INIT : χρησιμοποιείται για το 'initialization' του συστήματος. Όταν το σύστημα δεχτεί αυτή τη παράμετρο τότε θα δημιουργήσει όλα τα απαραίτητα IPTABLES rules (Linux Firewall rules σε επίπεδο πυρήνα) και θα προετοιμάσει το σύστημα για να δεχτεί περεταίρω εντολές για BLOCK ή UNBLOCK κάποιας διευθύνσης IP. Το action=init στέλνεται σε περιπτώσεις restart του συστήματος.
- **- IP=** : η διεύθυνση IP που θέλουμε να κάνουμε BLOCK ή UNBLOCK

Παράμετροι Initialization του συστήματος (action=init)

Το σύστημα για να είναι σε θέση αποκοπής επίθεσης, πρέπει να φορτώσουμε τους απαραίτητους κανόνες σε επίπεδο Firewall. Για το initialization του συστήματος στέλνουμε μέσω του API το action=init όπως πιο κάτω.

#curl -s -k -d "vcode=5RjHmeprAfSzLDyr7" -d "action=init" -d "ip=" https://192.168.1.235/api/index.php

Νοουμένου ότι το vcode (authentication key) είναι σωστό το API μέσω ενός εσωτερικού function (Εικ. 19 – API Main Function Call to Service) θα καλέσει εσωτερικά (localhost) στη πόρτα 22023 TCP το service το οποίο δημιουργήσαμε προηγουμένως. Το service με τη σειρά του θα τρέξει την εντολή **/bin/l2dfw_webconnector.sh .** Στη διαδικασία θα σταλούν αυτόματα και οι τιμές που αφορούν τις παραμέτρους ACTION IPADDRESS . Το λογισμικό (application) που χρησιμοποιείται για το connection μεταξύ του Web Interface και του service μας είναι το **nc (netcat) .**

```
function maincmd ($ACTION,$IPADDR){
        $COMMAND="echo $ACTION $IPADDR |nc -w 5 localhost 22023";
        $retval=shell_exec("$COMMAND"." 2>&1");
        #exec("$COMMAND","2>&1","$status");

        return "$retval";
};
```

Εικόνα 19 – API Main Function Call to Service

Η πιο πάνω διαδικασία θα δημιουργήσει τους απαραίτητους κανόνες (Firewall rules) που θα εγκατασταθούν στο Bridge Br0 που επίσης δημιουργήσαμε. Το script

/bin/ledfw_webconnector.sh αφού δεχτεί τη παράμετρο action=init θα εκτελέσει τις ακόλουθες δύο εντολές :

- /sbin/ipset create L2DFW iphash timeout 160
- /sbin/iptables -A FORWARD -m physidev –physdev-in enp1s0 -m set –match-set L2DFW src -j REJECT –reject-with icmp-host-unreachable

Η πρώτη εντολή θα χρησιμοποιήσει το εργαλείο ipset για να δημιουργήσει ένα σύνολο διαχειρίσιμων διευθύνσεων (data structure) με το όνομα L2DFW βασισμένο σε συνάρτηση κατακερματισμού για να αποθηκεύει και να διαχειρίζεται αποτελεσματικά μεγάλο όγκο IP διευθύνσεων.

Σημαντική παράμετρος σε αυτή την εντολή είναι και το **timeout 160 .** Η παράμετρος αυτή χρησιμοποιήθηκε επειδή θέλουμε οι IP διευθύνσεις μετά από ένα ορισμένο διάστημα – το timeout – αυτόματα να ελευθερώνονται από το σύστημα για πρακτικούς λόγους αλλά και για να ελευθερώνονται resources.

Η δεύτερη εντολή θα δημιουργήσει το iptables rule, δηλαδή το κανόνα που θα αποκόπτει τον εισβολέα σε επίπεδο Firewall. Ο κανόνας αυτός συνδυάζει τη δυνατότητα το Linux Firewall, τη φυσική κάρτα και το εργαλείο ipset.

Πιο αναλυτικά, ο κανόνας ελέγχει τη κίνηση (traffic) η οποία εισέρχεται από τη φυσική κάρτα δικτύου enp1s0 και εξέρχεται από τη φυσική κάρτα enp2s0 (οι δύο κάρτες που αποτελούν τη γέφυρα που φτιάξαμε – br0) . Ελέγχει το src IP σε συνάρτηση με το ipset data structure (L2DFW) και αν βρει το IP τότε το κάνει REJECT . Σε αντίθετη περίπτωση η κίνηση θα περάσει.

Για να βεβαιωθούμε ότι οι πιο πάνω κανόνες έχουν εγκατασταθεί επιτυχώς τρέχουμε τις εντολές :

 # iptables -L
 # ipset -L

Εικόνα 20 – Iptables,Ipset

Οι πιο πάνω εντολές (Εικ. 20 – Iptables,Ipset) θα μας δείξουν το status τόσο του IPTABLES firewalls όσο και του IPSET module.

Παράδειγμα χρήσης του API από περιφερειακά συστήματα.

Ένας από τους βασικούς στόχους της διατριβής μας ήταν η δημιουργία ενός γενικού συστήματος το οποίο θα μπορούσε εύκολα να συνδεθεί με πολλά περιφερειακά συστήματα. Η σύνδεση όπως επεξηγήσαμε γίνεται με το API το οποίο έχουμε δημιουργήσει.

Το Linux , αλλά και γενικά όλα τα λειτουργικά με δυνατότητες συνδεσμολογίας μέσω API υποστηρίζουν κάποιους standard τρόπους, όπως π.χ μέσω της εντολής 'curl' . Η curl (Client URL) είναι ένα ευρέως χρησιμοποιούμενο εργαλείο που υποστηρίζει διάφορα πρωτόκολλα δικτύου, συμπεριλαμβανομένων των http, https, ftp,sftp και πολλών άλλων. Μπορεί να χρησιμοποιηθεί για download ή upload δεδομένων χρησιμοποιώντας διάφορα πρωτόκολλα από τη γραμμή εντολών.

