Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

Διπλωματική Εργασία

Ανίχνευση και Αντιμετώπιση Επιθέσεων σε Περιβάλλον Linux: Εφαρμογή των IDS/IPS εντός Εικονικών Τοπικών Δικτύων (VLAN)

Καραμήτρος Εμμανουήλ

Α.Μ. 1067417

Επιβλέπων Καθηγητής

Ηλίας Αριστείδης, ΕΔΙΠ

Πάτρα, Φεβρουάριος 2024

© Copyright συγγραφής Καραμήτρος Εμμανουήλ, 2024

Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών & Πληροφορικής του Πανεπιστημίου Πατρών δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, κ. Αριστείδη Ηλία, για την πολύτιμη καθοδήγηση, την ενθάρρυνση και την εμπιστοσύνη που μου έδειξε αναθέτοντάς μου το θέμα της παρούσας εργασίας. Η συμβολή του ήταν καθοριστική τόσο στην επιστημονική όσο και στην προσωπική μου ανάπτυξη. Επιπλέον, εκφράζω την ευγνωμοσύνη μου προς την οικογένειά μου, για την αμέριστη υποστήριξη, την αγάπη και την κατανόηση που μου προσέφεραν καθ' όλη τη διάρκεια των σπουδών μου. Η συμπαράστασή τους ήταν για εμένα ανεκτίμητη.

Καραμήτρος Εμμανουήλ

Περίληψη

Οι εικονικές τοπικές δικτυακές υποδομές (VLANs) αποτελούν αναπόσπαστο μέρος του σύγχρονου εταιρικού περιβάλλοντος. Παρέχουν απομόνωση από εξωτερικές απειλές, ενώ παράλληλα βελτιστοποιούν τη συνολική αποδοτικότητα του δικτύου. Ωστόσο, τα VLANs παραμένουν ευάλωτα σε ενδογενείς κινδύνους όπου οι επιτιθέμενοι εκμεταλλεύονται τα κενά ασφαλείας και εξαπλώνονται στους υπόλοιπους χρήστες. Καθίσταται έτσι απαραίτητη η υιοθέτηση εσωτερικών μηχανισμών ανίχνευσης και αντιμετώπισης επιθέσεων, που θα είναι σε θέση να προσαρμοστούν στις απαιτήσεις κάθε δικτυακού περιβάλλοντος. Η παρούσα διπλωματική εργασία επικεντρώνεται στον σχεδιασμό, την υλοποίηση και την εφαρμογή ενός μηχανισμού Ανίχνευσης και Αντιμετώπισης Επιθέσεων (IDS/IPS) σε εικονικά τοπικά δίκτυα (VLAN) που απαρτίζονται από συσκευές Linux. Το προτεινόμενο σύστημα βασίζεται στην αρχιτεκτονική Network - Based Intrusion Detection and Prevention (NIDPS), χρησιμοποιώντας υπογραφές (Signatures) για την ανίχνευση γνωστών τύπων επιθέσεων και δυναμική προσαρμογή για την καταστολή τους. Παράλληλα με το εργαλείο Snort που χρησιμοποιείται για τον εντοπισμό και την παραγωγή ειδοποιήσεων σε μια συνθήκη εισβολής, εφαρμόζεται η καινοτόμα τεχνολογία Dynamic Shield Deploy (DSD), η οποία επιτρέπει την άμεση και αυτοματοποιημένη ανταπόκριση σε επιθέσεις, μέσω δυναμικής θωράκισης των συστημάτων – στόχων. Η υλοποίηση βασίζεται στη χρήση Bash Scripts και περιλαμβάνει μια ευρεία γκάμα εργαλείων διαχείρισης και διαμόρφωσης δικτύου, όπως Iptables, Tcpdump, Scp κ.α., που συνεργάζονται για την κατηγοριοποίηση και την καταστολή των απειλών. Το σύστημα αντιμετωπίζει τις επιθέσεις ARP Spoofing, Denial of Service (DoS), ICMP Redirection, Nmap Scans, SSH Password Authentication Brute Force, και DHCP Spoofing, ενώ παράλληλα παρέχει ανθεκτικότητα σε τεχνικές παράκαμψης (Evasion Techniques) που συχνά χρησιμοποιούνται για την αποφυγή ανίχνευσης. Το εργαλείο λογισμικού που αναπτύχθηκε, δοκιμάστηκε χρησιμοποιώντας το VirtualBox για τη δημιουργία ενός περιβάλλοντος προσομοίωσης. Η εργαστηριακή εγκατάσταση (Lab Setup) περιλαμβάνει εικονικές μηχανές (VMs) που αναπαριστούν τους προστατευόμενους Hosts (χρήστες ή διακομιστές), τους επιτιθέμενους αλλά και την κεντρική συσκευή ασφάλειας. Αυτή η διαμόρφωση επέτρεψε τη ρεαλιστική απεικόνιση σεναρίων των παραπάνω επιθέσεων, καθώς και την αξιολόγηση των δυνατοτήτων του μηχανισμού, δεδομένων των περιορισμένων διαθέσιμων υπολογιστικών πόρων. Συνολικά, το προτεινόμενο σύστημα συμβάλλει στην υπάρχουσα βιβλιογραφία του τομέα ασφάλειας σε LAN, προτείνοντας μια επεκτάσιμη λύση, με προοπτικές εφαρμογής σε πραγματικά περιβάλλοντα μικρών ή μεσαίων δικτύων.

Λέξεις Κλειδιά

VLANs, Ασφάλεια δικτύων, Intrusion Detection and Prevention System, Dynamic Shield Deploy, Snort, Bash Scripts, Ενδογενείς Επιθέσεις, Signatures, Network – Based, VMs, Linux.

Abstract

Virtual Local Area Networks (VLANs) are an integral part of modern corporate environments. They provide isolation from external threats while optimizing the overall efficiency of the network. However, VLANs remain vulnerable to internal threats, where attackers exploit security gaps and infiltrate other systems within the LAN. This makes the adoption of internal mechanisms for detecting and mitigating attacks essential, as they must be adaptable to the requirements of any network environment. This thesis focuses on the design, implementation, and application of an Intrusion Detection and Prevention System (IDS/IPS) mechanism in VLANs composed of Linux-based devices. The proposed system is based on the Network-Based Intrusion Detection and Prevention (NIDPS) architecture, utilizing signatures to detect known types of attacks and dynamic adaptation for their mitigation. Alongside the Snort tool, which is used for intrusion detection and alert generation, the innovative Dynamic Shield Deploy (DSD) technology is applied. DSD allows for immediate and automated responses to attacks by dynamically shielding the target systems. The implementation is based on the use of Bash Scripts and incorporates a wide range of network management and configuration tools, such as Iptables, Tcpdump, Scp, and others, which work together to classify and suppress threats. The system addresses attacks such as ARP Spoofing, Denial of Service (DoS), ICMP Redirection, Nmap Scans, SSH Password Authentication Brute Force, and DHCP Spoofing, while also providing resilience to evasion techniques often used to bypass detection. The developed software tool was tested using VirtualBox to create a simulation environment. The lab setup included virtual machines (VMs) representing protected hosts (users or servers), attackers, and the central security device. This configuration enabled a realistic representation of the described attack scenarios and an evaluation of the mechanism's capabilities under limited computational resources. Overall, the proposed system contributes to the existing literature in the field of LAN security by offering a scalable solution with potential applicability in real-world small or medium-sized network environments.

Keywords

VLANs, Network Security, Intrusion Detection and Prevention System, Dynamic Shield Deploy, Snort, Bash Scripts, Internal Attacks, Signatures, Network-Based, VMs, Linux.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ 1
Σημασία του προβλήματος 1
Εφαρμογές και ανεπάρκειες της κυβερνοασφάλειας σε τυπικά VLAN 4
Σκοπός και διάρθρωση της εργασίας 8
1. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ 10
1.1 Ανάλυση των IDS/IPS 10
1.2 Η τεχνολογία Dynamic Shield Deploy 15
1.3 Διαγραμματική Πλαισίωση Σχεδιασμού 17
1.5 Περιγραφή Επιθέσεων 25
2. ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ 46
2.1 Δημιουργία Περιβάλλοντος Προσομοίωσης 46
2.2 Ανάπτυξη Λογισμικού 52
2.2.1 Μονάδα Ανίχνευσης 53
2.1.2 Μονάδα Αντιμετώπισης 60
3. ΛΕΙΤΟΥΡΓΙΑ ΚΑΙ ΑΠΟΤΕΛΕΣΜΑΤΑ 86
3.1 Nmap Scans 87
3.2 SSH Password Authentication Bruteforce Attack 92
3.3 ARP Spoofing 96
3.4 ICMP Redirection 100
3.5 Denial of Service 104
3.6 DHCP Spoofing 107
4. ΒΙΛΒΙΟΓΡΑΦΙΚΗ ΕΠΙΣΚΟΠΙΣΗ 110
4.1 Μελέτες Σχεδιασμού και Υλοποίησης Συστημάτων IDS/IPS 110
4.2 Μελέτες Μεθόδων Ανίχνευσης και Αντιμετώπισης Συγκεκριμένων Επιθέσεων 113
5. ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΟΠΤΙΚΕΣ 117
5.1 Περιορισμοί και Συμπεράσματα της Εργασίας 117
5.2 Μελλοντικές Επεκτάσεις 118
ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ 120
ΠΑΡΑΡΤΗΜΑ ΚΩΔΙΚΑ 125

ΛΙΣΤΑ ΕΙΚΟΝΩΝ

Εικόνα 1. Παγκόσμια Αγορά Κυβερνοασφάλειας [1] 1
Εικόνα 2. Αναφορές περιστατικών πλαστοπροσωπίας και απάτης [5] 2
Εικόνα 3. Παγκόσμια Αγορά Κυβερνοεγκλήματος [9] 3
Εικόνα 4. Διάρθρωση ενός τυπικού Εικονικού Δικτύου 5
Εικόνα 5. Εισβολή του επιτιθέμενου στο δίκτυο μέσω του μεταγωγέα 6
Εικόνα 6. Συσκευές που διατρέχουν το μεγαλύτερο κίνδυνο παραβιάσεων [14] 6
Εικόνα 7. Τύποι επιθέσεων που μελετήθηκαν 8
Εικόνα 8. Αρχιτεκτονική Host Based IDS (HIDS) 12
Εικόνα 9. Αρχιτεκτονική Network Based IDS (NIDS) 12
Εικόνα 10. Τοποθέτηση ενός τυπικού IPS 14
Εικόνα 11. Γενική Αρχιτεκτονική Δικτύου 17
Εικόνα 12. Raspberry Pi 4 [19] 19
Εικόνα 13. Cisco SF300-24PP Managed Switch [20] 19
Εικόνα 14. Block Diagram αναπαράστασης της λειτουργίας του συστήματος 20
Εικόνα 15. TCP Flags 26
Εικόνα 16. Full Scan [1/2]. 27
Εικόνα 17. Full Scan [2/2] 28
Εικόνα 18. Xmas Scan 29
Εικόνα 19. ARP - Request 30
Εικόνα 20. ARP - Reply 31
Εικόνα 21. ARP Spoof Full Duplex [1/2] 32
Εικόνα 22. ARP Spoof Full Duplex [2/2] 33
Εικόνα 23. Brute Force Attack 35
Εικόνα 24. DHCP Discover / Offer 37
Εικόνα 25. DHCP Request / Acknowledgement 37
Εικόνα 26. Επισκόπηση DHCP Spoofing 39
Εικόνα 27. Πλεονασματική Διαδρομή Δεδομένων 41
Εικόνα 28. Αποστολή Πακέτου ICMP Redirect 42
Εικόνα 29. Νέα Διαδρομή Πακέτων 43
Εικόνα 30. Επισκόπηση του Host Machine 47

Εικόνα 31. Επισκόπηση του IDS/IPS Virtual Machine 47
Εικόνα 32. Επισκόπηση του Attacker Virtual Machine 47
Εικόνα 33. Επισκόπηση Victim1 Virtual Machine 48
Εικόνα 34. NAT Network. 49
Εικόνα 35. Internal Network 49
Εικόνα 36. Lab Setup 50
Εικόνα 37. Σύνδεση SSH στο Victim1 VM 51
Εικόνα 38. Ρύθμιση Μεταβλητών Δικτύου στη Mainscript.sh 52
Εικόνα 39. Εκτέλεση του Script 52
Εικόνα 40. Παραμετροποιήσεις αρχείου Snort.conf 53
Εικόνα 41. /var/log/snort/alert - Snort Alert Logs 59
Εικόνα 42. Εκκίνηση Snort στο Mainscript.sh 59
Εικόνα 43. Attack Verification & Classification 60
Εικόνα 44. Tcpdump Output σε Xmas Scan 61
Εικόνα 45. Process_flag() Function [1/2] 62
Εικόνα 46. Process_flag() Function [2/2] 62
Εικόνα 47. Mac2ip_convertor() & Ip_unspoof() Functions 63
Εικόνα 48. Network_scan() Function 64
Εικόνα 49. Destination_ipcheck() Function 65
Εικόνα 50. Nmaphandler.sh [1/2] 66
Εικόνα 51. Nmaphandler.sh [2/2] 66
Εικόνα 52. Nmap.sh Shield [1/2] 68
Εικόνα 53. Nmap.sh Shield [2/2] 68
Εικόνα 54. Bruteforcehandler.sh 69
Εικόνα 55. Bruteforce.sh Shield [1/2] 71
Εικόνα 56. Bruteforce.sh Shield [2/2] 71
Εικόνα 57. Arpspoofhandler.sh 72
Εικόνα 58. Arpspoof.sh Shield [1/3] 74
Εικόνα 59 Arpspoof.sh Shield [2/3] 74
Εικόνα 60. Arpspoof.sh. Shield [3/3] 75
Εικόνα 61. Redirectionhandler.sh 76
Εικόνα 62. Icmpredirect.sh Shield [1/2] 78
Εικόνα 63. Icmpredirect.sh Shield [2/2] 78
Εικόνα 64. Doshandler.sh [1/3] 79

Εικόνα 65. Doshandler.sh [2/3] 79
Εικόνα 66. Doshandler.sh [3/3] 80
Εικόνα 67. Ddos.sh Shield 81
Εικόνα 68. Dhcpspoofhandler.sh [1/2] 84
Εικόνα 69. Dhcpspoofhandler.sh [2/2] 84
Εικόνα 70. Dhcpspoofing.sh Shield [1/2] 85
Εικόνα 71. Dhcpspoofing.sh Shield [2/2] 85
Εικόνα 72. Αρχικοποίηση Συστήματος IDS/IPS 86
Εικόνα 73. Σύνοψη σαρώσεων Nmap 87
Εικόνα 74. Καταγραφή Wireshark για Xmas Scan 88
Εικόνα 75. Καταγραφή Wireshark για Full Scan 88
Εικόνα 76. Generated Report για Stealth Scan 90
Εικόνα 77. Iptables Ruleset του Victim1 [1/2] 90
Εικόνα 78. Generated Report για Null Scan χρησιμοποιώντας Decoy IP Addresses 91
Εικόνα 79. Iptables Ruleset του Victim1 [2/2] 91
Εικόνα 80. Καταγραφή Wireshark για SSH Password Auth. Brute Force Attack 93
Εικόνα 81. Hydra Output 93
Εικόνα 82. SSHD Config File & Iptables Ruleset του Victim 94
Εικόνα 83. Generated Report για Bruteforce Attack (Σενάριο 1) 95
Εικόνα 84. Generated Report για Bruteforce Attack (Σενάριο 2) 95
Εικόνα 85. Ettercap ARP Spoofing GUI 97
Εικόνα 86. Καταγραφή Wireshark για ARP Spoofing και υποκλοπή κυκλοφορίας 97
Εικόνα 87. Generated Report για Full Duplex ARP Spoofing Attack 98
Εικόνα 88. Σύνοψη Βασικών Ρυθμίσεων στο Victim1 99
Εικόνα 89. Καταγραφή Wireshark για ICMP Redirect For Host 101
Εικόνα 90. Σύνοψη Επίθεσης ICMP Redirect 102
Εικόνα 91. Generated Report για ICMP Redirection For Host 103
Εικόνα 92. Αποτελέσματα Ενεργειών του IDS/IPS 103
Εικόνα 93. Σύνοψη πλημμυρών Hping3 105
Εικόνα 94. Στατιστική Απεικόνιση Λειτουργίας του IDS/IPS σε DoS 105
Εικόνα 95. Generated Report για TCP - SYN Flood 106
Εικόνα 96. Generated Report για UDP Flood με Spoofed IP 106
Εικόνα 97. Ettercap DHCP Configurations 107
Εικόνα 98. Καταγραφή Wireshark Διαδικασίας DHCP 108

Εικόνα 99. Ρυθμίσεις Victim1 ύστερα από DHCP Spoofing Attack.....108
Εικόνα 100. Generated Report για DHCP Spoofing.....109
Εικόνα 101. Ρυθμίσεις Victim1 ύστερα από την παρέμβαση IDS/IPS 109
Εικόνα 102. Τοπολογία Scaled Δικτύων (Χωρισμένη για Λόγους Ευκρίνειας) 119