#curl -s -k -d "vcode=5RjHmeprAfSzLDyr7" -d "action=block" -d "ip=145.12.12.1" https://192.168.1.235/api/index.php

Η πιο πάνω εντολή θα μπορούσε να εκτελεστεί σε κάποιο SIEM σύστημα, ή γενικά σε οποιοδήποτε σύστημα με οποιοδήποτε λειτουργικό για να στείλει εντολή αποκοπής της διεύθυνσης IP 145.12.12.1 . Η εντολή περιλαμβάνει επίσης το action (action=block) αλλά και το authentication key (vcode=5R..) και το URL που βρίσκεται το API μας.

Αφού γίνει η ταυτοποίηση του συστήματος (authentication) το API μέσω του εσωτερικού service θα προσθέσει στο ipset τη διεύθυνση 145.12.12.1 .

Κεφάλαιο 4

Εκτέλεση Πειραμάτων

4.1 Πειράματα Απόδοσης

Η επιτυχής λειτουργία της συσκευής μας είναι στενά συνδεδεμένη με την αξιολόγηση και τον έλεγχο της απόδοσής της σε πραγματικά σενάρια. Μέσω της προσομοίωσης πραγματικών συνθηκών, τα πειράματα που πρόκειται να πραγματοποιηθούν επιδιώκουν να διερευνήσουν την απόδοση της συσκευής υπό διάφορες συνθήκες και φορτία εργασίας (load).

Μέσω της επιβάρυνσης και της προσπάθειας υπερφόρτωσης της συσκευής, σκοπεύουμε να αξιολογήσουμε την αντοχή, τη σταθερότητα και την απόδοσή της. Η διεξαγωγή αυτών των πειραμάτων αποτελεί κρίσιμη διαδικασία για την επιβεβαίωση της ορθής λειτουργίας της συσκευής και τη διασφάλιση της απόδοσής της σε πραγματικές συνθήκες χρήσης.

Στα πειράματα μας χρησιμοποιήσαμε αυτοματοποιημένα scripts για την άμεση και παράλληλη φόρτωση πολλών διευθύνσεων IP σε σχετικά λίγο χρονικό διάστημα. Τα scripts έτρεξαν και τοπικά και απομακρυσμένα. Η εκτέλεση σεναρίων απομακρυσμένης σύνδεσης είναι προσομοίωση σε πραγματικές συνθήκες όταν η σύνδεση γίνεται π.χ από κάποιο SIEM .

4.1.1 25K IPs Vs Performance – Remote SIEM

Σκοπός αυτού του τεστ ήταν να φορτώσουμε χιλιάδες διευθύνσεις IP από άλλο σύστημα μέσω του δικτύου. Με το τρόπο αυτό θα προσομοιώσουμε ένα ή πολλά SIEMs που στέλνουν εντολές block/unblock στο σύστημα μας.

Για να διεξαγωγή του πιο πάνω πειράματος χρησιμοποιήθηκε μία λίστα (αρχείο – 1 IP σε κάθε γραμμή) με ~ 25K IPs που για τις οποίες με ένα απλό script καλούσαμε το API μας για κάθε IP ξεχωριστά .

Το αρχείο (ban_ips.txt) είχε τη πιο κάτω δομή (sample)

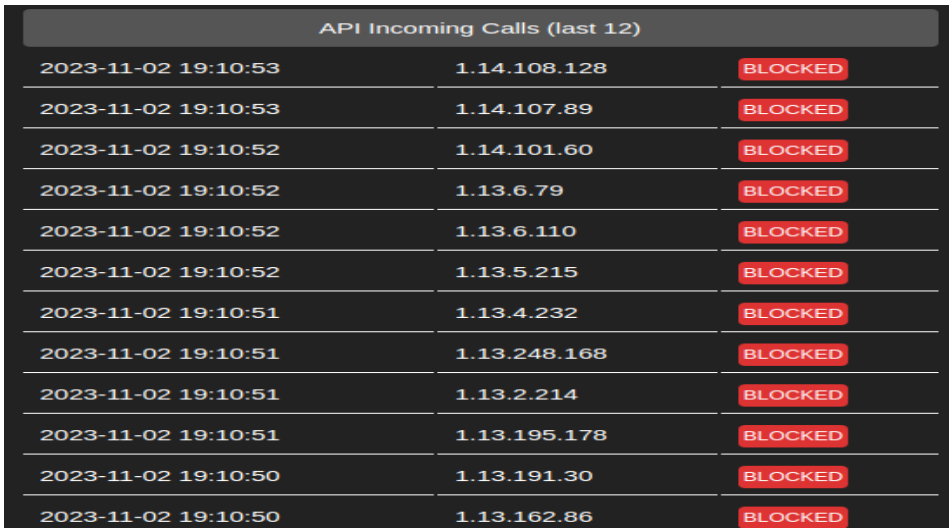
```
107.173.160.135
107.173.160.149
107.173.160.154
107.174.172.198
107.174.186.25
107.174.243.101
```

Το script που χρησιμοποιήσαμε:

```
for IP in `cat ban_ips.txt`
        do
        curl -s -k -d "vcode=5RjHmeprAfSzLDyr7" -d "action=block" -d "ip=$IP" https://192.168.1.235/api/index.php
done
```

Το πιο πάνω script θα καλεί το API μας μία φορά για κάθε ένα 1 IP . Τα τεστ μας έδειξαν ότι οι δυνατότητες του συστήματος μας (Εικ. 21 – API Remote Connections) να δεχτεί παράλληλες συνδέσεις (connections) από περιφερειακά συστήματα ήταν 4-5 το δευτερόλεπτο.

Ο περιορισμός των 4-5 παράλληλων συνδέσεων το δευτερόλεπτο οφείλεται στις δυνατότητες του WebServer μαζί με το TCP service που δημιουργήσαμε σε σχέση πάντα με τις δυνατότητες του συστήματος (hardware) που επιλέξαμε.



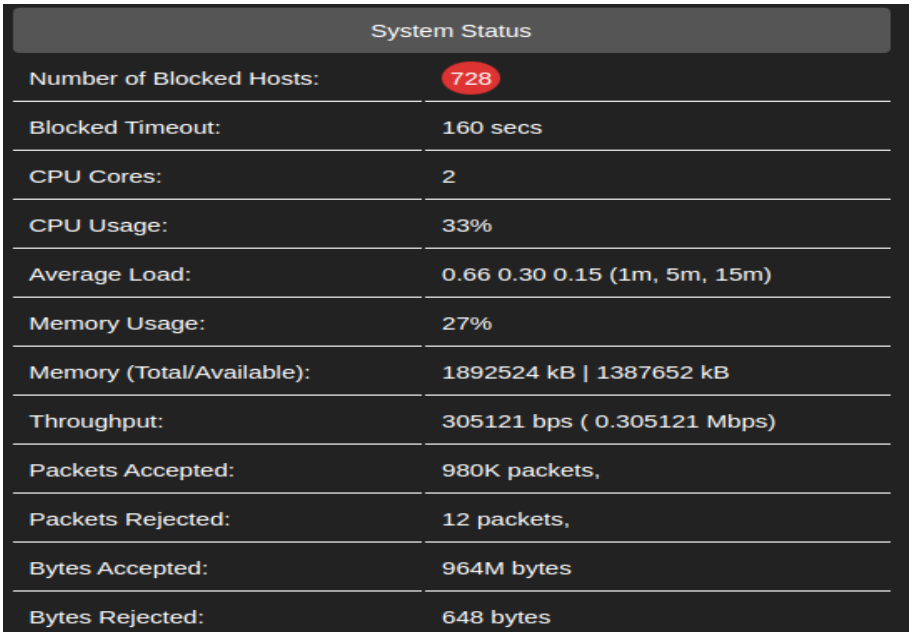
API Incoming Calls (last 12)

2023-11-02 19:10:53	1.14.108.128	BLOCKED
2023-11-02 19:10:53	1.14.107.89	BLOCKED
2023-11-02 19:10:52	1.14.101.60	BLOCKED
2023-11-02 19:10:52	1.13.6.79	BLOCKED
2023-11-02 19:10:52	1.13.6.110	BLOCKED
2023-11-02 19:10:52	1.13.5.215	BLOCKED
2023-11-02 19:10:51	1.13.4.232	BLOCKED
2023-11-02 19:10:51	1.13.248.168	BLOCKED
2023-11-02 19:10:51	1.13.2.214	BLOCKED
2023-11-02 19:10:51	1.13.195.178	BLOCKED
2023-11-02 19:10:50	1.13.191.30	BLOCKED
2023-11-02 19:10:50	1.13.162.86	BLOCKED

Εικόνα 21 – API Remote Connections

Λαμβάνοντας υπόψη τη παράμετρο **timeout = 160** (δηλαδή το κάθε IP θα είναι blocked για 160 δευτερόλεπτα) και σε συνάρτηση με τη δυνατότητα του για αποδοχή 4-5 παράλληλες συνδέσεις (concurrent connections) το δευτερόλεπτο από

περιφερειακά συστήματα, το σύστημα θα μπορεί να διαχειρίζεται περίπου 700 IPs σε παράθυρα των 160 δευτερολέπτων (Εικ. 22 – System Performance).



System Status	
Number of Blocked Hosts:	728
Blocked Timeout:	160 secs
CPU Cores:	2
CPU Usage:	33%
Average Load:	0.66 0.30 0.15 (1m, 5m, 15m)
Memory Usage:	27%
Memory (Total/Available):	1892524 kB | 1387652 kB
Throughput:	305121 bps (0.305121 Mbps)
Packets Accepted:	980K packets,
Packets Rejected:	12 packets,
Bytes Accepted:	964M bytes
Bytes Rejected:	648 bytes

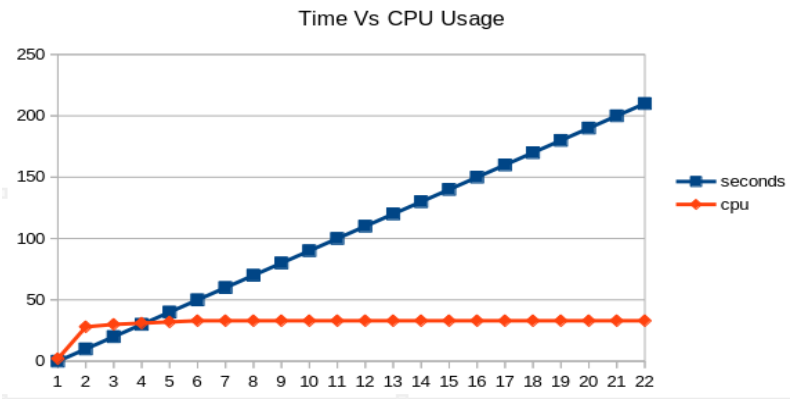
Εικόνα 22 – System Performance

Για να υπολογίσουμε την απόδοση του συστήματος πήραμε είκοσι δύο (22) μετρήσεις (μία μέτρηση κάθε δέκα δευτερόλεπτα) στη διάρκεια που έτρεχε το πιο πάνω script. Με τις μετρήσεις μας θέλαμε να μελετήσουμε πως επηρεάζεται το CPU και η μνήμη σε συνάρτηση με των αριθμό των IPs που φορτώναμε στο σύστημα.

a/a	Seconds	Number of Ips	CPU %	Memory %
1	0	0	2	26
2	10	65	28	26
3	20	138	30	26
4	30	203	31	27
5	40	260	32	27
6	50	312	33	27
7	60	355	33	27
8	70	402	33	27
9	80	451	33	27
10	90	496	33	27
11	100	553	33	27
12	110	585	33	27
13	120	620	33	27
14	130	655	33	27
15	140	702	33	27
16	150	740	33	27
17	160	722	33	27
18	170	725	33	27
19	180	715	33	27
20	190	726	33	27
21	200	718	33	27
22	210	729	33	27

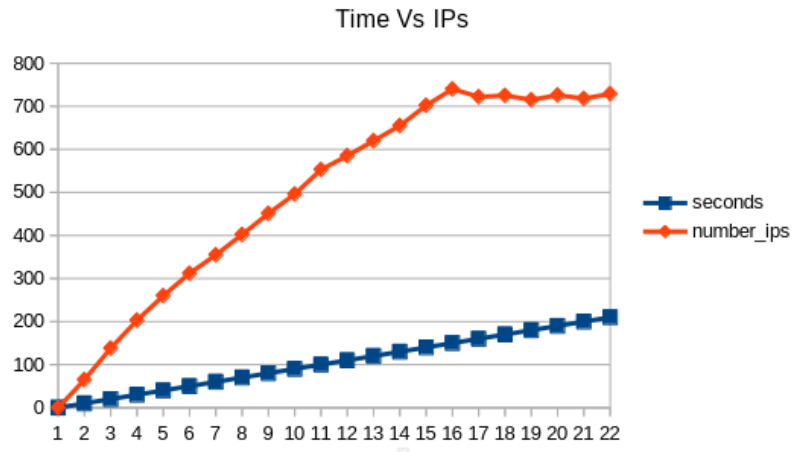
Εικόνα 23 – Πίνακας Μετρήσεων

Όπως χαρακτηριστικά φαίνεται και από το πιο πάνω πίνακα (Εικ. 23 – Πίνακας Μετρήσεων) , το σύστημα ανταποκρίθηκε πολύ ικανοποιητικά μέσα στα πλαίσια των δυνατοτήτων του. Το CPU δεν ξεπέρασε το 33% και η μνήμη παρέμεινε σταθερά στο 27% .