ΠΙΝΑΚΑΣ ΕΠΕΞΗΓΗΣΗΣ ΑΚΡΩΝΥΜΙΩΝ

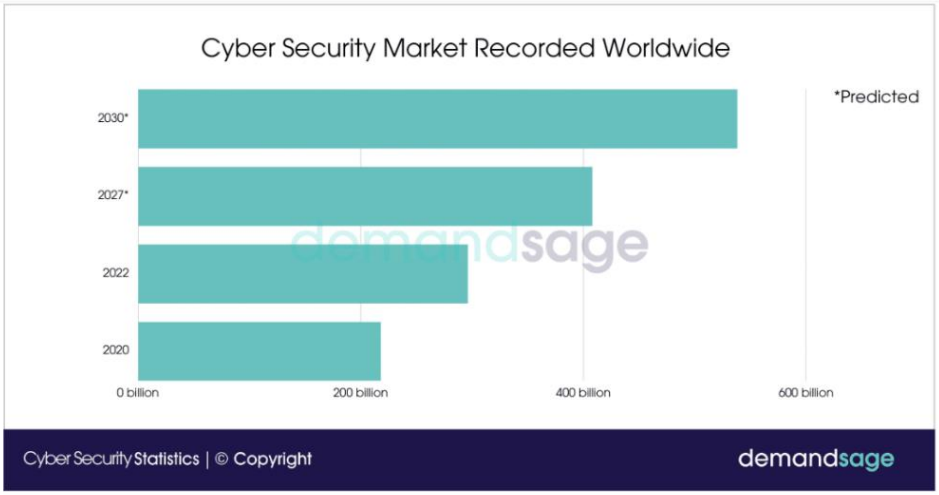
Συμβολισμός	Σημασία	Επεξήγηση
IT	Information Technology	Επαγγελματικοί τομείς που συσχετίζονται με υπολογιστικά πληροφοριακά συστήματα
LAN	Local Area Network	Σύνολο συνδεδεμένων υπολογιστικών συστημάτων που εκτείνονται σε περιορισμένη γεωγραφική περιοχή
VLAN	Virtual - LAN	Λογικά διαχωρισμένα δίκτυα συστημάτων εντός του ίδιου φυσικού LAN
IDS	Intrusion Detection System	Σύστημα παρακολούθησης κυκλοφορίας στο δίκτυο για παραβιάσεις στην πολιτική ασφάλειας
IPS	Intrusion Prevention System	Σύστημα πρόληψης και αναχαίτισης εντοπισμένων απειλών
OSI	Open System Interconnection	Μοντέλο υποδιαίρεσης των λειτουργιών ενός τηλεπικοινωνιακού δικτύου σε μια “κατακόρυφη” στοίβα
TCP	Transmission Control Protocol	Πρωτόκολλο αξιόπιστης μεταφοράς δεδομένων μεταξύ του επιπέδου δικτύου και του επιπέδου εφαρμογής
DSD	Dynamic Shield Deploy	Τεχνολογία αυτόματης αντιμετώπισης επιθέσεων βασισμένη σε αρχιτεκτονικές εκτεταμένου δικτύου
Nmap	Network Mapper	Εργαλείο ανοιχτού κώδικα για ανακάλυψη δικτύων και έλεγχο ασφάλειας
ARP	Address Resolution Protocol	Πρωτόκολλο αντιστοίχισης διευθύνσεων IP (Layer 3 Address) στις αντίστοιχες MAC (Layer 2 Address)

SSH	Secure Shell	Πρωτόκολλο παροχής ενός ασφαλούς τρόπου πρόσβασης και διαχείρισης απομακρυσμένων συστημάτων σε μη ασφαλή δίκτυα
DHCP	Dynamic Host Configuration Protocol	Πρωτόκολλο αυτόματης ανάθεσης διευθύνσεων IP, Default Gateway, DNS καθώς και Subnet Mask.
UDP	User Datagram Protocol	Πρωτόκολλο μεταφοράς δεδομένων όπως το TPC, βασιζόμενο στην ταχύτητα και όχι στην αξιοπιστία μετάδοσης.
ICMP	Internet Control Message Protocol	Πρωτόκολλο αποστολής μηνυμάτων σφάλματος, λειτουργικών πληροφοριών και διαγνωστικών δεδομένων κατά την επικοινωνία μεταξύ συσκευών
NAT	Network Address Translation	Μέθοδος αντιστοίχισης συνόλου IP διευθύνσεων σε ένα άλλο, τροποποιώντας τις πληροφορίες διεύθυνσης IP στην κεφαλίδα των πακέτων, καθώς διέρχονται μέσω ενός δρομολογητή κίνησης.

ΕΙΣΑΓΩΓΗ

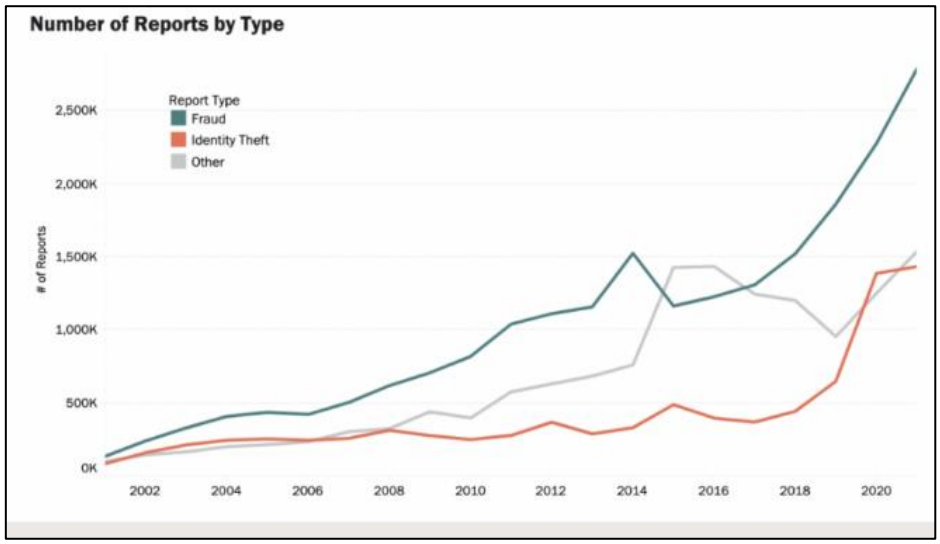
Σημασία του προβλήματος

Οι τεχνολογικές εξελίξεις στο οικοσύστημα της κυβερνοασφάλειας τα τελευταία χρόνια είναι εντυπωσιακές. Οργανισμοί και εταιρίες ανεξαρτήτως οικονομικού υποβάθρου έχουν αρχίσει να επενδύουν εντατικά στην ανάπτυξη τεχνολογιών που στοχεύουν στην προστασία των δεδομένων και των υποδομών τους από τις επικείμενες απειλές. Συγκεκριμένα, μικρομεσαίες επιχειρήσεις δαπανούν το 10% του προϋπολογισμού που προορίζεται να διανεμηθεί στον τομέα του IT (Information Technology), αποκλειστικά στην εγκαθίδρυση συστημάτων κυβερνοασφάλειας. Παράλληλα, το 55% των εν λόγω εταιριών ξοδεύουν συνεχώς αυξανόμενα χρηματικά ποσά σε ιδιωτικές ασφάλειες που σχετίζονται με τον κυβερνοχώρο (28% αύξηση των ασφαλειών από το 2023) [1]. Ενώ οι κυβερνοεπιθέσεις σε συνηθισμένες εταιρείες και επιχειρήσεις μπορούν, το πολύ να οδηγήσουν σε οικονομική ζημία ή στην αποκάλυψη δεδομένων σχετικά με την εσωτερική τους λειτουργία, οι επιθέσεις σε κρίσιμες κρατικές υποδομές (ενεργειακά ή δίκτυα τηλεπικοινωνιών) [2] θα μπορούσαν δυνητικά να εξελιχθούν σε απειλή για την εθνική ασφάλεια. Γίνεται λοιπόν λόγος για την άνθιση μιας αγοράς που χαίρει ασύλληπτου περιθώριου κέρδους. Η παρακάτω εικόνα απεικονίζει στατιστικά την πορεία της εν λόγω αγοράς στο κοντινό μέλλον, σε παγκόσμιο βεληνεκές.



Εικόνα 1. Παγκόσμια Αγορά Κυβερνοασφάλειας [1]

Θεμελιώδες αίτιο του παραπάνω φρενήρους ρυθμού ανάπτυξης της αγοράς δεν θα μπορούσε να είναι άλλο από τον βαθύτατο και πολυδιάστατο αντίκτυπο που έχουν οι κυβερνοεπιθέσεις στις σύγχρονες κοινωνίες. Εκτός από την προαναφερθείσα απειλή στη εθνική ασφάλεια και τις αναμενόμενες γεωπολιτικές εντάσεις, οι κυβερνοεπιθέσεις διαταράσσουν τις προσωπικές ζωές των ατόμων και κατακερματίζουν τον κοινωνικό ιστό. Η παραβίαση προσωπικών δεδομένων ενός ατόμου (Data Breach) συχνά οδηγεί σε εξάρσεις φαινομένων όπως κατάθλιψη, άγχος, κρίσεις πανικού, μετατραυματικό στρες, ακόμα και αυτοκτονίες [3]. Είναι επίσης συχνές οι περιπτώσεις όπου τα εν λόγω θύματα εξακολουθούν να φοβούνται ενδεχόμενα περιστατικά δεύτερης επίθεσης εις βάρος τους, αρκετό καιρό ύστερα από την πρωταρχική εκμετάλλευση των στοιχείων τους. Στην παρακάτω εικόνα παρατίθενται στοιχεία που αφορούν (τα σταθερά αυξανόμενα) επιβεβαιωμένα περιστατικά πλαστοπροσωπίας και απάτης (κυρίως οικονομικής φύσεως) [4]. Οι παράνομες αυτές πράξεις συσχετίζονται άμεσα με την παραβίαση δεδομένων, καθώς τα αποτελέσματα της συγκεκριμένης κυβερνοεπίθεσης αξιοποιούνται και διευκολύνουν διάπραξή τους.



Εικόνα 2. Αναφορές περιστατικών πλαστοπροσωπίας και απάτης [5]

Όσον αφορά τις συνέπειες των κυβερνοεπιθέσεων στη λειτουργία των σύγχρονων κοινωνιών, παρατηρείται έντονα η εξάπλωση παραπλανητικών πληροφοριών ως αποτέλεσμα παραβίασης δημοφιλών κοινωνικών πλατοφορμών. Η παραπληροφόρηση μπορεί να κλιμακωθεί ταχύτατα, ιδίως κατά τη διάρκεια κρίσιμων γεγονότων όπως εκλογές, κρίσεις δημόσιας υγείας ή φυσικές καταστροφές, όπου η εμπιστοσύνη σε αξιόπιστες πληροφορίες καθίσταται ζωτικής σημασίας. Σύμφωνα με την Ευρωπαϊκή Ένωση [6], η διάδοση ψευδών πληροφοριών σχετικά με την κλιματική αλλαγή, τις ασθένειες, τα εμβόλια καθώς και

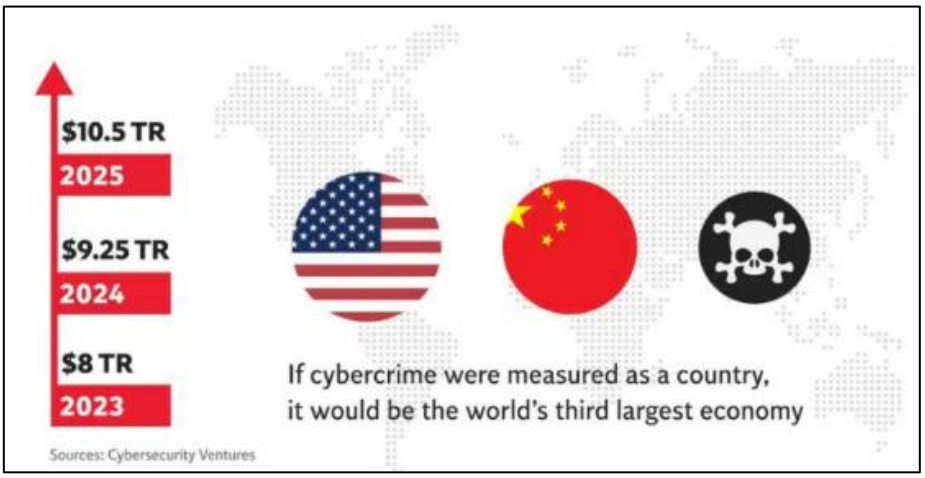
ακραίες απόψεις κατά συγκεκριμένων πληθυσμιακών τμημάτων έχουν επιφέρει ανυπέρβλητη φθορά στις επιμέρους κοινωνίες.

Ωστόσο, το κυριότερο αίτιο της ταχύτατης ανάπτυξης της αγοράς κυβερνοασφάλειας είναι οι δριμύτατες οικονομικές επιπτώσεις του κυβερνοεγκλήματος τόσο στις υφιστάμενες επιχειρήσεις, όσο και στην ευρύτερη οικονομία [7]. Οι εν λόγω επιπτώσεις διακρίνονται σε:

- Άμεσο κόστος. Οικονομικές απώλειες που προκύπτουν από την εκάστοτε επίθεση (κλοπές, απάτες, επιθέσεις Ransomware [8] κ.α.) καθώς και κόστη αντιμετώπισης των εν λόγω περιστατικών στον κυβερνοχώρο (επιχειρήσεις αποκατάστασης της ζημιάς – επένδυση σε ισχυρότερες άμυνες).

- Έμμεσο κόστος. Μείωση των μελλοντικών εσόδων, λόγω κρίσης εμπιστοσύνης στην ασφάλεια που προσφέρει η εταιρία στους εν δυνάμει πελάτες.

Για τις μικρές και τις μεσαίες επιχειρήσεις, μόλις μια καίρια παραβίαση της ασφάλειας σε ζωτικά συστήματά τους είναι αρκετή, προκειμένου να τις οδηγήσει σε οικονομικό αφανισμό. Όσον αφορά τις κολοσσιαίες εταιρίες – οργανισμούς, είναι φανερό ότι το έγκλημα στον κυβερνοχώρο (μέσω επιθέσεων γιγαντιαίου βεληνεκούς) μπορεί να οδηγήσει σε κλονισμό της ευρύτερης οικονομικής σταθερότητας. Συγκεκριμένα η παράνομη αγορά του κυβερνοεγκλήματος παρουσιάζει εντυπωσιακότερη ανάπτυξη σε σχέση με την αντίστοιχη της κυβερνοασφάλειας, καθώς τα τελευταία χρόνια κυμαίνεται στην κλίμακα των τρισεκατομμυρίων. Στην παρακάτω εικόνα παρουσιάζεται το ποσοτικοποιημένο οικονομικό εύρους που προκύπτει ετησίως από παράνομη δραστηριότητα στον κυβερνοχώρο, σε σύγκριση με τις δύο ισχυρότερες οικονομίες παγκοσμίως.



Εικόνα 3. Παγκόσμια Αγορά Κυβερνοεγκλήματος [9]

Εφαρμογές και ανεπάρκειες της κυβερνοασφάλειας σε τυπικά VLAN

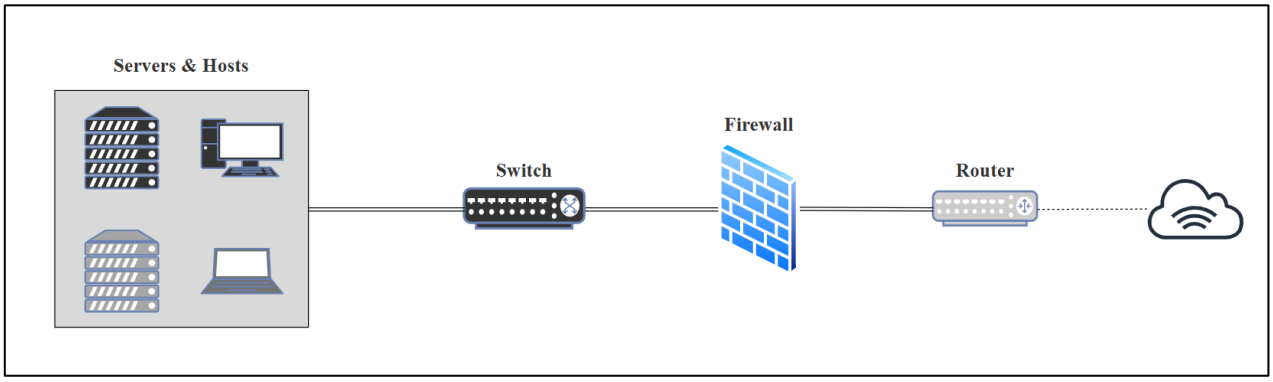
Μια συνηθισμένη εγκατάσταση εσωτερικού δικτύου στα πλαίσια μικρομεσαίων εταιριών, συντίθεται από μια πληθώρα Εικονικών Τοπικών Δικτύων (VLANs - Virtual Local Area Networks). Τα εν λόγω αποτελούν λογικά δίκτυα υπολογιστικών συστημάτων εντός του ίδιου φυσικού LAN (Local Area Network), τα οποία είναι εντελώς διαχωρισμένα μεταξύ τους. Σε σύγκριση με αντίστοιχες αρχιτεκτονικές παραδοσιακών, φυσικών Τοπικών Δικτύων (LANs) προσφέρουν δύο σημαντικά πλεονεκτήματα: ενισχυμένη ασφάλεια και βελτιωμένες επιδώσεις δικτύου [10]. Η αναβάθμιση της ασφάλειας σχετίζεται με την ίδια τη φύση του VLAN, καθώς απομονώνοντας τα επιμέρους δίκτυα μεταξύ τους, δίνεται η δυνατότητα διαβάθμισής τους με βάση την επικινδυνότητά τους καθώς και εφαρμογής αντίστοιχης Πολιτικής Ασφάλειας (Cybersecurity Policy). Παράλληλα, με τη χρήση των VLAN - αντί ενός ενιαίου LAN - μειώνεται σημαντικά η κυκλοφορία μεταδόσεων (broadcast traffic), καθώς τα συστήματα εντός του εικονικού δικτύου δεν λαμβάνουν αντίστοιχα broadcast μηνύματα από άλλα VLANs, όπως θα γινόταν στην περίπτωση του ενιαίου LAN όπου ο παραπάνω τύπος μηνυμάτων θα ήταν ορατός σε όλο το εσωτερικό δίκτυο της εταιρίας. Αποτέλεσμα του τελευταίου αποτελεί η αποδοτικότερη διαχείριση του υφιστάμενου εύρους ζώνης (bandwidth), και η αναβάθμιση της συνολικής επίδοσης του δικτύου. Ένα VLAN μπορεί να αποτελείται από:

1. Local Servers (Τοπικούς Εξυπηρετητές). Συστήματα που αναλαμβάνουν την παροχή διαφόρων υπηρεσιών στο αντίστοιχο εταιρικό τμήμα.
2. Hosts (Client Computers). Σταθμοί εργασίας (workstations) ή Laptops για τους υπαλλήλους της εταιρίας, όπου επικοινωνούν μεταξύ τους, καθώς και με τους αντίστοιχους Servers.