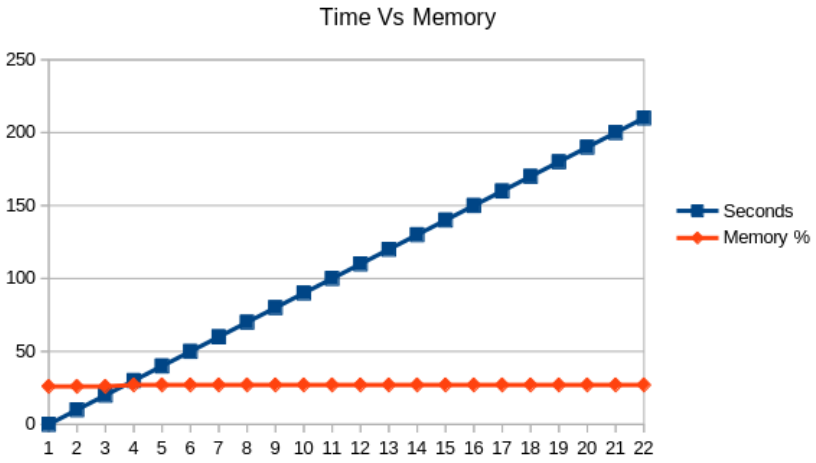
Εικόνα 24 - Time Vs CPU Usage

Στη πιο πάνω γραφική παράσταση (Εικ. 24 – Seconds Vs CPU Usage) φαίνεται ότι το CPU φτάνει στο 33% δευτερόλεπτα μετά που το σύστημα αρχίζει να δέχεται πολλαπλές συνδέσεις. Μετά, ανεξαρτήτως των IPs που φορτώνονται παραμένει σταθερό.



Εικόνα 25 – Time Vs IPs

Όπως έχουμε προαναφέρει η παράμετρος **timeout 160** διαδραματίζει ουσιαστικό ρόλο στην απόδοση του συστήματος αλλά και συνάμα το προστατεύει από υπερφόρτωση. Στην γραφική παράσταση Time Vs IPs (Εικ. 25 – Time Vs IPs) παρατηρούμε ότι το σύστημα στο σημείο 16 (150 δευτερόλεπτα) βρίσκεται στο σημείο σταθερότητας του όπου τα αρχικά IPs που φορτώσαμε ξεκινούν να κάνουν expired. Από το σημείο αυτό ξεκίνα το σύστημα να κινείται υπό τη μορφή 'Κινούμενων Παραθύρων' (Moving Windows) και να σταθεροποιείτε πλήρως .



Εικόνα 26 – Time Vs Memory

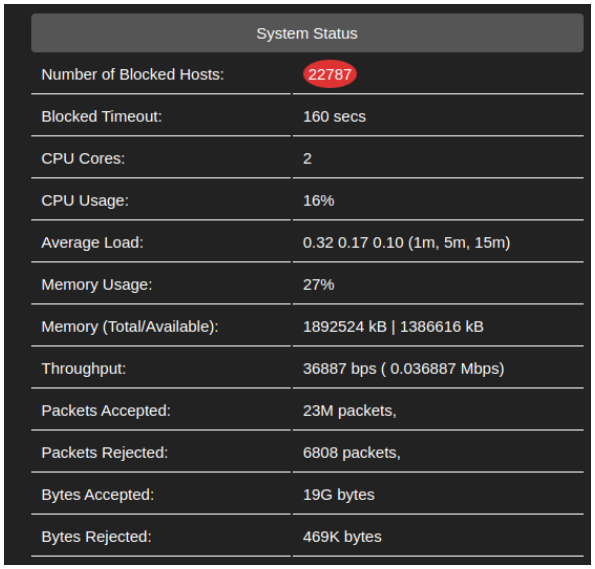
Η πιο πάνω γραφική παράσταση (Εικ. 26 – Time Vs Memory) δείχνει τη σταθερότητα που έχει η μνήμη του συστήματος η οποία αποδεικνύεται ότι δεν επηρεάζεται καθόλου.

4.1.2 25K IPs Vs Performance - Local

Σκοπός του τεστ είναι να ελέγξουμε κατά πόσο η απόδοση του συστήματος επηρεάζεται με τη φόρτωση μερικών χιλιάδων IPs κατευθείαν στο σύστημα (χωρίς τη χρήση του API – τοπικά). Για το τεστ χρησιμοποιήσαμε λίστα με χιλιάδες IPs την οποία φορτώσαμε στο σύστημα μας τοπικά χρησιμοποιώντας το ακόλουθο scipt :

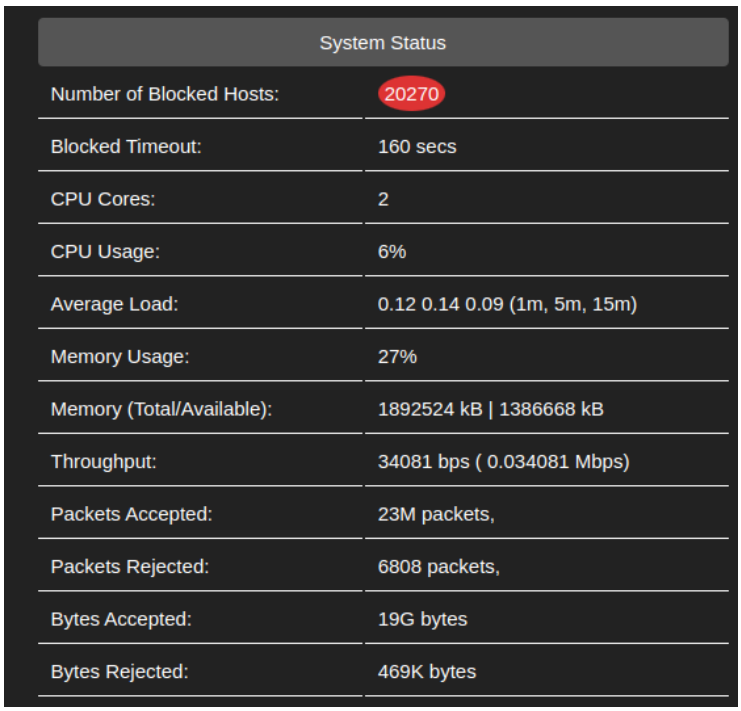
```
for IP in `cat ban_ips.txt`
do
ipset -! add L2DFW $IP
done
```

Κατά τη διάρκεια του πιο πάνω τεστ παρατηρήσαμε ότι το σύστημα ανταποκρίθηκε πάρα πολύ καλά και το CPU του δεν ξεπέρασε το 20% κατά τη διάρκεια που φορτώναμε τα IPs στο IPSET (Εικ. 27 - .Κατά τη διάρκεια της φόρτωσης IPs)



Εικόνα 27 – Κατά τη διάρκεια της φόρτωσης IPs

Ακολούθως, όπως βλέπουμε και από την Εικόνα 28 (Εικ. 28 – Μετά που φορτώσαμε τα IPs), η απόδοση του συστήματος δεν επηρεαζόταν καθόλου από τον αριθμό των IPs που ήταν φορτωμένα .



Εικόνα 28 – Μετά που φορτώσαμε τα IPs

4.1.3 Firewall Logs

Για να ελέγξουμε την αποτελεσματικότητα του συστήματος φορτώσαμε μέσω του API μας κάποια απομακρυσμένη IP διεύθυνση (213.7.196.211) που γνωρίζαμε και από εκεί προσπαθήσαμε να καλέσουμε τη δική μας IP διεύθυνση (93.109.180.207) στη πόρτα 80. Το firewall μας αντίδρασε και σταμάτησε την ένωση (connection) όπως αναμενόταν. Ταυτόχρονα δημιουργήθηκε το ακόλουθο log στο /var/log/messages αρχείο :

```
Nov  8 19:40:54 localhost kernel: L2DFW IN=br0 OUT=br0 PHYSIN=enp1s0 PHYSOUT=enp2s0
MAC=52:54:00:ee:6b:80:24:0b:88:04:21:4d:08:00 SRC=213.7.196.211 DST=93.109.180.207 LEN=52
TOS=0x00 PREC=0x20 TTL=52 ID=29457 DF PROTO=TCP SPT=33084 DPT=80 WINDOW=32 RES=0x00
ACK FIN URGP=0
```

Αναλύοντας λεπτομερώς το πιο πάνω log βλέπουμε ότι η κίνηση (traffic) είχε σαν IN και OUT τη γέφυρα br0 με τη φυσική κάρτα enp1s0 σαν INBOUND INTERFACE (PHYSIN) και την φυσική κάρτα enp2s0 σαν OUTBOUND INTERFACE (PHYSOUT). Επίσης, φαίνονται και τα Source και Destination Ips και η πόρτα (80) που έγινε το block.

4.1.4 Σύγκριση αποτελεσμάτων / πειράματων

Με τα πιο πάνω πειράματα διαφάνηκε ότι η απόδοση του Firewall μας χρησιμοποιώντας το συνδιασμό IPTABLES/IPSET δεν επηρεάζεται ιδιαίτερα από των αριθμό των IPs που φορτώνουμε τουλάχιστον μέχρι τον αριθμό των 25K που ήταν τα τεστ μας.

Με βάση το πιο πάνω και λαμβάνοντας υπόψη τις δυνατότητες rule expiration (blocked timeout 160 secs) μπορούμε να συμπεράνουμε με ασφάλεια ότι η συσκευή που φτιάξαμε μπορεί να ανταποκριθεί επαρκώς και σε πιο μεγάλα περιβάλλοντα.

Ο λόγος που δεν καταφέραμε να φορτώσουμε τον ίδιο αριθμό IPs και απομακρυσμένα οφείλεται στις δυνατότητες του WebServer του συστήματος να διαχειριστεί τις πολλαπλές ενώσεις (multiple concurrent connections). Φυσικά οι 4-5 ενώσεις ανά δευτερόλεπτο είναι εξαιρετικά ψηλός αριθμός αν σκεφτούμε ότι υπό πραγματικές συνθήκες θα μπορούμε να διαχειριστούμε 4-5 καινούργιες επιθέσεις το δευτερόλεπτο.

Κεφάλαιο 5
Επίλογος

5.1 Επίλογος

Σε αυτή τη μεταπτυχιακή διατριβή προσπαθήσαμε να δημιουργήσουμε ένα Δυναμικό Τείχος Προστασίας Χαμηλού Κόστους(Low Cost Dynamic Firewall) τύπου appliance που θα μπορεί να χρησιμοποιείται από διάφορα Cyber Security συστήματα σαν το μέσω εκτέλεσης των αποτελεσμάτων ανίχνευσης τους π.χ αποκοπεί μίας εισβολής από συγκεκριμένο attacker (διεύθυνση IP) στα συστήματα ενός οργανισμού.

Αφού μελετήσαμε τις υπάρχουσες τεχνολογίες και τις δυνατότητες τους στο πλαίσιο αποκοπής κάποιας κυβερνοεπίθεσεις παρατηρήσαμε ότι των πλείστων οι δυνατότητες σταματούσαν στον εντοπισμό της επίθεσης και δεν έπαιρναν κάποιο περαιτέρω action π.χ να έχουν κάποιο τρόπο να σταματήσουν τον επιτιθέμενο αποκόπτοντας των μέσω κάποιου είδους Firewall rule ή κάποιου παρόμοιου μηχανισμού.