3. Switch (Μεταγωγέας). Είναι υπεύθυνος για τη διασύνδεση των υπολογιστικών συστημάτων εντός του VLAN με βάση τη διεύθυνση MAC της κάθε συσκευής. Σε ορισμένες περιπτώσεις μπορεί εξυπηρετεί παράλληλες λειτουργίες δρομολογητή προκειμένου να διασφαλιστεί η δυνατότητα επικοινωνίας μεταξύ των VLANs (Layer 3 Switch [11]). Προφανώς, η ύπαρξη VLAN δικτύων προϋποθέτει managed – τύπου μεταγωγείς.

4. Router (Δρομολογητής). Συνδέει το εσωτερικό εικονικό δίκτυο με αντίστοιχα εξωτερικά δίκτυα (εν προκειμένω το διαδίκτυο).

5. Firewall (Τείχος προστασίας). Σύστημα ασφαλείας (συσκευή) το οποίο ρυθμίζει την εισερχόμενη και εξερχόμενη κυκλοφορία του εικονικού δικτύου με βάση προκαθορισμένους κανόνες ασφαλείας.

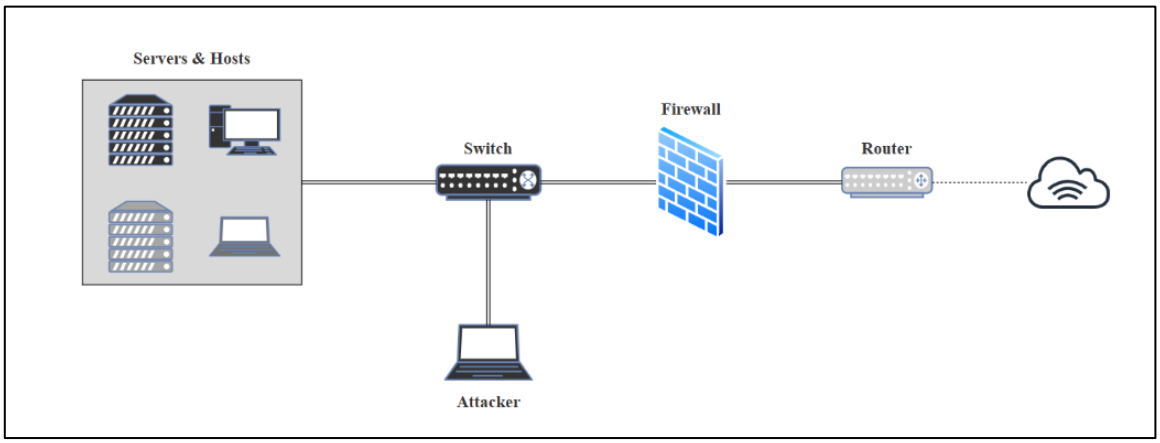


Εικόνα 4. Διάρθρωση ενός τυπικού Εικονικού Δικτύου

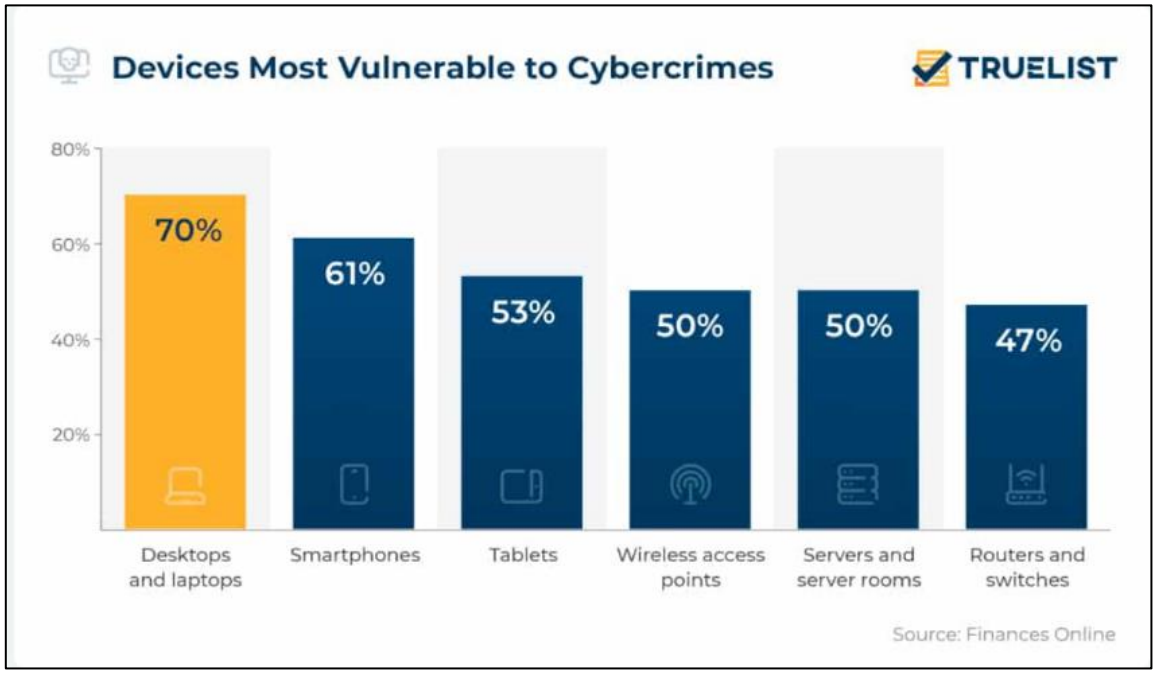
Στις σύγχρονες επιχειρήσεις τα τείχη προστασίας κατέχουν κρίσιμο ρόλο στη διαφύλαξη της ασφάλειας σε όλη την εμβέλεια του φυσικού και εικονικού δικτύου. Δεν χρησιμεύουν μόνο ως φράγμα έναντι εξωτερικών απειλών, αλλά παράλληλα περιορίζουν την εξάπλωση (Lateral Movement) μιας επικείμενης επίθεσης, στα υπόλοιπα εικονικά δίκτυα του οργανισμού. Με την εγκατάσταση λοιπόν των τειχών προστασίας, οι επιχειρήσεις μπορούν να ελέγχουν αποδοτικά τις εξωτερικές απειλές, να περιορίζουν την πρόσβαση μη εξουσιοδοτημένων ατόμων όπου χρειάζεται και να εφαρμόζουν κυμαινόμενης αυστηρότητας πρωτόκολλα ασφάλειας, προστατεύοντας έτσι κρίσιμους υλικούς πόρους και ευαίσθητα δεδομένα. Η υπάρχουσα βιβλιογραφία υποδεικνύει σημαντική πρόοδο στην τεχνολογία των firewall, ιδιαίτερα την τελευταία δεκαετία [12, 13].

Δεδομένου ότι τα τείχη προστασίας τοποθετούνται συνήθως στην περίμετρο του δικτύου (και πριν το Router - βλ. Σχήμα 1), η πρωταρχική τους λειτουργία είναι το φιλτράρισμα της κυκλοφορίας από εξωτερικά δίκτυα (κυρίως το Internet) προς το εσωτερικό, και το αντίστροφο. Το γεγονός αυτό όμως συχνά καθιστά το δίκτυο ευάλωτο σε ενδογενείς απειλές, καθώς κάποιος κακόβουλος φορέας μπορεί συνδέσει τη συσκευή του στο εκάστοτε εικονικό δίκτυο (μέσω του εταιρικού μεταγωγέα). Σε αυτή την περίπτωση, μπορεί να επιχειρήσει ανενόχλητα lateral movement στις εσωτερικές συσκευές του VLAN, καθώς η εμβέλεια των φίλτρων του τείχους προστασίας δεν είναι αρκετή προκειμένου να εντοπίσει την παραβίαση ασφάλειας. Το παραπάνω απεικονίζεται διαγραμματικά στο Σχήμα 2, ενώ η Εικόνα 3 αναδεικνύει την

εφικτότατα του παραπάνω σεναρίου, καθώς οι μεταγωγείς διατρέχουν ιδιαίτερα υψηλό κίνδυνο όσον αφορά την τρωτότητα σε παραβιάσεις ασφαλείας.



Εικόνα 5. Εισβολή του επιτιθέμενου στο δίκτυο μέσω του μεταγωγέα



Εικόνα 6. Συσκευές που διατρέχουν το μεγαλύτερο κίνδυνο παραβιάσεων [14]

Το lateral movement εντός του VLAN μπορεί να επιτευχθεί όχι μόνο μέσω της φυσικής σύνδεσης του επιτιθέμενου στον εταιρικό μεταγωγέα, αλλά και από περιπτώσεις όπου οι συσκευές ορισμένων hosts (υπαλλήλων) του δικτύου προσβάλλονται από κατάλληλο κακόβουλο λογισμικού (malware). Σε αυτή την περίπτωση, ο επιτιθέμενος έχει τη δυνατότητα - μεταξύ άλλων - να διαχειριστεί το προσβαλλόμενο σύστημα με σκοπό να εξαπολύσει επιθέσεις κατά ευαίσθητων host του δικτύου και να εξαπλωθεί σε αυτό. Η ταχύρρυθμη ανάπτυξη νέων τύπων malware υπερτερεί της παραγωγής ανανεωμένων firewall όπου θα τα αναγνωρίζουν έγκαιρα και θα τα αναχαιτίζουν κατά την εισαγωγή τους στο προστατευόμενο δίκτυο. Κυρίως λόγω της ευέλικτης φύσης των malware, οι επιτιθέμενοι εκμεταλλεύονται το στατικό μοντέλο των firewall. Δημιουργούν προηγμένα λογισμικά τα οποία περνούν απαρατήρητα από τείχη προστασίας που δεν έχουν συντηρηθεί, παραμετροποιηθεί και ενημερωθεί επαρκώς, περιμένοντας κάποιο ανθρώπινο λάθος προκειμένου να ενεργοποιηθούν (συνήθως με το πάτημα κάποιο link). Επιβεβαίωση των παραπάνω αποτελεί το γεγονός ότι το ανθρώπινο λάθος είναι η κυριότερη αιτία παραβιάσεων της κυβερνοασφάλειας παγκοσμίως [14].

Αναλύοντας τα δύο παραπάνω σενάρια, εξάγονται τα εξής συμπεράσματα αναφορικά με την εφαρμογή συστημάτων κυβερνοασφάλειας σε ένα συνηθισμένου εύρους VLAN:

- Αδυναμία διατήρησης της ασφάλειας σε δίκτυα που βασίζονται αποκλειστικά σε περιμετρικές άμυνες (firewall) λόγω ενδογενών απειλών.

- Ανάγκη εφαρμογής ενός συμπληρωματικού στρώματος ασφαλείας (Layer of Security) στο δίκτυο μέσω συστημάτων που ανιχνεύουν επικείμενες απειλές, τις καταστέλλουν και προφυλάσσουν τους προστατευόμενους hosts και servers από αντίστοιχη μελλοντική επίθεση.

Σκοπός και διάρθρωση της εργασίας

Σκοπός λοιπόν της παρούσας εργασίας είναι η ανάπτυξη ενός κεντρικοποιημένου Intrusion Detection System / Intrusion Prevention System (IDS/IPS) για άμεση και αυτόματη δράση σε περιπτώσεις συγκεκριμένων επιθέσεων εντός του δικτύου, σε πραγματικό χρόνο. Το εν λόγω σύστημα έχει εγκατασταθεί και παραμετροποιηθεί κατάλληλα, προκειμένου να είναι σε θέση να επιβλέπει την κυκλοφορία στο δίκτυο, να εντοπίζει τις ενδογενείς απειλές και τέλος να ενεργεί με το βέλτιστο δυνατό τρόπο πρόληψης και αναχαίτισης τους. Σημαντικό χαρακτηριστικό του λογισμικού που αναπτύχθηκε αποτελεί η δυνατότητα ευελιξίας και επεκτασιμότητας (scalability) που προσφέρει το σύστημα, όχι μονό όσον αφορά την εφαρμογή του σε δίκτυα μεγαλύτερης εμβέλειας από αυτό που μελετάται, αλλά και για επιθέσεις από ευρύτερο σύνολο κακόβουλων συσκευών. Η υλοποίηση του συστήματος ασφαλείας και η διαδικασία ελέγχου της σωστής και εύρυθμης λειτουργίας του, σε αντίστοιχο εικονικό δίκτυο προσομοίωσης, πραγματοποιήθηκε σε περιβάλλον Linux. Το ίδιο συνέβη και για τις προσομοιώσεις των επιθέσεων σε “θύματα” hosts και servers (διαφόρων υπηρεσιών) από “κακόβουλη” συσκευή, ενώ το πείραμα στο σύνολό του πραγματοποιήθηκε μέσω εικονικών μηχανών (virtual machines) αντίστοιχου λογισμικού εικονικοποίησης (virtualization software). Οι τύποι επιθέσεων που ανιχνεύθηκαν και καταπολεμήθηκαν σε ικανοποιητικό βαθμό από το IDS/IPS, παρουσιάζονται παρακάτω και αναλύονται ενδελεχώς στη συνέχεια της παρούσας εργασίας. Σημειώνεται ότι κατά τα σενάρια εξαπόλυσης των επιθέσεων συμπεριλαμβάνεται και η χρήση γνωστών τεχνικών διαφυγής (evasion techniques) οι οποίες συμβάλλουν στην απόκρυψη της ταυτότητας του επιτιθέμενου και την παράκαμψη του IDS.

ARP SPOOFING
DHCP SPOOFING
DENIAL OF SERVICE - DOS
ICMP REDIRECTION ATTACK
NMAP SCAN(s)
SSH PASSWORD AUTHENTICATION BRUTE FORCE ATTACK

Εικόνα 7. Τύποι επιθέσεων που μελετήθηκαν

Στο πρώτο Κεφάλαιο παρατίθεται η επεξήγηση του απαραίτητου θεωρητικού υποβάθρου. Ο αναγνώστης στα επιμέρους υποκεφάλαια, εισάγεται στις έννοιες των Intrusion Detection - Prevention Systems και τους διάφορους τύπους αυτών, όπως διακρίνονται από την υφιστάμενη βιβλιογραφία. Ακολουθεί εκτεταμένη θεωρητική μελέτη της δομής και των λειτουργιών του συστήματος που αναπτύχθηκε στην παρούσα εργασία, καθώς και ανάλυση όλων των παραπάνω τύπων επιθέσεων Στο δεύτερο Κεφάλαιο ο αναγνώστης μεταβαίνει από τη σχεδίαση του μοντέλου στην υλοποίηση του λογισμικού. Εξηγείται αρχικά το περιβάλλον και οι συνθήκες πειράματος υπό τις οποίες θα λειτουργήσει το εν λόγω σύστημα, ενώ στη συνέχεια παρουσιάζονται τα καίρια δομικά στοιχεία του κώδικα, όπως προέκυψαν από τη μελέτη του θεωρητικού υποβάθρου. Δίνεται έμφαση τόσο στις προγραμματιστικές λεπτομέρειες όσο και στη συνδεσιμότητα των επιμέρους υποσυστημάτων, προσφέροντας σαφή εικόνα για την πρακτική υλοποίηση. Το τρίτο Κεφάλαιο περιέχει την προσομοίωση ποικίλων σεναρίων επιθέσεων στο πειραματικό δίκτυο, καθώς και τις ενέργειες του IDS/IPS ως αποτέλεσμα των παραπάνω. Το σύστημα αξιολογείται βάσει αποτελεσματικότητας της ανίχνευσης, χρόνου απόκρισης και τις και των γενικότερων επιπτώσεων στο προστατευόμενο δίκτυο. Ακολουθεί η βιβλιογραφική επισκόπηση του Κεφαλαίου 4, όπου αναλύονται και σχολιάζονται μελέτες συστημάτων ή μηχανισμών παρεμφερών λειτουργιών με αυτό της εργασίας, σε επίπεδο σχεδιασμού, υλοποίησης και απόδοσης. Στο τελευταίο Κεφάλαιο επισημαίνονται τα συμπεράσματα και οι περιορισμοί της παρούσας υλοποίησης και προτείνονται ορισμένες κατευθύνσεις για μελλοντική έρευνα και βελτίωση.

1

ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

1.1 Ανάλυση των IDS/IPS

Σημαντική σημείωση πριν την ανάλυση που ακολουθεί είναι η ξεκάθαρη έμφαση στη διαφοροποίηση των συστημάτων IDPS και IPS, καθώς στην υφιστάμενη βιβλιογραφία οι έννοιές τους τείνουν να συγχέονται συχνά. Στην παρούσα εργασία οι μονάδες (modules) των μηχανισμών Detection και Prevention επιτελούν διαφορετικές διεργασίες σε διακριτά χρονικά πλαίσια, και συνιστούν συνολικά διαφορετικές οντότητες ενός ευρύτερου συστήματος. Κύρια διαφορά των δύο αποτελεί η ιδιότητα του IPS να επηρεάζει άμεσα τη ροή των δεδομένων μέσα στο δίκτυο, σε αντίθεση με την φύση των IDS η οποία βασίζεται κυρίως σε παθητική παρακολούθηση και γρήγορη εξαγωγή ειδοποιήσεων.

Intrusion Detection Systems

Τα Συστήματα Ανίχνευσης Εισβολών (Intrusion Detection Systems) είναι εργαλεία λογισμικού ή υλικού τα οποία αυτοματοποιούν τη διαδικασία παρακολούθησης των συμβάντων που λαμβάνουν χώρα σε ένα υπολογιστικό σύστημα ή δίκτυο, και τα αναλύουν για ενδείξεις παραβιάσεων ασφαλείας [15]. Μπορούν να ταξινομηθούν σε τρεις ευρύτερες κατηγορίες μεθοδολογίας, ανάλογα με τις θεμελιώδεις αρχές και στρατηγικές που διέπουν τον εκάστοτε τρόπο προσέγγισης της ανίχνευσης [16].