Σε όσα συστήματα υπήρχε η δυνατότητα αποκοπής γινόταν είτε στο ίδιο το μηχάνημα όπως π.χ ένα in-line Ips σύστημα είτε παρεχόταν η δυνατότητα να αποσταλεί εντολή σε κάποιο άλλο που είχε τέτοιες δυνατότητες. Συστήματα με τέτοιες δυνατότητες πέραν του υψηλού κόστους , πολλές φορές είχαν και θέματα compatibility. Αξιοσημείωτο είναι ότι σε όλες τις περιπτώσεις το οποιοδήποτε action δεν θα προστάτευε το κυριότερο από τα συστήματα , το Firewall.

Με το δικό μας appliance είχαμε καταφέρει να λύσει τα προαναφερθέντα προβλήματα (κόστος και compatibility) με τη δημιουργία μιας συσκευής εξαιρετικά χαμηλού κόστους αλλά συνάμα υψηλών δυνατοτήτων. Ο τρόπος σύνδεσης που δημιουργήθηκε (μέσω ενός απλού καλέσματος σε ένα API) επιτρέπει στο οποιοδήποτε οργανισμό να αναπτύξει μηχανισμούς Active Defense από τα υπάρχοντα συστήματά του με μηδαμινό κόστος.

Συνοπτικά, έχουμε δημιουργήσει ένα appliance :
- εξαιρετικά χαμηλού κόστους
- που παρέχει ένα πολύ απλό και αποτελεσματικό τρόπο ενσωμάτωσις (integration) με άλλα περιφερειακά συστήματα
- που δεν μπορεί να δεχθεί επίθεση από το Internet παρόλο που φυσικά βρίσκεται σε ζώνη Internet .
- με δυνατότητες Dynamic Firewalling δηλ. να αφαιρεί κανόνες Firewall με βάση το χρόνο ενεργοποίησης τους και να αποφορτίζεται όσο αφορά τα resources του.
- που προστατεύει ακόμα και το ίδιο το Firewall.

5.2 Μελλοντική Έρευνα

Η συσκευή που δημιουργήσαμε αποτελεί τη βάση για ανάπτυξη ενός πραγματικά πολύ αποτελεσματικού μηχανισμού αντιμετώπισης κυβερνοεπιθέσεων σε επίπεδο ασφάλειας περιμέτρου ενός οργανισμού. Χαρακτηρίζεται από υψηλές δυνατότητες, ευκολία στη διασύνδεση και εκπληκτική απόδοση, χαρακτηριστικά που την κάνουν ελκυστική για περαιτέρω μελέτη και ανάπτυξη.

Η παρούσα μεταπτυχιακή διατριβή είχε στόχο την βασική ανάπτυξη της συσκευής μέχρι το σημείο της αρχικής υλοποίησης. Η συσκευή έχει σαφώς προοπτικές να αναπτυχθεί περισσότερο όσο αφορά τη λειτουργικότητά της, π.χ να αναβαθμιστεί και να εμπλουτιστεί το WebInterface της με διάφορα γραφικά, όπως γραφικές παραστάσεις απόδοσης hardware και δικτύου.

Επίσης, θα μπορούσαν να αναπτυχθούν plugins για διάφορα SIEMs όπως π.χ ElasticSearch, OpenSearch, Logstash κτλ με σκοπό την εύκολη διασύνδεση τους με το δικό μας σύστημα.

Επιπρόσθετα, λόγω της δυνατότητας σύνδεσης του συστήματός μας σε σημείο μπροστά από το Firewall και της δυνατότητας του αποκοπής χιλιάδων διευθύνσεων IPs χωρίς επηρεασμό της απόδοσης του θα μπορούσε να λειτουργήσει και σαν συσκευή αποκοπής DDoS επιθέσεων.

Παράρτημα Α
Πηγαίος Κώδικας

A.1 Source Code (API)

File /var/www/html/api/index.php

```
<?php

/*
 This is the API of l2dfw
 Created by Constantinos Efstathiou (efstathiou.c@gmail.com)

 1 Nov 2023

 Part of my master thesis @ Open University of Cyprus

 HOW IT WORKS
 ==================
   Accepts the following input parameters:
        - vcode  : authentication key
        - action : BLOCK/UNBLOCK/INIT
        - ip     : IP address


*/

// FUNCTIONS
// ----------

include('../dbconnect.php');
mysql_connect("localhost",$dbuser,$dbpass);
@mysql_select_db($dbname) or die("DB ERROR");
```

```
function maincmd ($ACTION,$IPADDR){
        $COMMAND="echo $ACTION $IPADDR |nc localhost 22023";
        $retval=shell_exec("$COMMAND"." 2>&1");



        return "$retval";
};

// INPUT PARAMETERS

$VCODE=$_POST['vcode'];
$ACTION=$_POST['action'];
$IPADDR=$_POST['ip'];

if ($VCODE == ''){
        echo "Sorry, we required you to authenticate";
        exit();
};
if ($ACTION == '' ){
        echo "Oops, no action defined";
        exit();
};

// Authenticate 1st
$query="select id from auth where vcode='$CODE'";
$result=mysql_query($query) or die ("ERROR".mysql_error());
$row=mysql_fetch_row($result);
$aid=$row[0];
if (is_numeric($aid) && $aid > 0){
        }else{
        echo "Authentication failed";
        exit();
};
```

```
// START OF MAIN
// ----------------
$RETVAL=maincmd("$ACTION","$IPADDR");

if ((preg_match('/OK/',$RETVAL))){
        if ($ACTION == "block"){
                //add to DB
                $query="insert into l2dfw_logs(id,sysid,logtype,datetime,src,action,loginfo)
values ('','$sysid','INFO',’DATE’,'$IPADDR','BLOCKED','Host [$IPADDR] blocked')";
                mysql_query($query) or die ("ERROR".mysql_error());
        };
        if ($ACTION == "unblock"){
                //add to DB
        $query="insert into l2dfw_logs(id,sysid,logtype,datetime,src,action,loginfo)
values ('','$sysid','INFO',CURRENT_TIMESTAMP,'$IPADDR','UNBLOCKED','Host
[$IPADDR] unblocked')";
                mysql_query($query) or die ("ERROR".mysql_error());
        };
};

echo "RETVAL=$RETVAL ACTION=$ACTION IP=$IPADDR";
```

A.2 Source Code (Dashboard)

File /var/www/html/index.php

```
<?php


session_start();

include('dbconnect.php');

include('version.php');


mysql_connect("localhost","$dbuser","$dbpass");

@mysql_select_db($dbname) or die ("FATAL1-ERROR".mysql_error());
```

```
//FUNCTIONS
//---------------
function redirect($page) {
        header('Location: ' . $page);
        exit();
};


if ($_SESSION['l2dfw_login'] != true) {
        redirect('login.php');
        exit();
};

/* Browser Info */
$userip=$_SERVER['REMOTE_ADDR'];



/* Real time TABLES */
$FWLOGS=<<<EOF
<script>
   $(document).ready(function(){
            $('#fw_logs').load('dashboard/fw_logs.php?randval='+ Math.random());
     });
</script>
<div align="center" style="width:100%;" id="fw_logs"><span style="font-
weight:bold;font-size:130%;"><br><br><div style="height:300px;"><br>Please
wait...</div></span></div>
```

```
EOF;


$APILOGS=<<<EOF
<script>
   $(document).ready(function(){
      var refreshId = setInterval(function() {
            $('#api_logs').load('dashboard/api_logs.php?randval='+ Math.random());
      }, 2000);
     });
</script>
<div align="center" style="width:100%;" id="api_logs"><span style="font-weight:bold;font-size:130%;"><br><br><div style="height:300px;"><br>Please wait...</div></span></div>


EOF;


$html=<<<EOF
<!doctype html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>


<title>L2DFW (Dynamic Firewall)</title>
<style>
  * {
     background-color:#222;
     color:#fff;
```

```
    font-family:"Arial";
    font-size:15px;
}
</style>
</head>
<table style="width:1100px;border-collapse:collapse;" cellpadding="10" cellspacing="5" border="0" align="center">
<tr>
        <th colspan="2" style="padding:2em;border-bottom:2px solid #444;">
                <img src="images/fw_logo.jpg" style="height:50px;border-radius:4px;"><br><span style="font-size:180%;"> Dynamic Firewall</span>
        </th>
</tr>
<tr>
<td style="vertical-align:top;width:50%;">$FWLOGS</td>
<td style="vertical-align:top;">$APILOGS</td>
</tr>
</table>
</html>
EOF;

echo $html;
?>
```

A.3 Source Code (Dashboard – API Logs)

File /var/www/html/dashboard/api_logs.php

```
<?php
include('../dbconnect.php');
mysql_connect("localhost",$dbuser,$dbpass);
@mysql_select_db($dbname) or die("DB ERROR");
```

```
// Generate api logs table
$query="select logtype,datetime,src,action,loginfo from l2dfw_logs order by id desc limit 12";
$result=mysql_query($query) or die ("ERROR".mysql_error());
while(list($logtype,$datetime,$src,$action,$loginfo)=mysql_fetch_array($result)){
        if ($action == 'BLOCKED'){
                $css_style='font-size:80%;background-color:#dd3333;color:#fff;border-radius:4px;padding:4px 3px;';
                }else{
                $css_style='font-size:80%;background-color:#008000;color:#fff;border-radius:4px;padding:4px 3px;';
        };
        $apilogs.=<<<EOF
        <tr>
                <td class="trow">$datetime</td>
                <td class="trow">$src</td>
                <td class="trow"><span style="$css_style">$action</span></td>
        </tr>
EOF;

};
$html=<<<EOF
<style>
  td.trow {
     border-bottom:1px solid #fff;
  };
</style>
<table cellpadding="10" style="width:100%;">
<tr>
```

```
        <td colspan="5" style="text-align:center;background-color:#555;border-radius:4px;color:#fff;">API Incoming Calls (last 12) </td>

</tr>

$apilogs

</table>

EOF;

echo $html;

?>
```

A.4 Source Code (Dashboard – FW Logs)

File /var/www/html/dashboard/fw_logs.php

```
<?php

function maincmd ($ACTION,$IPADDR){

        $COMMAND="echo $ACTION $IPADDR |nc -w 5 localhost 22023";

        $retval=shell_exec("$COMMAND"." 2>&1");

        return "$retval";

};

$fwperf=maincmd("system_perf","");

/*

$html=<<<EOF

<style>

  td.trow {

     border-bottom:1px solid #fff;

  };

</style>

<table cellpadding="10" style="width:100%;">

<tr>

        <td colspan="5" style="text-align:center;background-color:#555;border-radius:4px;color:#fff;">System Status</td>

</tr>
```

```
$fwperf
</table>
EOF;
echo $html;

?>
```

A.5 Source code (Service)