- **Signature Based Detection (SD):** Βασίζεται στην αναζήτηση συγκεκριμένων “υπογραφών” (Signatures) μέσα στην κυκλοφορία, δηλαδή συγκεκριμένα μοτίβα ή δείκτες με σκοπό την αντιστοίχιση τους με προκαθορισμένες απειλές. Κάθε “υπογραφή” αποτελεί ουσιαστικά ένα δακτυλικό αποτύπωμα μιας προηγουμένως εντοπισμένης επίθεσης ή κακόβουλης δραστηριότητας.

- **Anomaly Based Detection (AD):** Εντοπίζει επιθέσεις με βάση την ανάπτυξη προφίλ "κανονικής" συμπεριφοράς δικτύου, παρακολουθώντας τα βασικά μοτίβα κίνησης ή λειτουργίες του συστήματος. Με την πάροδο του χρόνου είναι σε θέση να ανιχνεύει αποκλίσεις από το παραπάνω προφίλ κίνησης.

- **Stateful Protocol Analysis (SPA):** Όμοια κατηγορία με την παραπάνω, υπό την έννοια του καθορισμού μιας "κανονικής" συμπεριφοράς κυκλοφορίας και ανίχνευσης των αποκλίσεων από αυτή. Ωστόσο το εν λόγω προφίλ βασίζεται σε λεπτομερή εξέταση της κατάστασης των πρωτοκόλλων δικτύου μεταξύ των συσκευών, κυρίως στο πλαίσιο των επιπέδων 3,4 και 7 του OSI (Open System Interconnection). Για παράδειγμα, σε μια συνεδρία του πρωτοκόλλου TCP (Transmission Control Protocol), εντοπίζει ενδεχόμενες απουσίες ορισμένων βημάτων του three – way – handshake.

Ακόμα ένας τρόπος θεωρητικής προσέγγισης των IDS, είναι η ταξινόμηση τους ανάλογα με το πεδίο εφαρμογής τους. Αναλυτικότερα, σύμφωνα με τους συγγραφείς του [16], πρόκειται για την τοποθέτηση του IDS σε ένα σύστημα ή δίκτυο, εστιάζοντας παράλληλα στον τρόπο με τον οποίο συλλέγει και ερμηνεύει τα δεδομένα. Με αυτό το κριτήριο κατά νου, προκύπτουν οι εξής κατηγορίες τύπου τεχνολογίας:

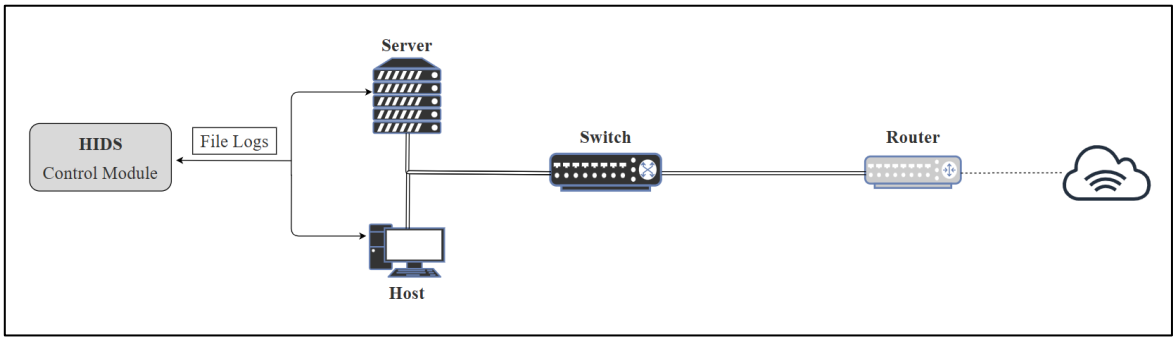
- **Host Based IDS (HIDS):** Εγκαθίσταται σε μεμονωμένους Hosts (π.χ. υπολογιστές ή διακομιστές) και παρακολουθεί τις δραστηριότητες που αφορούν ειδικά το συγκεκριμένο σύστημα. Αναλύει κρίσιμα system και application logs (αρχεία συστήματος και εφαρμογών) με σκοπό να επιβεβαιώσει την ακεραιότητά τους. Παρουσιάζεται αντίστοιχο παράδειγμα αρχιτεκτονικής στην Εικόνα 8, για δύο Hosts.

- **Network Based IDS (NIDS):** Τοποθετείται σε κατάλληλα σημεία στο δίκτυο, προκειμένου να έχει την εποπτεία της κυκλοφορίας όλου (ή ενός προκαθορισμένου μέρους) του δικτύου. Αναλύει τα πακέτα (packets) που ανταλλάσσονται μεταξύ των συστημάτων και ανιχνεύει τις επικείμενες επιθέσεις. Παρουσιάζεται αντίστοιχο παράδειγμα αρχιτεκτονικής στην Εικόνα 9, για δύο Hosts.

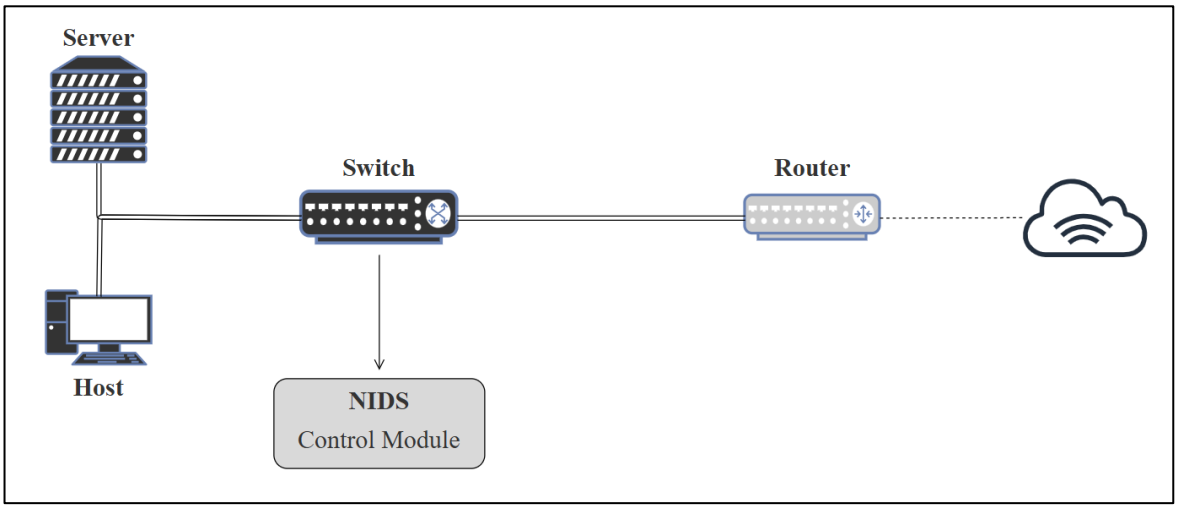
- **Wireless IDS (WIDS):** Όπως υποδηλώνει το όνομά τους, η συγκεκριμένη κατηγορία συστημάτων ειδικεύεται σε ασύρματα δίκτυα και αποτελεί ξεχωριστό κεφάλαιο της μελέτης των IDS.

- **Network Behavior Analysis (NBA):** Λειτουργεί όπως το NIDS, ωστόσο αντί για μεμονωμένα πακέτα εστιάζει σε ευρύτερα μοτίβα κυκλοφορίας μέσα στο δίκτυο.

- **Mixed IDS (MIDS):** Αποτελεί συνδυασμός όλων των παραπάνω κατηγοριών.



Εικόνα 8. Αρχιτεκτονική Host Based IDS (HIDS)



Εικόνα 9. Αρχιτεκτονική Network Based IDS (NIDS)

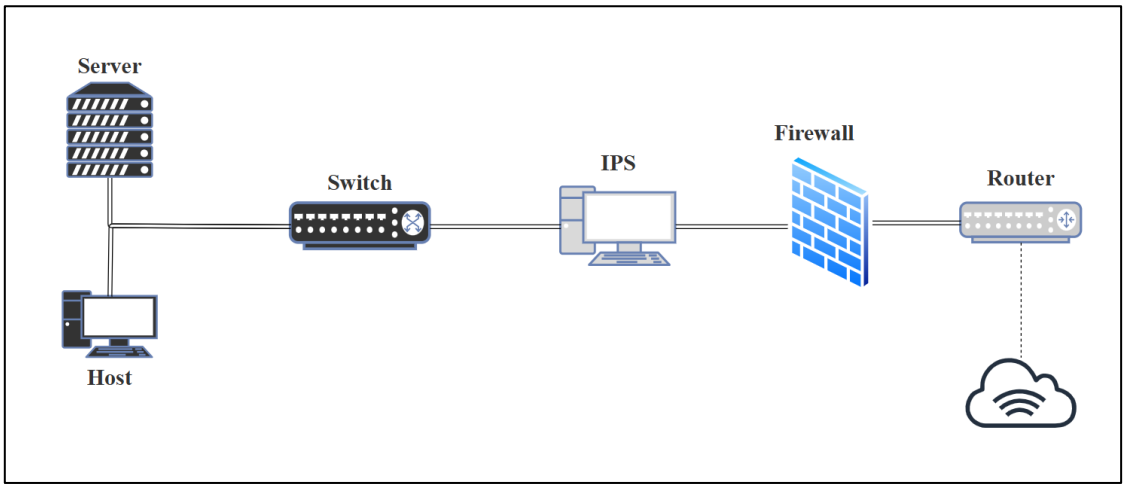
Αντλώντας λοιπόν από τους δύο παραπάνω πυλώνες κατηγοριών, προκύπτει ότι το σύστημα ανίχνευσης επιθέσεων που αναπτύχθηκε για την παρούσα εργασία ακολουθεί τα πρότυπα της βασικής στρατηγικής των Signature Based Detection συστημάτων, καθώς και τον τύπο τεχνολογίας NIDS. Οι λόγοι επιλογής των SD έναντι των AD αφορούν κυρίων τον μεγάλο όγκο των false positive alerts (ψευδών θετικών ειδοποιήσεων) που τείνουν να παράγουν τα τελευταία, σε συνδυασμό με την έλλειψη διαφάνειας σε περίπτωση ανίχνευσης απειλής. Βασίζονται μεταξύ άλλων σε στατιστικά μοντέλα και αλγόριθμους μηχανικής μάθησης με αποτέλεσμα όταν εντοπίζουν μια κυκλοφοριακή ανωμαλία, την επισημαίνουν ως δυνητικά κακόβουλη, χωρίς πάντα να εξηγείται ποιος συγκεκριμένος κανόνας ή πρότυπο παραβιάστηκε. Όσον αφορά τα συστήματα SPA, απαιτείται μεγάλος όγκος πόρων για την ανάπτυξη και την εφαρμογή τους (καθώς απευθύνονται σε δίκτυα μεγάλου εύρους) ξεφεύγοντας έτσι από το πεδίο εφαρμογής της παρούσας μελέτης. Όσον αφορά την επιλογή του τύπου τεχνολογίας του συστήματος, η περιγραφή στην υποενότητα “Σκοπός και διάρθρωση της Εργασίας” φωτογραφίζει πλήρως τις λειτουργίες ενός NIDS. Για τις ανάγκες λοιπόν της παρούσα εργασίας, πρέπει να παρέχεται στο σύστημα ορατότητα σε όλο το εύρος του εκάστοτε εικονικού δικτύου προκειμένου να είναι σε θέση να προστατέψει τις αντίστοιχες συνδεδεμένες συσκευές. Προφανώς, γίνεται λόγος για ενσύρματα δίκτυα, ενώ αποφεύγεται εξ ορισμού η εστίαση σε ευρύτερα μοτίβα κυκλοφορίας λόγω επιλογής της μεθοδολογίας SD. Δια της ατόπου απαγωγής λοιπόν συμπεραίνεται ότι η χρήση του προτύπου NIDS σε συνδυασμό με τη μεθοδολογία SD είναι ιδανικά για τις περιπτώσεις χρήσης της συγκεκριμένης εργασίας, καταλήγοντας έτσι σε ένα SD – NIDS τύπου σύστημα.

Intrusion Prevention Systems

Λαμβάνοντας υπόψη τη διάκριση μεταξύ modules που πραγματοποιήθηκε στην αρχή της ενότητας, τα Συστήματα Πρόληψης Εισβολών (Intrusion Prevention Systems) ορίζονται ως εργαλεία λογισμικού ή υλικού που προσφέρουν δυναμικές και αυτοματοποιημένες λύσεις στην αναχαίτηση ή την πρόληψη των επιβεβαιωμένων - από το IDS - επιθέσεων. Μπορούν να καθυστερήσουν σημαντικά τις απόπειρες εκμετάλλευσης ενός ευπαθούς συστήματος στο δίκτυο (vulnerability), προσφέροντας έτσι πολύτιμο χρόνο στις ομάδες διαχείρισης προκειμένου να δημιουργήσουν τα κατάλληλα λογισμικά επιδιόρθωσης (patches) [17]. Δεν είναι δυνατή η ταξινόμησή τους σε μεγάλες κατηγορίες ανάλογα με τις εκάστοτε τεχνολογίες αντιμετώπισης περιστατικών (Incident Response - IR), λόγω της πληθώρας των τύπων επιθέσεων που αναλαμβάνουν να αντιμετωπίσουν. Ωστόσο, ένα συνηθισμένο Intrusion Prevention System που

εγκαθίσταται σε μικρά δίκτυα [18], τοποθετείται στην άμεση διαδρομή των δεδομένων (inline – συνήθως μεταξύ του μεταγωγέα και του τείχους προστασίας) προκειμένου η κυκλοφορία να περνάει από αυτό προτού φτάσει στον προορισμό της. Πρωταρχικός σκοπός του είναι να εκτελεί τις παρακάτω ενέργειες:

1. Περιορισμός του όγκου εισερχόμενης και εξερχόμενης κυκλοφορίας.
2. Απόρριψη κακόβουλων πακέτων.
3. Αποκλεισμός κυκλοφορίας που προέρχεται από ύποπτες διευθύνσεις.
4. Ρύθμιση των τειχών προστασίας για την αποφυγή μελλοντικών επιθέσεων.
5. Πρόσθετες εξατομικευμένες ενέργειες.



Εικόνα 10. Τοποθέτηση ενός τυπικού IPS

Η μονάδα λοιπόν της αντιμετώπισης και πρόληψης (Prevention Module) που αναπτύχθηκε στην παρούσα εργασία ακολουθεί τα πρότυπα ενός τυπικού μηχανισμού Incident Response, δίνοντας ιδιαίτερη βάση στο βήμα 5 (εξατομικευμένες ενέργειες ανάλογα με την κατηγορία επίθεσης και τον τύπο θύματος). Ωστόσο ο τρόπος με τον οποίο τοποθετείται μέσα δίκτυο και αναλαμβάνει να ολοκληρώσει τα εν λόγω βήματα στηρίζεται σε μια καινοτόμα τεχνολογία (Dynamic Shield Deploy), σχεδιασμένη κυρίως για την πρόληψη των ενδογενών απειλών που αναλύθηκαν στην Εισαγωγική Ενότητα. Το εν λόγω σύστημα λοιπόν στην ολότητά του μπορεί να χαρακτηριστεί ως ένα SD – NIDPS, βασιζόμενο σε τεχνολογία Dynamic Shield Deploy.

1.2 Η τεχνολογία Dynamic Shield Deploy

Γενική Περιγραφή

Όπως τα firewalls, έτσι και τα συνηθισμένα IPS σε Inline Mode πασχίζουν να αποτρέψουν επιθέσεις οι οποίες πηγάζουν από τα ενδότερα του ίδιου του δικτύου. Μια νέα προσέγγιση καταπολέμησης του παραπάνω φαινομένου, που δεν έχει μέχρι σήμερα αναλυθεί στη σχετική βιβλιογραφία, είναι η τεχνολογία αυτόματης αντιμετώπισης επιθέσεων DSD (Dynamic Shield Deploy). Βασική προϋπόθεση για την εφαρμογή και τη λειτουργία της είναι η επέκταση των υφιστάμενων αρχιτεκτονικών δικτύων μικρής και μεσαίας εμβέλειας, ως προς τη δημιουργία επιπλέον καναλιών επικοινωνίας. Μέσω αυτών, συνδέεται η φυσική συσκευή που φέρει το λογισμικό ανίχνευσης και αντιμετώπισης, με το εκάστοτε σύστημα που ανήκει στο προστατευόμενο σύνολο. Συνίσταται έτσι ένα νέο, εσωτερικό (απομονωμένο) δίκτυο, μέσα στο οποίο έχουν τη δυνατότητα να επικοινωνούν μόνο οι παραπάνω συμμετέχοντες, ανεξαρτήτως της κατάστασης του κυρίως δικτιού.

Ξεκινώντας λοιπόν από το πέρας της διαδικασίας ανίχνευσης, μόλις διαγνωστεί μια απειλή από το αρχείο καταγραφής ειδοποιήσεων της μονάδας ID (Intrusion Detection Module), πραγματοποιούνται κατάλληλες αυτοματοποιημένες ενέργειες προκειμένου να αντληθούν οι απαραίτητες πληροφορίες σχετικά με την κατηγορία της επίθεσης, τις διευθύνσεις των επιτιθέμενων, καθώς και αυτές των προστατευόμενων συστημάτων που εκτίθενται στον κίνδυνο. Σε αυτό το σημείο, ανάλογα με τον τύπο της επίθεσης και τα δεδομένα που έχουν συλλεχθεί από τα logs του Detection Module, ενδέχεται να εκτελεστεί σχετικός έλεγχος για τεχνικές παραποίησης της ταυτότητας του επιτιθέμενου με σκοπό την υπεκφυγή του από το αμυνόμενο σύστημα (evasion techniques). Στην περίπτωση όπου το παραπάνω σενάριο επιβεβαιωθεί, πραγματοποιείται έλεγχος ανίχνευσης των πραγματικών διευθύνσεων των κακόβουλων συστημάτων.

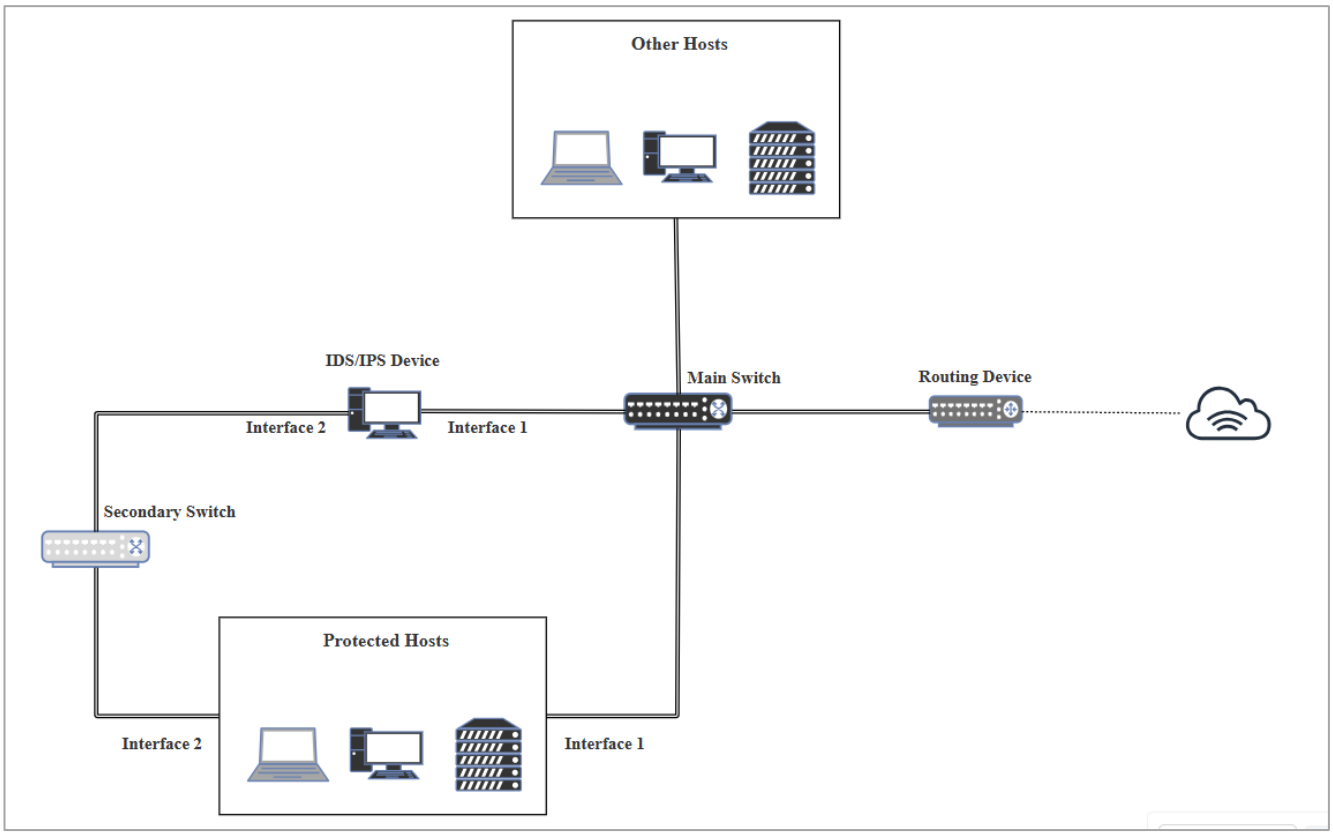
Ύστερα, ενεργοποιείται συνεδρία ασφαλούς απομακρυσμένης σύνδεσης στις μηχανές – θύματα, χρησιμοποιώντας την επέκταση του δικτύου που αναφέρθηκε παραπάνω. Σκοπός της συγκεκριμένης συνεδρίας αποτελεί η εγκατάσταση ειδικά σχεδιασμένων πρωτοκόλλων ασφαλείας (Shields – Ασπίδες) τα οποία έχουν προκύψει από δυναμική ανάλυση των δεδομένων που συλλέχθηκαν μέχρι το συγκεκριμένο χρονικό σημείο. Το πρωτόκολλο αυτό εκτελείται αυτόματα και θωρακίζει το θύμα καθώς περιέχει μια αλληλουχία ενεργειών που κυμαίνονται από απλές ρυθμίσεις με σκοπό την απόκλιση του επιτιθέμενου και την απόρριψη των κακόβουλων πακέτων, έως σύνθετες παραμετροποιήσεις στον πύρινα του λειτουργικού συστήματος (kernel).

Αυτή η πρωτότυπη τεχνολογία ενσωμάτωσης της Μονάδας Πρόληψης (Prevention Module) στα δίκτυα ενισχύει την αποτελεσματικότητα του μηχανισμού άμυνας σε αυτά, καθώς καθιστά δυνατή την άμεση και προσαρμοσμένη απόκριση, προτού η επίθεση επιτύχει να προκαλέσει εκτεταμένη ζημιά. Ενώ άλλες μέθοδοι πρόληψης συχνά περιορίζονται σε στατική ανάλυση και γενική απόκριση, η συγκεκριμένη τεχνική προσφέρει μια πλήρως δυναμική λύση, εξασφαλίζοντας πολύτιμο χρόνο στις ομάδες διαχείρισης της ασφάλειας να δράσουν αποτελεσματικά.

1.3 Διαγραμματική Πλαισίωση Σχεδιασμού

Προκειμένου να αναλυθεί πλήρως η θεωρητική υπόσταση του συστήματος που μελετάται, είναι απαραίτητη η οπτικοποίηση των βασικών του λειτουργιών μέσω αντίστοιχων διαγραμμάτων. Τα τελευταία, θα βοηθήσουν στην κατανόηση της λειτουργίας του συστήματος ασφάλειας στην ολότητά του, μέσω της επίδειξης των αλληλεπιδράσεων μεταξύ των επιμέρους υποσυστημάτων του, κατά τη διάρκεια μιας επίθεσης. Επίσης, είναι εξίσου σημαντική η ανάδειξη μιας γενικής τοπολογίας δικτύου, η οποία θα πληροί τις προϋποθέσεις αρχιτεκτονικών που συζητήθηκαν παραπάνω (δύο δικτυακές ζώνες), ως πρότυπο για την ενσωμάτωση του συστήματος IDS/IPS. Στο παρακάτω Network Diagram παρουσιάζεται μια γενική αναπαράσταση αρχιτεκτονικής των εν λόγω δικτύων.

Network Diagram



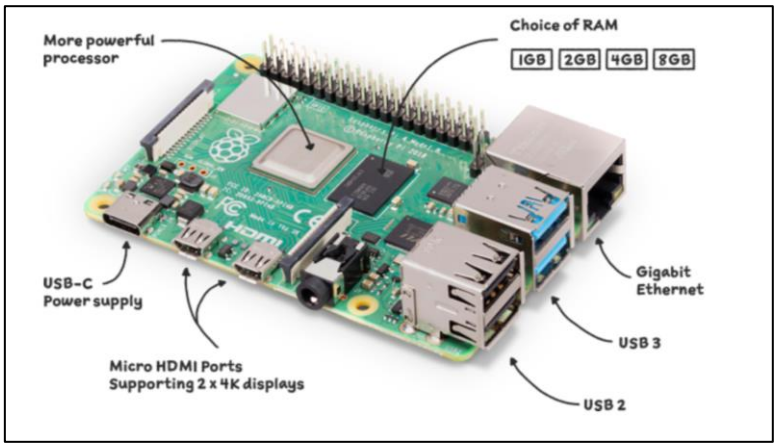
Εικόνα 11. Γενική Αρχιτεκτονική Δικτύου

Παρατηρώντας το παραπάνω διάγραμμα, το σύστημα υλικού “IDS/IPS Device” φέρει το λογισμικό ανίχνευσης και αντιμετώπισης επιθέσεων. Μέσω του “Main Switch” (πρωτεύων μεταγωγέας) συνδέεται στο κύριο δίκτυο όπου συνυπάρχει με τα σύνολα των “Προστατευόμενων” και “’Άλλων” συστημάτων. Από το “Interface 1” έχει τη δυνατότητα να δέχεται όλη την υφιστάμενη κυκλοφορία του κυρίως δικτύου, ανεξάρτητα από τον αποστολέα και τον παραλήπτη των επιμέρους πακέτων πληροφορίας που μεταδίδονται σε αυτό. Τα πακέτα ελέγχονται για ενδείξεις παραβιάσεων του πρωτοκόλλου ασφάλειας (ακολουθώντας τη στρατηγική Signature Based Detection που αναλύθηκε παραπάνω), και στη συνέχεια προκύπτει η κατάλληλη ειδοποίηση για την επιβεβαίωσης της επίθεσης. Σε αυτό το σημείο ενεργοποιείται και ο μηχανισμός της τεχνολογίας Dynamic Shield Deploy. Χρησιμοποιείται το δευτερεύων δίκτυο, που διαχειρίζεται από τον “Secondary Switch”, και συγκεκριμένα η διεπαφή 2 (“Interface 2”) προκειμένου να εδραιωθεί η προαναφερθείσα συνεδρία σύνδεσης με τις μηχανές του συνόλου Protected Hosts. Μέσω αυτής μεταφέρονται και εκτελούνται τα εξατομικευμένα πρωτόκολλα θωράκισης στις συσκευές – στόχους.

Ενδεικτικό του χαρακτηριστικού απομόνωσης του δευτερεύοντος δικτύου από εξωγενείς παράγοντες αποτελεί η έλλειψη μιας “Routing Device” (Συσκευής δρομολόγησης). Η τελευταία μπορεί να αναπαριστά οποιαδήποτε φυσική συσκευή διαθέτει δυνατότητες δρομολόγησης, όπως Routers και Layer 3 Switches για δρομολόγηση σε άλλα VLANs, Gateways για δρομολόγηση στο Internet ή Load Balancers για απαιτήσεις μεγαλύτερων δικτύων (όχι συνηθισμένο στο συγκεκριμένο πεδίο εφαρμογής).

Στη συγκεκριμένη παράγραφο θα πραγματοποιηθεί μια μικρή αναφορά στις επιμέρους φυσικές συσκευές που δομούν μια αρχιτεκτονική δικτύου της Εικόνας 11, εφόσον η παρούσα μελέτη βασίστηκε σε προσομοίωση των παραπάνω. Ξεκινώντας λοιπόν από την “IDS/IPS Device”, αναπαριστά οποιαδήποτε συσκευή υλικού η οποία παρέχει αρκετούς υπολογιστικούς πόρους, και μπορεί να τρέξει κάποια διανομή Linux, εφόσον η παρούσα εργασία είναι βασισμένη σε αυτό. Προτείνεται η χρήση του Raspberry Pi 4, καθώς παρέχει όλα τα απαραίτητα δομικά μέρη μιας ικανής υπολογιστικής μονάδας, που θα λειτουργεί σε υψηλή απόδοση, δεδομένου του μικρού/μεσαίου μεγέθους των δικτύων που εφαρμόζεται. Προτιμάται κυρίως λόγω της προσιτής τιμής του, αλλά και τις ευελιξίας του τόσο όσον αφορά το μικρό μέγεθος του, όσο και το λογισμικό ανοιχτού κώδικα που υποστηρίζει (Raspberry Pi OS βασισμένο σε Debian Linux). Στην Εικόνα 12 παρατίθεται το μοντέλο Raspberry Pi 4. Η “IDS/IPS Device”, συνδέεται στον “Main Switch” μέσω κατάλληλης θύρας Ethernet και λαμβάνει όλη την κυκλοφορία του κυρίως δικτύου. Συνεπώς, χρειάζεται ένας μεταγωγέας ο οποίος εκτός από τη δυνατότητα διαχείρισης του φόρτου της κυκλοφορίας σε ένα δίκτυο μικρομεσαίων απαιτήσεων, πρέπει να υποστηρίζει τη δημιουργία Virtual – LANs, αλλά και την λειτουργία Port Mirroring. Η τελευταία, δίνει τη δυνατότητα του “αντικατοπτρισμού”

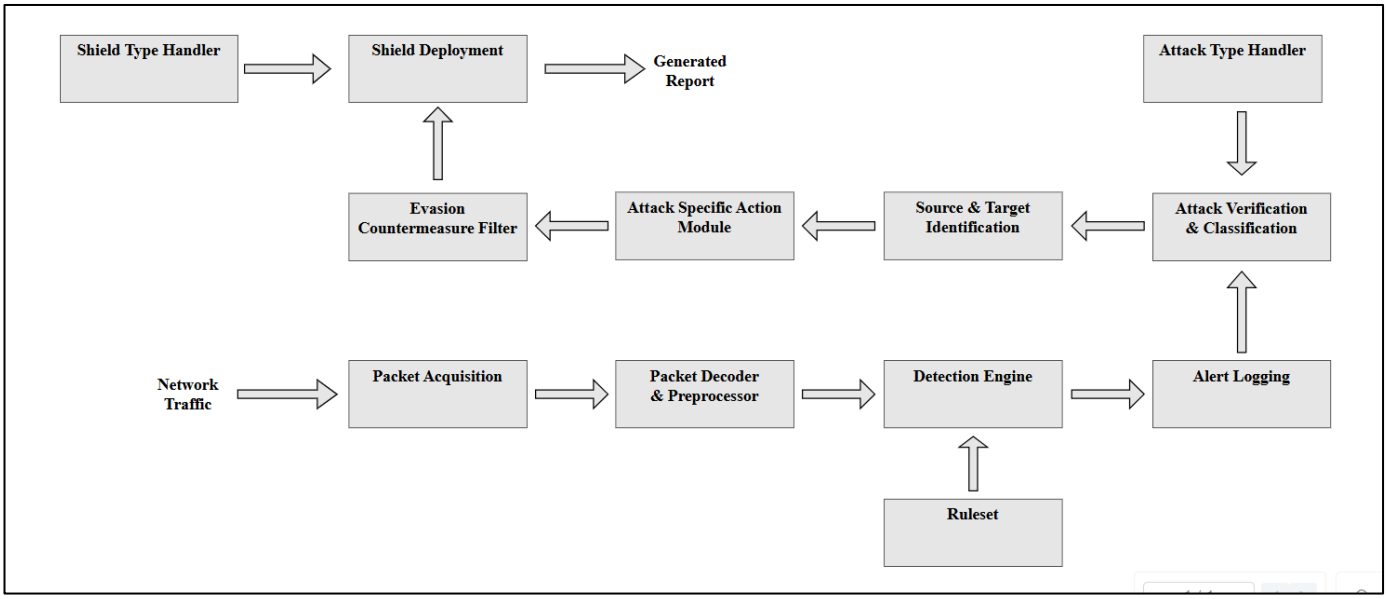
την κυκλοφορίας που υφίσταται μέσα στο δίκτυο, στην θύρα μεταγωγής που είναι συνδεδεμένη η συσκευή ασφάλειας, ανεξαρτήτως Port αποστολέα ή παραλήπτη. Ιδανικό για τα παραπάνω αλλά και για τα πρόσθετα προηγμένα χαρακτηριστικά που διαθέτουν, είναι οι managed μεταγωγείς της σειράς "Cisco Business 300" [20], και συγκεκριμένα αυτοί που διαθέτουν από 10 έως 25 συνολικές θύρες Ethernet, δεδομένου του περιορισμένου αριθμού συνολικών Hosts. Όσον αφορά τον "Secondary Switch", συνδέονται σε αυτόν τόσο η συσκευή IDS/IPS όσο και οι προστατευόμενοι Hosts, ενεργοποιώντας δεύτερη διεπαφή δικτύου, την οποία είτε την έχουν ήδη ενσωματωμένη, είτε τη δημιουργούν από αντάτπορες USB – To – Ethernet. Για λόγους βελτιωμένης ασφάλειας, και συγκεκριμένα για να ενσωματωθεί μηχανισμός πρόσβασης μόνο σε εξουσιοδοτημένα άτομα στο συγκεκριμένο δίκτυο, λόγω της αυξημένης εμπιστευτικότητας που πρέπει να υπάρχει σε αυτό, προτείνεται η απόκτηση παρεμφερούς μεταγωγέα με τον "Main Switch". Ωστόσο, μπορεί να διαθέτει λιγότερες θύρες από το τελευταίο, καθώς οι προστατευόμενοι Hosts αναπαριστούν μόνο ένα υποσύνολο των συσκευών στο δίκτυο. Ενδεικτικά, στην Εικόνα 13 παρατίθεται ο managed μεταγωγέας Cisco SF300-24PP αποτελούμενος από 24 Ports. Σημειώνεται ότι η ίδια βασική προϋπόθεση για την εγκατάσταση Linux διανομών ισχύει και για όλους τους υπόλοιπους Hosts του δικτύου.



Εικόνα 12. Raspberry Pi 4 [19]

Εικόνα 13. Cisco SF300-24PP Managed Switch [20]

Block Diagram



Εικόνα 14. Block Diagram αναπαράστασης της λειτουργίας του συστήματος

Στο παραπάνω διάγραμμα, απεικονίζεται η δομή του συστήματος που αναπτύχθηκε, καθώς και η φυσική ροή των λειτουργιών της, όπως προκύπτει από τις αλληλεπιδράσεις των επιμέρους υποσυστημάτων του. Η διαδικασία ανίχνευσης εισβολών ξεκινά από την εισαγωγή των πακέτων δικτύου στη μονάδα Packet Acquisition και τελειώνει στην Alert Logging. Η δομή της είναι ενδεικτική των Signature Based Network Συστημάτων Ανίχνευσης Εισβολών (SD – NIDS) και χρησιμοποιείται από τα δημοφιλέστερα εργαλεία της αγοράς [21, 22]. Όσον αφορά τον τομέα της αντιμετώπισης επιθέσεων, το σύνολο των υπόλοιπων Block του διαγράμματος απεικονίζει τις επιμέρους λειτουργίες της τεχνολογίας DSD, από την εξαγωγή και την επεξεργασία των δεδομένων του αρχείου καταγραφών ειδοποιήσεων, έως την τελική εγκατάσταση του εκάστοτε εξατομικευμένου Shield. Στη συσκευή υλικού που φέρει το σύστημα αυτό, θα προκύπτει κατάλληλη αναφορά, σύμφωνα με τα αποτελέσματα της εκτέλεσης του Shield, αλλά και όλα τα Outputs που κατασκευάστηκαν για να δείξουν τη λειτουργία των εκάστοτε μονάδων αντιμετώπισης επιθέσεων. Ακολουθεί σύντομη παρουσίαση των λειτουργιών του κάθε Block.