File /bin/l2dfw_webconnector.sh

```
#!/bin/bash
# Created by Constantinos Efstathiou
#
# This is part of my Master Thesis @ Open University of Cyprus
#
# Contact me : efstathiou.c@gmail.com
# ------------------------------------------------------------
tempfile=`tempfile 2>/dev/null` || tempfile=/tmp/test$$
trap "rm -f $tempfile" 0 1 2 5 15
logfile="/var/log/l2dfw/l2dfw.log";

if [ -f /etc/l2dfw/l2dfw.cfg ]
        then
        source /etc/l2dfw/l2dfw.cfg
fi

IPTABLES=/usr/sbin/iptables
IPSET=/usr/sbin/ipset
```

```
# Start of Main
# ------------

read action value1
# action : init/system_status/block/unblock/system_perf
# value1 : ip (if action=block or action=unblock)

# SYSTEM STATUS
# -----------------
if [ "$action" = "system_status" ]
        then
        #check for the iptables rule
        $IPTABLES -L -n |grep L2DFW >/dev/null
        if [ "$?" = "0" ]
                then
                $IPSET -L |grep L2DFW >/dev/null
                if [ "$?" = "0" ]
                        then
                        echo "OK";
                        exit 0;
                fi
        fi
        echo "ERROR";
        exit 0;
fi
# INITIALIZATION
# ------------------
if [ "$action" = "init" ]
```

```
        then
        $IPSET create L2DFW iphash timeout $TIMEOUT
        $IPTABLES -A FORWARD -m physdev --physdev-in
$INBOUND_INTERFACE -m set --match-set L2DFW src -j REJECT --reject-with
icmp-host-unreachable
        echo "INIT=DONE";
        exit 0;
fi
# [Un]Block an IP address
# -------------------------
if [ "$action" = "block" ]
        then
        $IPSET add L2DFW $value1 >/dev/null
        if [ "$?" = "0" ]
                then
                echo "OK";
                exit 0;
                else
                echo "FAILED";
                exit 0;
        fi
fi
if [ "$action" = "unblock" ]
      then
      $IPSET  del L2DFW $value1 >/dev/null
      if [ "$?" = "0" ]
            then
            echo "OK";
            exit 0;
```

```
            else
            echo "FAILED";
            exit 0;
        fi
fi
# System Performance Monitor
# -------------------------
# Will be displayed in the Web Interface
# Example Output :
#       <tr>
#           <td class="trow">$datetime</td>
#           <td class="trow">$src</td>
#           <td class="trow">$action</td>
#       </tr>
if [ "$action" = "system_perf" ]
        then
        output='';
        #Num of ban hosts
        NUM_OF_HOSTS=`$IPSET -L |grep -v Header |grep timeout |wc -l`
        output="
        <tr>
        <td class=\"trow\">Number of Blocked Hosts:</td>
        <td class=\"trow\" colspan=\"2\"><span style=\"background-
color:#dd3333;color:#fff;border-
radius:50%;padding:5px;\">$NUM_OF_HOSTS</span></td>
        </tr>";
        # timeout
        output+="
        <tr>
```

```
<td class=\"trow\">Blocked Timeout:</td>
<td class=\"trow\" colspan=\"2\">$TIMEOUT secs</td>
</tr>";
# CPUs
CPUS=`cat /proc/cpuinfo |grep processor |wc -l`
output+="
<tr>
<td class=\"trow\">CPU Cores:</td>
<td class=\"trow\" colspan=\"2\">$CPUS</td>
</tr>";
# Load Avgr
load_avg=`cat /proc/loadavg |awk {'print $1,$2,$3}'`
output+="
<tr>
<td class=\"trow\">Average Load:</td>
<td class=\"trow\" colspan=\"2\">$load_avg (1m, 5m, 15m)</td>
</tr>";
# Memory
totalmem=`cat /proc/meminfo |grep MemTotal: |awk {'print $2,$3'}`
total=`cat /proc/meminfo |grep MemTotal: |awk {'print $2'}`
availablemem=`cat /proc/meminfo |grep MemAvailable: |awk {'print $2,$3'}`
available=`cat /proc/meminfo |grep MemAvailable: |awk {'print $2'}`
mempercentage=$(echo "scale=2; ($available * 100) / $total" | bc)
result=$(echo "100 - $mempercentage" |bc)
memusage=`printf "%.0f%%\n" $result`
output+="
<tr>
<td class=\"trow\">Memory Usage:</td>
```

```
        <td class=\"trow\" colspan=\"2\">$memusage</td>
        </tr>";
        output+="
        <tr>
        <td class=\"trow\">Memory (Total/Available):</td>
        <td class=\"trow\" colspan=\"2\">$totalmem  |  $availablemem</td>
        </tr>";
        #Throughput
        # Initial values
        old_tx_bytes=$(ifconfig $INBOUND_INTERFACE | awk '/TX packets/{print
$5}')
        # Wait for some time (e.g., 10 seconds)
        sleep 4
        # New values
        new_tx_bytes=$(ifconfig $INBOUND_INTERFACE | awk '/TX packets/{print
$5}')
        # Calculate throughput in bits per second
        throughput_bps=$(8 * (new_tx_bytes - old_tx_bytes) / 10))
        throughput_mbps=$(echo "scale=0; $throughput_bps / 1000000" | bc)
      formatted_mbps=$(printf "%.6f" $throughput_mbps)
        #echo "Network Throughput (TX) is ${throughput_bps} bps or
throughput_mbps Mbps"
        output+="
        <tr>
        <td class=\"trow\">Throughput:</td>
        <td class=\"trow\" colspan=\"2\">${throughput_bps} bps ( $formatted_mbps
Mbps)</td>
        </tr>";
        #Accepted/Rejected Packects
```

```
        PKTSACCEPTED=`$IPTABLES -L -n -v |grep FORWARD |grep ACCEPT
|awk '{print $5,$6}'`

        BYTESACCEPTED=`$IPTABLES -L -n -v |grep FORWARD |grep ACCEPT
|awk '{print $7,$8}' |sed -e 's/)//g'`

        PKTSREJECTED=`$IPTABLES -L -n -v |grep REJECT |awk '{print $1}'`

        BYTESREJECTED=`$IPTABLES -L -n -v |grep REJECT |awk '{print $2}'`

        output+="

        <tr>

        <td class=\"trow\">Packets Accepted:</td>

        <td class=\"trow\" colspan=\"2\">$PKTSACCEPTED</td>

        </tr>

        <tr>

        <td class=\"trow\">Packets Rejected:</td>

        <td class=\"trow\" colspan=\"2\">$PKTSREJECTED packets,</td>

        </tr>

        <tr>

        <td class=\"trow\">Bytes Accepted:</td>

        <td class=\"trow\" colspan=\"2\">$BYTESACCEPTED</td>

        </tr>

        <tr>

        <td class=\"trow\">Bytes Rejected:</td>

        <td class=\"trow\" colspan=\"2\">$BYTESREJECTED bytes</td>

        </tr>

";

        echo "$output";

        exit 0;

fi

echo "Unknwon ERROR [$action $value1 $value2]";

exit 0;
```

Βιβλιογραφία

[1] Cornelius, Michaelis, Julius,Haslbeck, Maximilia, Carle, Georg. Diekmann, "Verified iptables firewall analysis". 2016

[2] Oktivasari, Prihatin, Zain, Ayu Rosyida, Agustin, Maria,Kurniawan, Asep, Murad, Fachroni arbi, Anshor, Muhammad fabian. Oktivasari, "Analysis of Effectiveness of Iptables on Web Server from Slowloris Attack". 2022

[3] Madani, A. Rezayi, S. Gharaee, H. Madani, "Log Management comprehensive architecture in Security Operation Center (SOC)," 2011.

[4] Man-qi Zhang, Yuan-long Jiang, Jian-hua Huang, "The design and implement of the centralized log gathering and analysis system," Zhangjiajie, China, 2012.

[5] J. Babbin, “Security Log Management : Identifying Patterns in the Chaos.” 2016

[6] H. Van Styn, "Advanced Firewall configurations with ipset". 2018