Packet Acquisition

Μονάδα συλλογής όλων των πακέτων εισερχόμενης κυκλοφορίας στην καθορισμένη διεπαφή δικτύου και μεταφοράς τους στα επόμενα στάδια του αγωγού ανίχνευσης.

Packet Decoder & Preprocessor

Η μονάδα του αποκωδικοποιητή επιμελείται κυρίως του προσδιορισμού των υποκείμενων πρωτοκόλλων που χρησιμοποιούνται σε κάθε πακέτο, ξεκινώντας από τα πρωτόκολλα του Data Link Layer και ανεβαίνοντας μέχρι και αυτά του Application Layer. Οι Preprocessors εκτελούνται ύστερα από το Decode και χρησιμεύουν κατά βάση στην κανονικοποίηση της κίνησης (Normalization) και τη ανασύνθεση των κατακερματισμένων πακέτων (Reassembly), προετοιμάζοντας τα για την μηχανή ανίχνευσης. Μπορούν να χρησιμεύσουν παράλληλα και στην ανίχνευση επιθέσεων οι οποίες δεν μπορούν να αναπαρασταθούν από κάποιο υφιστάμενο Ruleset (βλ. παρακάτω).

Detection Engine

Αναλύει την κυκλοφορία βάσει προκαθορισμένων κανόνων για τον εντοπισμό επιθέσεων. Συγκεκριμένα, αντιστοιχίζει μοτίβα των δεδομένων από τα πακέτα που δέχεται από την προηγούμενη μονάδα, με σύνολο υπογραφών (signatures) που υποδεικνύουν την ύπαρξη απειλής. Επισημαίνει τις ύποπτες δραστηριότητες με κατάλληλες ειδοποιήσεις – συναγερμούς.

Ruleset

Βιβλιοθήκη κατασκευασμένων κανόνων, που καθορίζουν μοτίβα επίθεσης. Περιλαμβάνει κανόνες με υπογραφές των 6 κατηγοριών επιθέσεων που θα αναλυθούν, καθώς και ορισμένους που υποδεικνύουν υπό – κατηγορίες των εν λόγω επιθέσεων. Πρέπει να διατηρείται συνεχώς ενήμερη με νέους κανόνες από τη διαχείριση, προκειμένου να αποφύγει τη στατική φύση της (μειονέκτημα στα σύγχρονα συστήματα κυβερνοασφάλειας).

Alert Logging

Καταγράφει τα συμβάντα όπου η Detection Engine αντιστοίχισε με κανόνα και επισήμανε με ειδοποίηση – συναγερμό, προκειμένου να προχωρήσουν στην επόμενη μονάδα για περεταίρω ανάλυση. Συγκεκριμένα, αποθηκεύονται οι διευθύνσεις IP (και MAC) προέλευσης και προορισμού, ο τύπος επίθεσης, καθώς και ενδεχομένως περεταίρω πληροφορίες όπως το πρωτόκολλο η χρονοσφραγίδα (timestamp) και οι αριθμοί θυρών (Ports).

Attack Verification & Classification

Ενεργοποιείται μόνο όταν προκύψει μια νέα καταγραφή στα Alert Logs, και ανάλογα με τον τύπο των επιθέσεων, είναι σε θέση να διασταυρώσει κατάλληλα τις ειδοποιήσεις προκειμένου να επαληθεύσει εάν αποτελούν πραγματικές επιθέσεις ή ψευδώς θετικές (μόνο εάν χρειαστεί). Παράλληλα, ταξινομεί τις επιθέσεις σύμφωνα με των τύπο που αναγράφεται στα Logs των ειδοποιήσεων και βάσει αυτού, επιλέγεται η στρατηγική αντιμετώπισης από την μονάδα Attack Type Handler.

Attack Type Handler

Αποθετήριο συνόλων προκαθορισμένων οδηγιών που επιλέγονται από το σύστημα, ανάλογα με την παραπάνω ταξινόμηση. Σε κάθε τύπο επίθεσης που μελετάται, τα Logs που προκύπτουν πρέπει να διαχειριστούν κάθε φορά με διαφορετικό τρόπο από τις επερχόμενες μονάδες προκειμένου να προκύψουν έγκυρα αποτελέσματα. Οι οδηγίες αυτές λοιπόν τίθενται από τον παρόν χειριστή και ρυθμίζουν το παραπάνω ζήτημα. με αποτέλεσμα κάθε τύπος επίθεσης να μελετάται ξεχωριστά. Βοηθάει επίσης στην ευελιξία και την κλιμάκωση του συστήματος σε μεγαλύτερο σύνολο τύπου επιθέσεων.

Source & Target Identification

Έχοντας λάβει τις κατάλληλες οδηγίες από το παραπάνω, η συγκεκριμένη μονάδα εξάγει και καταγράφει όλες τις διευθύνσεις IP (όπου χρειαστεί και MAC) από τις επικυρωμένες ειδοποιήσεις της μονάδας Classification. Εν συνεχεία τις διακρίνει σε κακόβουλες (επιτιθέμενου) και προστατευόμενες (συστήματα – στόχοι της επίθεσης). Φροντίζει να απορρίπτει τις διπλότυπες καταγραφές.

Attack Specific Action Module

Μονάδα εκτέλεσης προκαθορισμένων ενεργειών σύμφωνα με τις οδηγίες που αντλήθηκαν από το Attack Type Handler, δηλαδή με βάση την ταξινόμηση της επίθεσης. Η στρατηγική αντιμετώπισης εξατομικεύεται στην εκάστοτε επίθεση ξεχωριστά, συνεπώς εάν μέχρι την παρούσα μονάδα απαιτούνται περεταίρω βήματα επεξεργασίας και αλληλεπίδρασης με τον χρήστη, εκτελούνται εδώ. Δεν είναι υποχρεωτική σε όλα τα σενάρια επιθέσεων, και παραλείπεται όταν το καθορίζει ο Attack Type Handler.

Evasion Countermeasure Filter

Όπως και η παραπάνω, έτσι και αυτή λειτουργεί σύμφωνα με τη βασική γραμμή που έθεσε ο Handler. Όταν χρειαστεί, δέχεται τις κακόβουλες διευθύνσεις IP (ενδεχομένως και τις MAC) που προέκυψαν από το αποτέλεσμα της μονάδας Source & Target Identification, και τις ελέγχει για γνωστές τεχνικές παραποίησης ή απόκρυψης. Τις συγκρίνει με τις ήδη γνωστές διευθύνσεις των προστατευόμενων συστημάτων, ενώ παράλληλα σαρώνει το δίκτυο (Network Scan) προκειμένου να ανακαλύψει περεταίρω στοιχεία για τις κακόβουλες IP και MAC. Στην περίπτωση όπου ο έλεγχος επαληθευτεί για μια ή παραπάνω διευθύνσεις, τότε αυτές απορρίπτονται από τη συνέχεια της διαδικασίας, ενώ αν το Scan επιστρέψει αξιόπιστα δεδομένα, χρησιμοποιούνται και αυτά. Οι τελικές κακόβουλες IP (ή και MAC) που θα εξέλθουν του φίλτρου θα καταλήξουν εν τέλει σε λίστα του Firewall κάποιων Hosts, επομένως η μονάδα αυτή χρειάζεται ιδιαίτερη προσοχή προκειμένου να μην μπλοκάρονται εκ παραδρομής διευθύνσεις όπως αυτή του Router ή άλλων βασικών συσκευών.

Shield Deployment

Το σύστημα φτάνει σε αυτή τη μονάδα έχοντας περάσει από οποιοδήποτε συνδυασμό των παραπάνω δύο μονάδων. Είτε έχει προηγηθεί επεξεργασία και στις δύο μονάδες, είτε σε μια, είτε η ροή δεδομένων έχει φτάσει κατευθείαν από τη μονάδα Source & Target Identification, πρωταρχικός σκοπός της συγκεκριμένης είναι η ομαδοποίηση των κακόβουλων και των προστατευόμενων διευθύνσεων, καθώς και διαφόρων δεδομένων που θα χρειαστούν στην θωράκιση των θυμάτων. Για κάθε προστατευόμενη διεύθυνση που δέχεται επίθεση, ανάλογα με τον τύπο της απειλής (δηλαδή την οδηγία του Attack Type Handler) επιλέγει το κατάλληλο Shield από τον Shield Type Handler και ενσωματώνει σε αυτό τα παραπάνω ομαδοποιημένα δεδομένα. Στη συνέχεια, επιφορτίζεται με την εγκατάσταση και την εκτέλεση των Shields σε όλες τις

διευθύνσεις προστατευόμενων συστημάτων που δέχονται επίθεση, όπως αυτές προκύπτουν από την προηγούμενη ανάλυση.

Shield Type Handler

Όπως ο προηγούμενος Handler, έτσι και αυτός λειτουργεί ως αποθετήριο των Shields που θα εκτελεστούν στα συστήματα που δέχονται επίθεση, και σε πραγματικό χρόνο θα τα θωρακιστούν από την απειλή. Κάθε Shield επιλέγεται δυναμικά, καθώς όπως εξηγήθηκε και παραπάνω, κάθε επίθεση διαχειρίζεται με ξεχωριστή στρατηγική. Είναι ανάγκη να ενσωματωθεί με τις περισσότερες δυνατές επιβεβαιωμένες κακόβουλες διευθύνσεις IP (ή και MAC) καθώς μια κεντρική λειτουργία που επιτελείται σε όλα τα σενάρια των επιθέσεων είναι η ρύθμιση του τείχους προστασίας του Host προκειμένου να απορρίπτει την εισερχόμενη/εξερχόμενη κυκλοφορία από/προς τις εν λόγω διευθύνσεις. Η υπόλοιπες εντολές αφορούν επί το πλείστων εξατομικευμένες ενέργειες, όπως η αναδιαμόρφωση των σχετικών δομών δεδομένων του συστήματος που αλλοιώθηκαν από την επίθεση.

1.5 Περιγραφή Επιθέσεων

Σε αυτή την ενότητα θα εξεταστούν οι επιθέσεις που αναλύθηκαν στο πλαίσιο της συγκεκριμένης μελέτης. Κάθε επίθεση θα περιγραφεί λεπτομερώς με σκοπό να αναδειχθούν τα κύρια χαρακτηριστικά της και να αποσαφηνιστεί ο τρόπος με τον οποίο εκμεταλλεύεται τρωτά σημεία του συστήματος ή του δικτύου. Θα καλυφθεί ο σκοπός της επίθεσης, η μεθοδολογία, και οι συνέπειες για το σύστημα-στόχο.

NMAP SCAN(s)

Το Nmap (Network Mapper) είναι ένα δημοφιλές εργαλείο ανοικτού κώδικα για την ανακάλυψη δικτύων και τον έλεγχο ασφάλειας [23]. Αναπτύχθηκε από τον Gordon Lyon και χρησιμοποιείται ευρέως από διαχειριστές δικτύων και επαγγελματίες στο χώρο του Cyber Security για την ταυτοποίηση συσκευών σε ένα δίκτυο, την ανακάλυψη ανοικτών θυρών, την ανίχνευση υπηρεσιών και εφαρμογών που λειτουργούν σε αυτές τις θύρες με σκοπό την αξιολόγηση της ασφάλειας των δικτυακών συστημάτων. Οι κυρίαρχες δυνατότητες του είναι οι εξής:

1. Host Discovery: Σαρώνει (Scan) όλο το υφιστάμενο δίκτυο ή μεμονωμένες συσκευές προκειμένου να αποκαλύψει τα συστήματα που είναι ενεργά ανά πάσα στιγμή.

2. Port Scanning: Ανιχνεύει τις ανοιχτές θύρες (open ports) στις συσκευές του δικτύου και παράλληλα επιδεικνύει ποιες υπηρεσίες είναι διαθέσιμες στα εν λόγω Ports. Ανεξάρτητα λοιπόν από τον αριθμό της θύρας, εάν είναι ανοιχτή μπορεί να ανιχνευθεί από το Nmap. Ορισμένες συνηθισμένες θύρες υπηρεσιών περιλαμβάνουν την HTTP στη 80 και την SSH στην 22.

3. Service and OS Detection: Εκτός από τις διάφορες υπηρεσίες που μπορούν να ανιχνευτούν από το Nmap, δίνεται η δυνατότητα εντοπισμού συγκεκριμένων λογισμικών (και των εκδόσεών τους) που λειτουργούν σε ένα ανοιχτό port παρέχουν την παραπάνω υπηρεσία. Παράλληλα, είναι σε θέση να προσδιορίσει το λειτουργικό σύστημα που χρησιμοποιούν οι συσκευές – στόχοι.

4. Network Mapping: Δημιουργεί έναν οπτικό χάρτη της τοπολογίας δικτύου, ο οποίος υποδεικνύει τον τρόπο σύνδεσης καθώς και συσχετισμού μεταξύ των συσκευών.

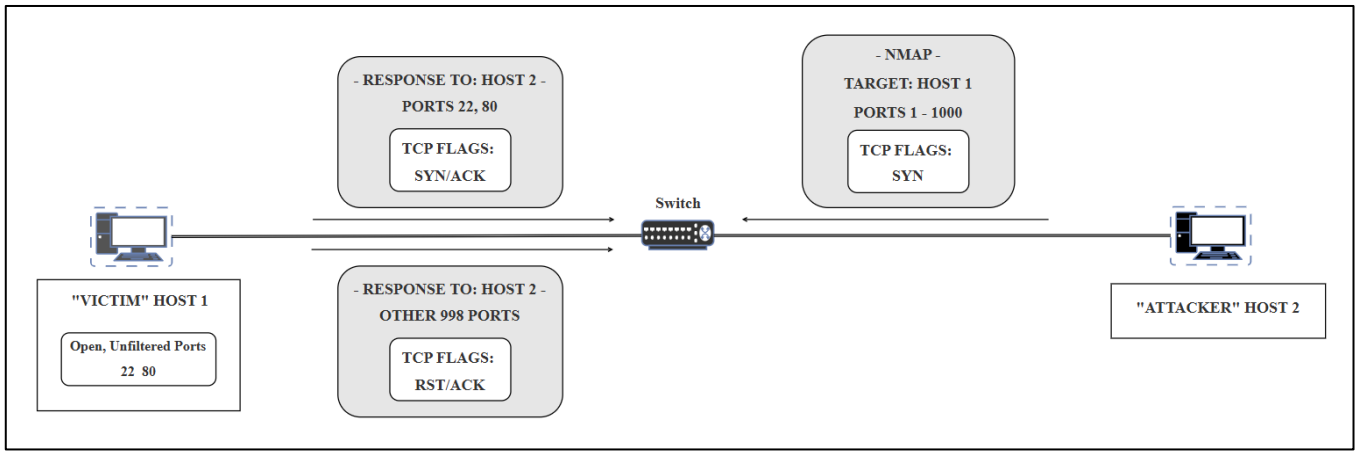
Συμπεραίνεται λοιπόν εύκολα ότι ένα τόσο ισχυρό εργαλείο μπορεί να χρησιμοποιηθεί στα πλαίσια των κυβερνοεπιθέσεων. Δεν μπορεί από μόνο του να ευσταθεί ως τρόπος εξαπόλυσης επιθέσεων, ωστόσο εκμεταλλεύεται ευρέως για την αναγνώριση των συστημάτων που αποτελούν στόχο. Η τεχνική αυτή (reconnaissance) αποτελεί την πρώτη φάση πολλών κυβερνοεπιθέσεων, κατά τη διάρκεια της οποίας οι επιτιθέμενοι συλλέγουν πληροφορίες σχετικά με τα συστήματα και τα δίκτυα-στόχους για να εντοπίσουν πιθανές ευπάθειες που θα μπορούσαν να αξιοποιηθούν. Στην παρούσα εργασία, επιλέχθηκαν 5 τεχνικές σάρωσης που ανήκουν στην κατηγορία του Port Scanning, και είναι εξαιρετικά δημοφιλείς. Συγκεκριμένα, μελετήθηκαν τα Stealth, Full, Xmas, Fin και Null Scans σε κατάλληλο σύστημα – θύμα, με σκοπό τον προσδιορισμό της κατάσταση των θυρών του (ανοιχτές, κλειστές ή φιλτραρισμένες). Όλες οι παραπάνω τεχνικές σάρωσης εκμεταλλεύονται το πρωτόκολλο TCP και συγκεκριμένα τον τρόπο που τα περισσότερα συστήματα ανταποκρίνονται κατά την προσπάθεια εδραίωσης ενός Three Way Handshake. Στην Εικόνα 15 παρουσιάζονται τα Flags των Standard TCP Headers που χρησιμοποιήθηκαν για τα scans, καθώς και οι αντίστοιχες ερμηνείες τους. Όπως είναι γνωστό, τα εν λόγω Bit Ελέγχου ρυθμίζουν την κατάσταση της σύνδεσης και είναι βασικά για τη μετάδοση των δεδομένων μεταξύ του αποστολέα και του παραλήπτη. Ακολουθεί σύντομη επίδειξη των παραπάνω τύπων scan.

SYN – Synchronize	Εκκινεί μια σύνδεση TCP
ACK – Acknowledge	Αναγνώριση των ληφθέντων δεδομένων
FIN – Finish	Ομαλός τερματισμός σύνδεσης
RST – Reset	Απότομος τερματισμός σύνδεσης
PSH – Push	Αποστολή δεδομένων στην εφαρμογή άμεσα

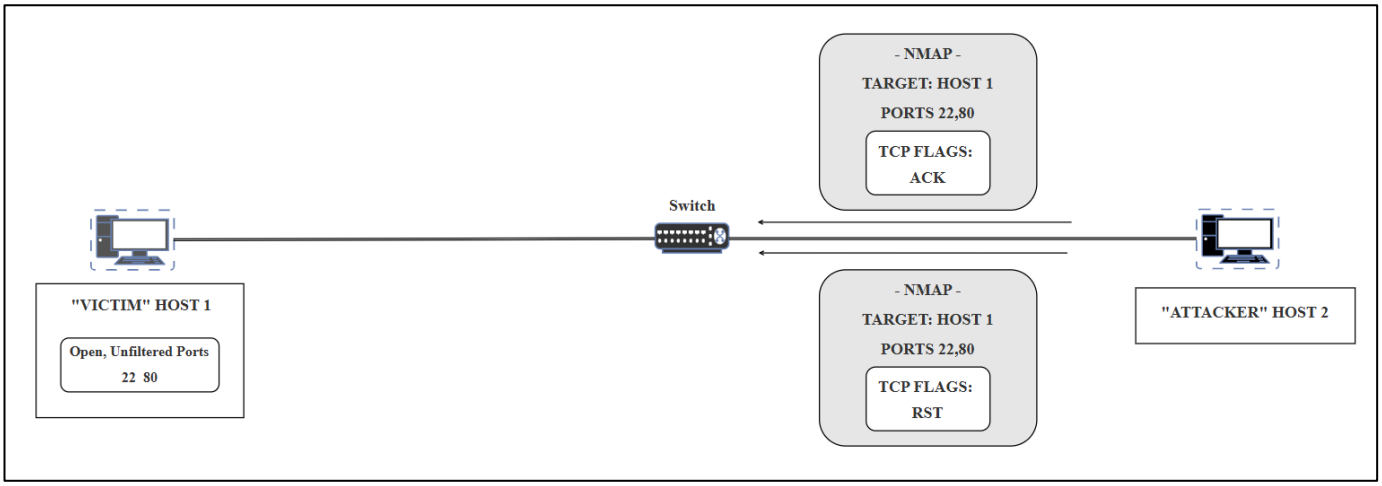
Εικόνα 15. TCP Flags

Full & Stealth Scan

Ξεκινώντας από τον πιο απλό και ευθύ τρόπο σαρώσεων, μέσω του full scan ελέγχεται εάν μια θύρα είναι ανοιχτή εγκαθιστώντας μια πλήρη σύνδεση με αυτή, δηλαδή ολοκληρώνοντας όλα τα απαιτούμενα βήματα μιας τριμελούς χειραψίας (SYN – SYN/ACK – ACK). Εάν η θύρα είναι ανοιχτή, θα προκύψει επιτυχημένη τριμελής χειραψία, ενώ στην αντίθετη περίπτωση (κλειστή), ο στόχος θα απαντήσει με ένα πακέτο RST υποδεικνύοντας το προφανές και τερματίζοντας τη σύνδεση. Ύστερα από την εγκαθίδρυση της σύνδεσης, ο σαρωτής έχοντας λάβει την πληροφορία που χρειάζεται στέλνει ένα δικό του πακέτο RST προκειμένου να την κλείσει αμέσως και να μην γίνει αντιληπτός. Η τεχνική αυτή είναι ιδιαίτερα ακριβής στις ενδείξεις ανοιχτών θυρών, ωστόσο τα σαφή ίχνη που αφήνει στα συστήματα καταγραφής συνδέσεων είναι εύκολα ανιχνεύσιμα. Στην εικόνα 16 παρουσιάζεται η διαδικασία, της αρχικής αποστολής των πακέτων τύπου TCP - SYN στις θύρες 1 έως 1000 του στόχου (η default επιλογή του εργαλείου Nmap), και η απάντησή του με SYN/ACK για αυτές που είναι ανοιχτές (πχ. 80 και 22). Για τις υπόλοιπες κλειστές θύρες απαντάει με TCP RST/ACK για να τερματίσει τις προσπάθειες σύνδεσης στις κλειστές θύρες. Στην εικόνα 17, αποτυπώνεται το δεύτερο σκέλος της σάρωσης, όπου η κακόβουλη συσκευή στέλνει τα αντίστοιχα πακέτα ACK στον στόχο. Σχεδόν ακαριαία, και αφού έχει καταγράψει τις ανοιχτές θύρες, στέλνει παράλληλα TCP - RST για να τερματίσει τη σύνδεση.



Εικόνα 16. Full Scan [1/2].

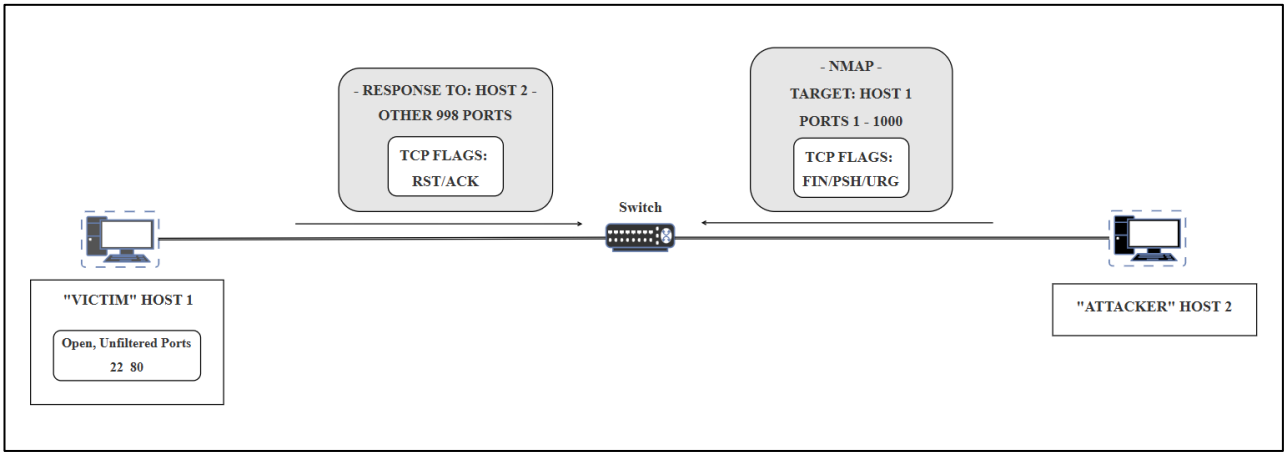


Εικόνα 17. Full Scan [2/2]

Προχωρώντας στο Stealth Scan, ακολουθείται η ίδια λογική ανίχνευσης ανοιχτών θυρών με παραπάνω, με τη διαφορά ότι το επιτιθέμενο σύστημα δεν στέλνει το τελικό ACK για να επιβεβαιωθεί η έναρξη της σύνδεσης, και αντ' αυτού την τερματίζει με RST. Εφόσον έχει ήδη λάβει μήνυμα SYN/ACK μπορεί να αποφανθεί για την κατάσταση του Port (Open, Unfiltered) και δεν προχωράει στην άσκοπη ολοκλήρωση της χειραψίας. Παράλληλα, η εν δυνάμει σύνδεση δεν καταγράφεται σε ενδεχόμενα Log Files που παρακολουθούνται από IDS καθιστώντας τα Stealth Scans μια πιο βιώσιμη επιλογή από τα Full Scans.

Null/Xmas/Fin Scan

Η βασική ιδέα αυτών των τύπων σαρώσεων είναι η αποστολή πακέτων με μη φυσιολογικούς συνδυασμούς σημαιών που δεν ευθυγραμμίζονται με την τυπική διαδικασία σύνδεσης TCP. Τα συστήματα – στόχοι, όταν βλέπουν αυτά τα πακέτα σε κλειστές θύρες απαντούν όπως και παραπάνω, με αναγνώριση και τερματισμό. Ωστόσο όταν φτάνουν σε ανοιχτά Ports, το σύστημα δεν ξέρει πως να επεξεργαστεί τους ασυνήθιστους συνδυασμούς σημαιών επειδή το πακέτο δεν ευθυγραμμίζεται με καμία έγκυρη κατάσταση σύνδεσης. Αποτέλεσμα του παραπάνω είναι να αγνοήσει τα πακέτα που φτάνουν στις ανοιχτές θύρες και να μην στείλει καμία απάντηση, επιβεβαιώνοντας ότι οι συγκεκριμένες δεν είναι κλειστές. Συγκεκριμένα, σε ένα Xmas Scan, τα Flags που τίθενται από το σαρωτή είναι τα FIN, PSH και URG. Όπως υποδεικνύει το όνομά τους, στα Fin Scans έχουν τεθεί οι σημαίες Fin ενώ στα Null δεν τίθεται καμία. Ενδεικτικά στην παρακάτω εικόνα παρατίθεται η διαδικασία ενός Xmas Scan.



Εικόνα 18. Xmas Scan

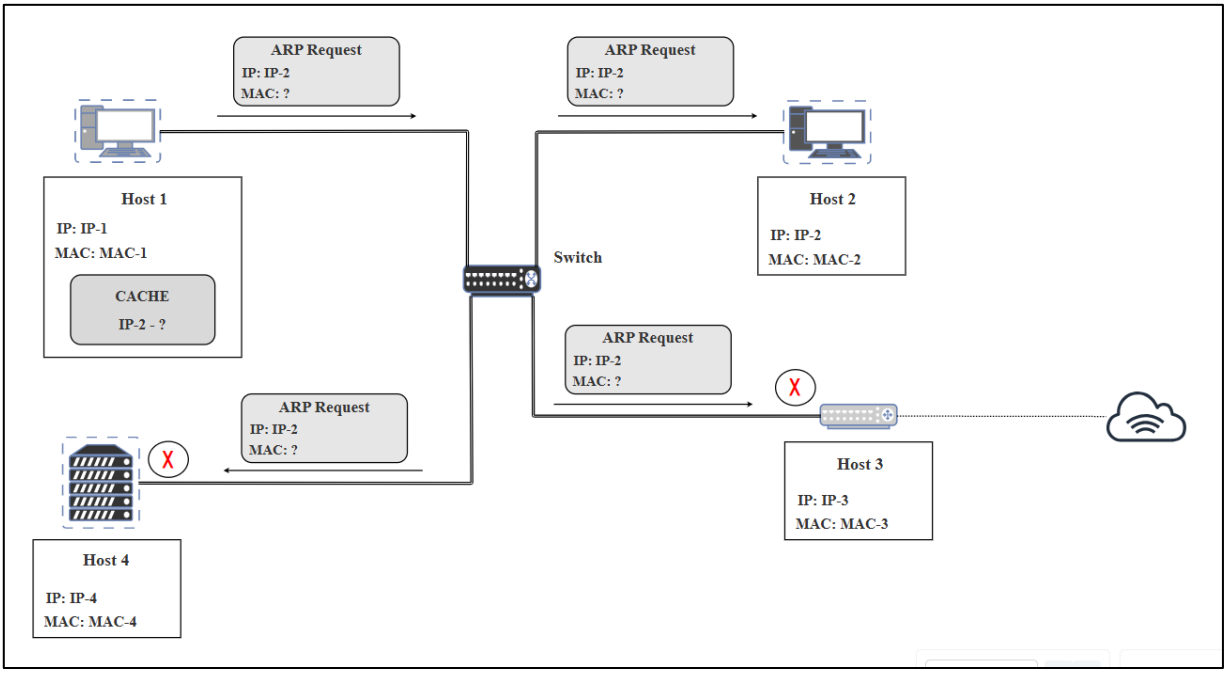
ARP SPOOFING

Το πρωτόκολλο ARP (Address Resolution Protocol) χρησιμοποιείται για την αντιστοίχιση μιας διεύθυνσης IP (επιπέδου 3 στο μοντέλο OSI), στην διεύθυνση MAC (επιπέδου 2) της ίδιας συσκευής. Είναι απαραίτητο για την επικοινωνία εντός των τοπικών δικτύων (LAN) και χρησιμοποιείται κυρίως σε δίκτυα IPv4. Όταν ένας Host επιθυμεί να αποστείλει πακέτα επικοινωνίας σε κάποιον άλλον μέσα στο LAN πρέπει να γνωρίζει την διεύθυνση MAC του παραλήπτη. Έχοντας ήδη γνώση της IP του τελευταίου, ο αποστολέας ελέγχει την κρυφή μνήμη αντιστοιχίσεων IP με MAC (ARP Cache). Στην περίπτωση όπου ο έλεγχος δεν επιστρέψει κάποια MAC ακολουθούνται τα παρακάτω βήματα προκειμένου να την ανακτήσει:

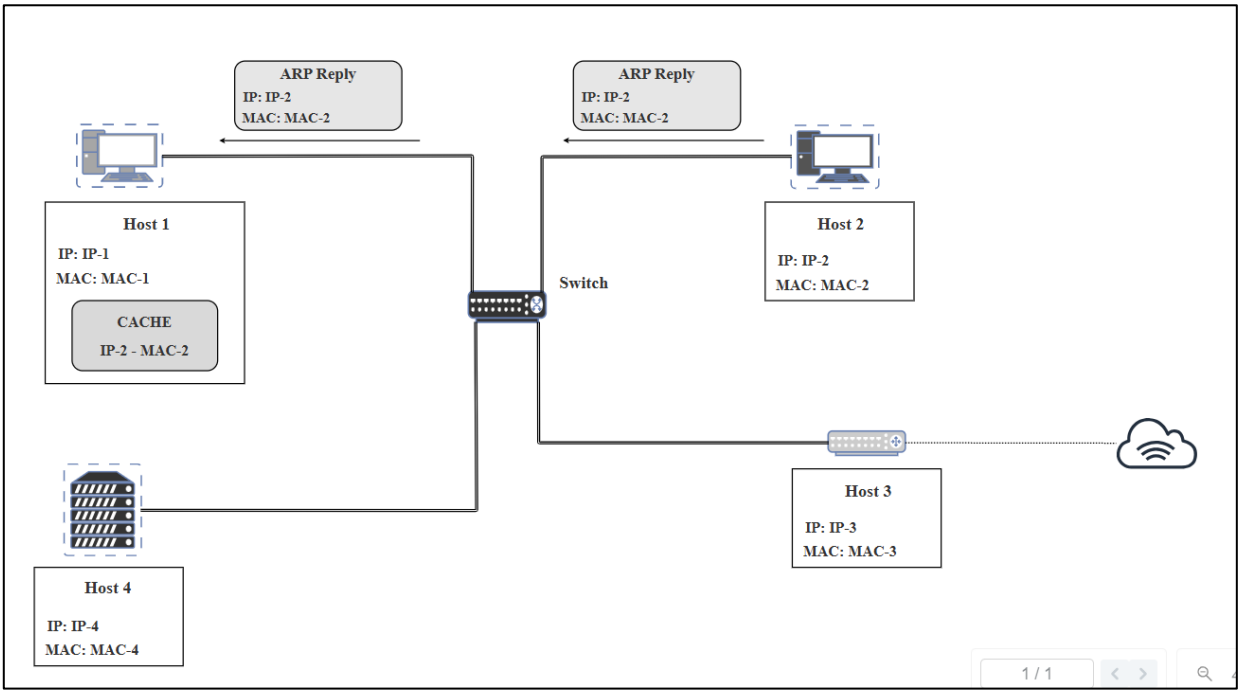
1) **ARP Request.** Εκπέμπει σε όλο το δίκτυο (Broadcast) ένα πακέτο αναζήτησης της διεύθυνσης MAC που αναζητεί, βάσει της της IP που ήδη διαθέτει. Το πακέτο αυτό περιέχει μεταξύ άλλων τις δύο διευθύνσεις του αποστολέα, καθώς και την IP του εν δυνάμει παραλήπτη. Το πεδίο της MAC του τελευταίου μένει κενό με σκοπό να συμπληρωθεί από τον κατάλληλο Host. Όταν οι υπόλοιπες συσκευές παραλάβουν το παραπάνω μήνυμα, συγκρίνουν την IP με αυτή που ήδη διαθέτουν. Εάν αυτές δεν ταυτίζονται τότε αγνοούν το μήνυμα.

2) **ARP Reply.** Εάν ο παραπάνω έλεγχος επαληθευτεί, τότε ο Host που παρέλαβε το ARP Request αποστέλλει ένα αντίστοιχο πακέτο απάντησης (Reply). Αυτό περιέχει όλες τις πληροφορίες του προηγούμενου συμπεριλαμβανομένου και του πεδίου της ζητούμενης MAC Address. Σημειώνεται ότι στέλνεται απευθείας και όχι μέσω Broadcast.

3) **Caching.** Η συσκευή μόλις παραλάβει το πακέτο Reply, ανανεώνει την αντιστοιχία IP – MAC της ARP Cache του και πλέον είναι σε θέση να προχωρήσει στην επιθυμητή επικοινωνία. Οι εν λόγω καταχωρήσεις ARP έχουν μικρή διάρκεια ζωής.



Εικόνα 19. ARP - Request

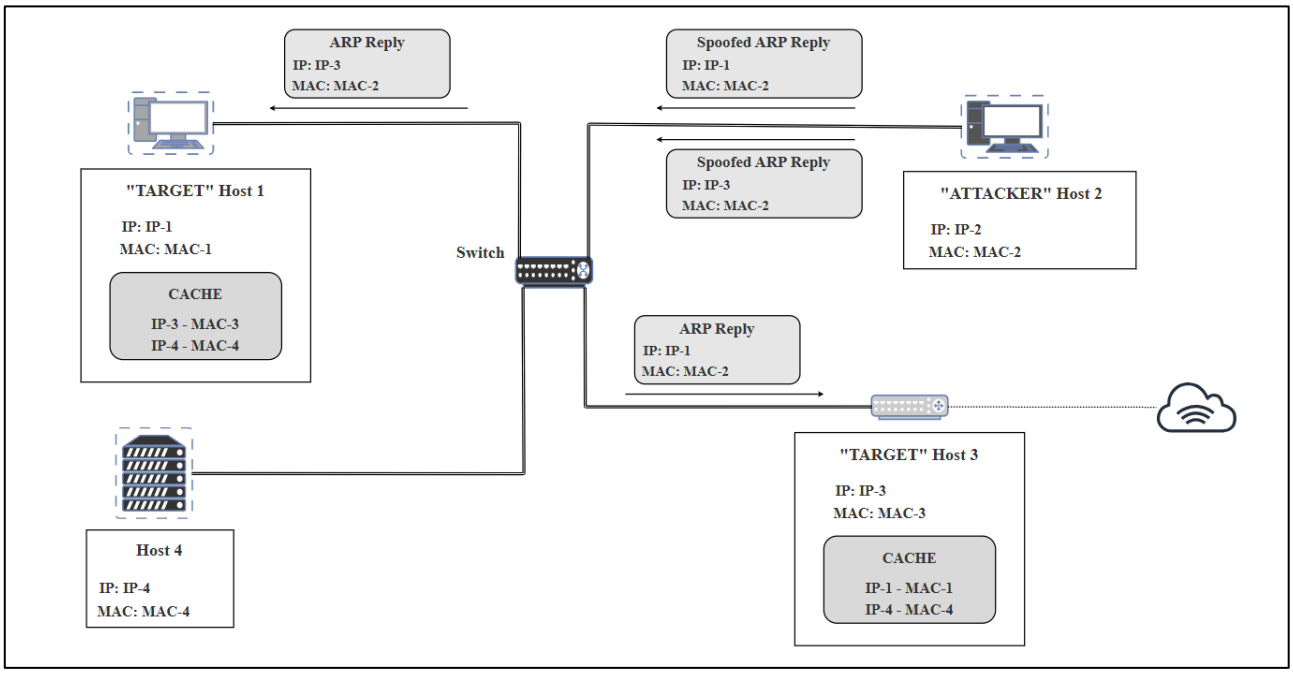


Εικόνα 20. ARP - Reply

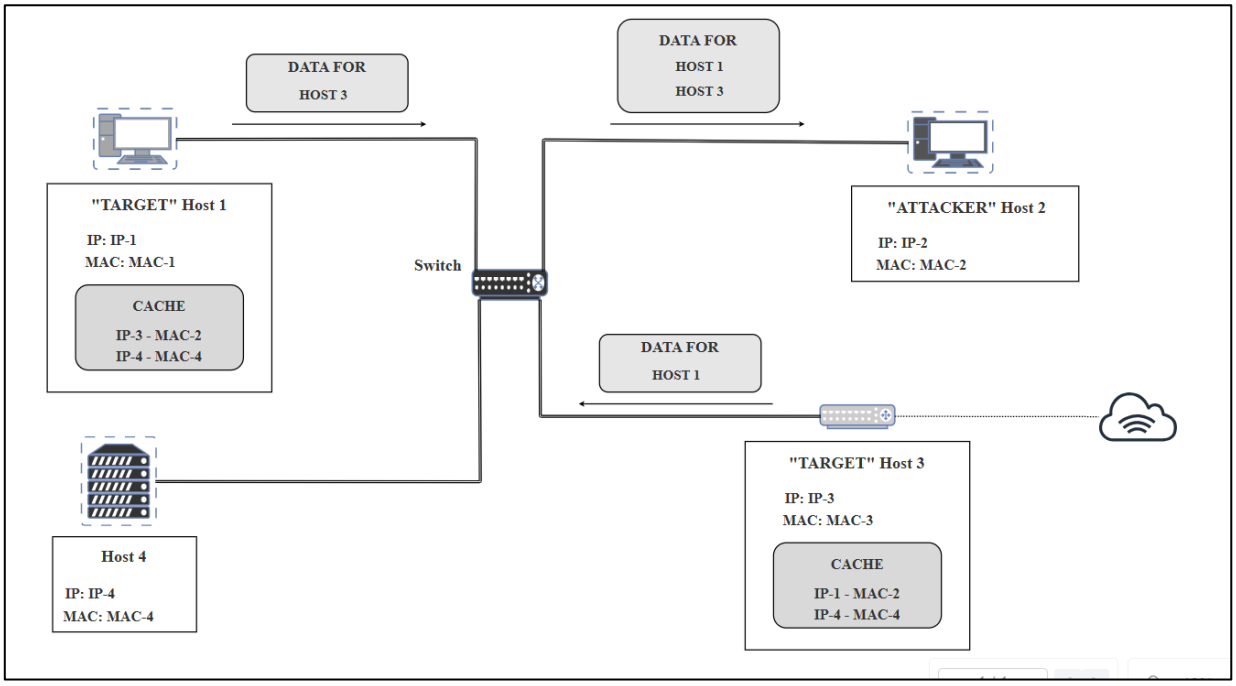
Το γεγονός που καθιστά το πρωτόκολλο ARP ευάλωτο σε απειλές είναι η έλλειψη ελέγχου της ταυτότητας του αποστολέα ενός ARP Reply, καθώς ένας τέτοιος μηχανισμός θα επιβράδυνε σημαντικά τη διαδικασία. Ως εκ τούτου, οι επιτιθέμενοι στέλνουν μη ζητηθείσες ARP Replies με σκοπό να συσχετίσουν τη διεύθυνση MAC τους με την IP μιας άλλης συσκευής. Μόλις ένας Host παραλάβει το παραπάνω παραποιημένο (spoofed) πακέτο θα ανανεώσει την cache του με τα ψευδή στοιχεία, με αποτέλεσμα να στέλνει τα δεδομένα του στον επιτιθέμενο και όχι στον επιθυμητό παραλήπτη. Η επίθεση αυτή ονομάζεται ARP Spoofing (ή ARP Poisoning) και ανήκει στην ευρύτερη κατηγορία των Man in the Middle Attacks, κατά τις οποίες ο επιτιθέμενος "μπαίνει στη μέση" των ροών επικοινωνίας στο δίκτυο και υποκλέπτει την κυκλοφορία. Από εκείνο το σημείο, το κακόβουλο σύστημα είναι σε θέση να αποσπάσει ευαίσθητες πληροφορίες από μη κρυπτογραφημένα δεδομένα (ή να προσπαθήσει να τα αποκρυπτογραφήσει), να τροποποιήσει την κυκλοφορία εισάγοντας κακόβουλο περιεχόμενο, και τέλος να τα την αποστείλει στον προβλεπόμενο παραλήπτη προκειμένου να μην προξενήσει υποψίες στους χρήστες. Οι επιθέσεις ARP Spoof διακρίνονται σε δύο κατηγορίες: Half και Full Duplex.

Full Duplex. Ο επιτιθέμενος περιορίζει την επίθεση σε δύο θύματα με σκοπό να επιτελέσει αμφίδρομη υποκλοπή δεδομένων. Ταυτίζει τη MAC του με δύο IP του δικτύου και στη συνέχεια στέλνει ARP Replies και στις δύο συσκευές με τις παραπάνω διευθύνσεις. Μόλις οι μνήμες των δύο θυμάτων συσχετίσουν την κακόβουλη MAC με την κατάλληλη IP, τότε ο επιτιθέμενος θα είναι σε θέση να υποκλέπτει τα πακέτα που προέρχονται από οποιαδήποτε από τις δύο συσκευές. Αποτελεί τον δημοφιλέστερο τρόπο εξαπόλυσης της συγκεκριμένης επίθεσης καθώς σε αυτό το σενάριο παρακολουθείται αμφίδρομα η επικοινωνία μεταξύ των συστημάτων. Η διαδικασία της παρουσιάζεται στην Εικόνα 21, ενώ στην 22 παρατηρείται η αποστολή των δεδομένων στην κακόβουλη συσκευή εξαιτίας των “δηλητηριασμένων” ARP μνημών.

Half Duplex. Ο επιτιθέμενος χρησιμοποιεί μια διεύθυνση IP προκειμένου να την ταυτίσει με τη δική του MAC και ύστερα στέλνει σε έναν ή περισσότερους Hosts Spoofed πακέτα ARP Reply. Έτσι θα είναι σε θέση να παρακολουθεί μονομερώς την επικοινωνία των Hosts που έλαβαν τα παραπάνω πακέτα, με τη συσκευή της οποίας η διεύθυνση IP εκλάπη. Δεν προτιμάται, καθώς η παρακολούθηση και η υποκλοπή βασικών πληροφοριών στοιχειοθετείται παρακολουθώντας και τα δύο μέρη της “συζήτησης”.



Εικόνα 21. ARP Spoof Full Duplex [1/2]



Εικόνα 22. ARP Spoof Full Duplex [2/2]

SSH PASSWORD AUTHENTICATION BRUTE FORCE ATTACK

Το SSH (Secure Shell) αποτελεί πρωτόκολλο δικτύου που παρέχει έναν ασφαλές τρόπο πρόσβασης και διαχείρισης απομακρυσμένων συστημάτων, πάνω σε ένα μη ασφαλές δίκτυο. Τα δεδομένα που ανταλλάσσονται μεταξύ του πελάτη (SSH Client) και του διακομιστή (SSH Server) κατά τη διάρκεια μιας συνεδρίας είναι κρυπτογραφημένα και προστατεύονται από υποκλοπές. Ο πρωταρχικός λόγος χρήσης του πρωτοκόλλου είναι η δυνατότητα Remote Command Execution, όπου επιτρέπεται στον Client να εκτελεί εντολές για το σύστημα του Server, σαν να υπήρχε η φυσική παρουσία του στη συσκευή του διακομιστή. Ο δεύτερος βασικός λόγος χρήσης του SSH, είναι η ασφαλής μεταφορά αρχείων μέσω της δυνατότητας κρυπτογράφησης. Δημοφιλή εργαλεία, όπως το SCP (Secure Copy) χρησιμοποιούν το SSH για να στείλουν αρχεία από τη μια συσκευή στην άλλη. Ένας Server αντικατοπτρίζει οποιαδήποτε συσκευή υλικού που έχει τη δυνατότητα να εκτελεί λογισμικό διακομιστή SSH (πχ. OpenSSH) και είναι ρυθμισμένος έτσι ώστε να

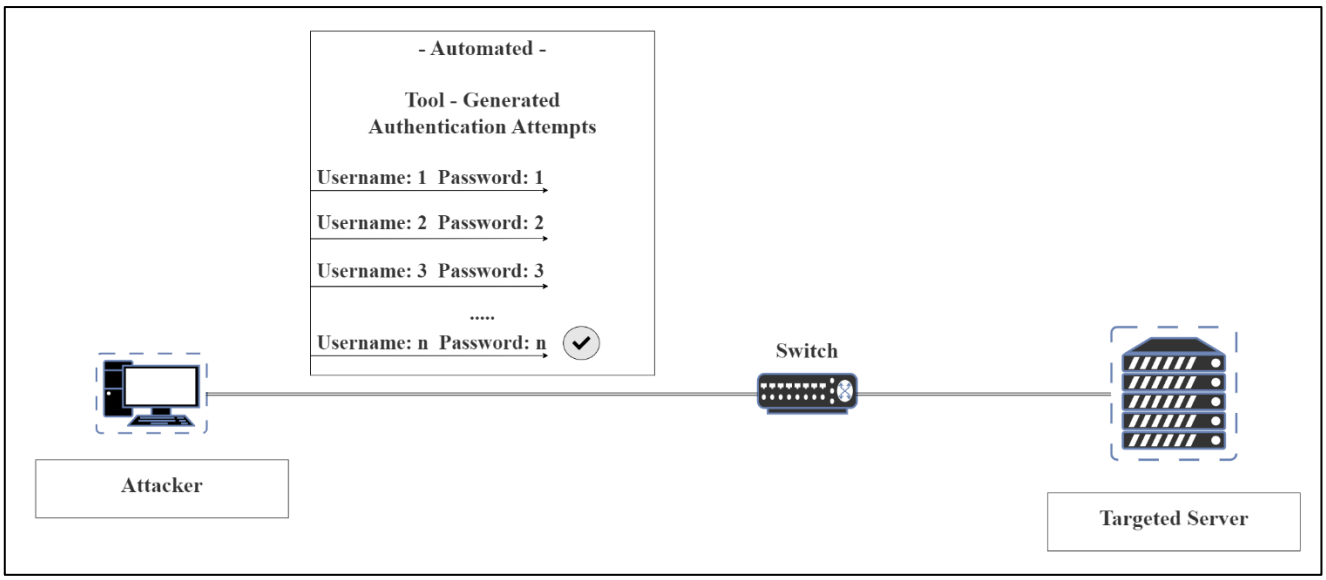
"ακούει" και να δέχεται αντίστοιχες συνδέσεις. Η σύνδεση αυτή συνήθως γίνεται μέσω TCP στη θύρα 22, μόνο από εξουσιοδοτημένο προσωπικό, το οποίο ταυτοποιείται με τους δύο παρακάτω τρόπους:

Password Authentication. Ο διακομιστής χρησιμοποιεί κωδικούς πρόσβασης που καταχωρήθηκαν από τον πελάτη για να επαληθεύσει την ταυτότητα του. By Default, στα εργαλεία λογισμικού SSH δεν υπάρχει θεωρητικό όριο στις προσπάθειες εισαγωγής κωδικού από τη μεριά του Client, μέχρι αυτός να αποκλειστεί από τη διαδικασία.

Key – Based Authentication. Ο διαχειριστής των συστημάτων δημιουργεί ένα ζευγάρι κλειδιών σε κάθε Client συσκευή που θέλει να ταυτοποιείται με αυτόν τον τρόπο, ένα Δημόσιο (Public) και ένα Ιδιωτικό (Private) Κλειδί. Ενώ το Private Key παραμένει στις συσκευές των πελατών, τα Δημόσια Κλειδιά μεταφέρονται σε κατάλληλο αρχείο του διακομιστή, και κάθε φορά που κάποιος Client επιθυμεί να συνδεθεί μέσω SSH χρησιμοποιεί το δικό του ιδιωτικό κλειδί σε συνδυασμό με αυτό που στάλθηκε, παρακάμπτοντας τελείως το βήμα εισαγωγής κωδικού (τεχνολογία Challenge Response [24]).

Είναι λοιπόν προφανές ότι η μέθοδος της εισαγωγής κωδικών είναι ευάλωτη σε μια ενδεχόμενη επίθεση Brute Force, όπου ο κακόβουλος χρήστης με τη βοήθεια κατάλληλου αυτοματοποιημένου εργαλείου προσπαθεί να μαντέψει τον κωδικό πρόσβασης δοκιμάζοντας κάθε φορά έναν διαφορετικό. Σε μια συνηθισμένη επίθεση τέτοιου είδους, τα περισσότερα εργαλεία χρησιμοποιούν όλους δυνατούς συνδυασμούς χαρακτήρων, αριθμών και συμβόλων, μέχρι να βρεθεί ο σωστός. Ωστόσο, η μέθοδος αυτή για μεγαλύτερου μήκους κωδικούς μπορεί αποδειχθεί μάταια, καθώς βασιζόμενοι απλά σε τυχαίες ακολουθίες τα κακόβουλα συστήματα πρέπει να δαπανήσουν χρόνο (μήνες ή ακόμα και έτη) και υπολογιστικούς πόρους για να μαντέψουν έναν απαιτητικό κωδικό. Ωστόσο, όσον αφορά τη δημιουργία νέων κωδικών πρόσβασης, πολλές φορές παρατηρούνται ξεκάθαρα μοτίβα προβλέψιμων ακολουθιών ή προσωπικών πληροφοριών. Οι άνθρωποι συχνά επιλέγουν κωδικούς που τους είναι εύκολο να θυμούνται, όπως ονόματα, ημερομηνίες γέννησης ή κοινές φράσεις, προσθέτοντας συνήθως ακολουθίες χαρακτήρων όπως "123" ή "!". Οι επιτιθέμενοι εκμεταλλεύονται τα προβλέψιμα αυτά μοτίβα και εξαπολύουν Dictionary Brute force Attacks όπου τα αντίστοιχα εργαλεία με παραπάνω χρησιμοποιούν βιβλιοθήκες πιθανών κωδικών πρόσβασης με συνδυασμούς των εν λόγω μοτίβων καθώς και παραλλαγών τους. Αυτή η

προσέγγιση μειώνει δραματικά τον χρόνο που απαιτείται για να ταυτοποιηθεί ένας κωδικό πρόσβασης σε σύγκριση με την παραπάνω ειδικά αν ο κωδικός πρόσβασης του στόχου ακολουθεί κοινά μοτίβα που έχουν δημιουργηθεί από άνθρωπο. Ενδεικτικά, στην παρακάτω εικόνα παρουσιάζεται η διαδικασία Brute Force σε μια υπηρεσία διακομιστή που απαιτεί Username και Password για επιτυχημένη ταυτοποίηση.



Εικόνα 23. Brute Force Attack

DHCP SPOOFING

Το DHCP (Dynamic Host Configuration Protocol) αποτελεί πρωτόκολλο διαχείρισης δικτύου που χρησιμοποιείται για την αυτόματη ανάθεση διευθύνσεων IP και άλλων απαραίτητων λεπτομερειών δικτύωσης μιας συσκευής. Οι εν λόγω λεπτομέρειες περιλαμβάνουν μεταξύ άλλων, την ρύθμιση μάσκας (Subnet Mask) καθώς και των διευθύνσεων του προκαθορισμένου δρομολογητή (Default Gateway) και του διακομιστή DNS. Με την αυτοματοποίηση αυτής της διαδικασίας, το DHCP διευκολύνει τη διαχείριση των διευθύνσεων IP στα δίκτυα, μειώνοντας την ανάγκη για χειροκίνητη διαμόρφωσή τους σε κάθε συσκευή. Η διαδικασία DHCP αποτελείται από την ακολουθία τεσσάρων βημάτων. Discover - Offer - Request - Acknowledge (ή εν συντομία DORA). Σημειώνεται ότι ένας DHCP Server μπορεί να αναπαρασταθεί από

οποιαδήποτε υλική συσκευή η οποία φέρει κατάλληλο λογισμικό ανάθεσης των παραπάνω διευθύνσεων, σε οποιοδήποτε Client του δικτύου τις ζητήσει.

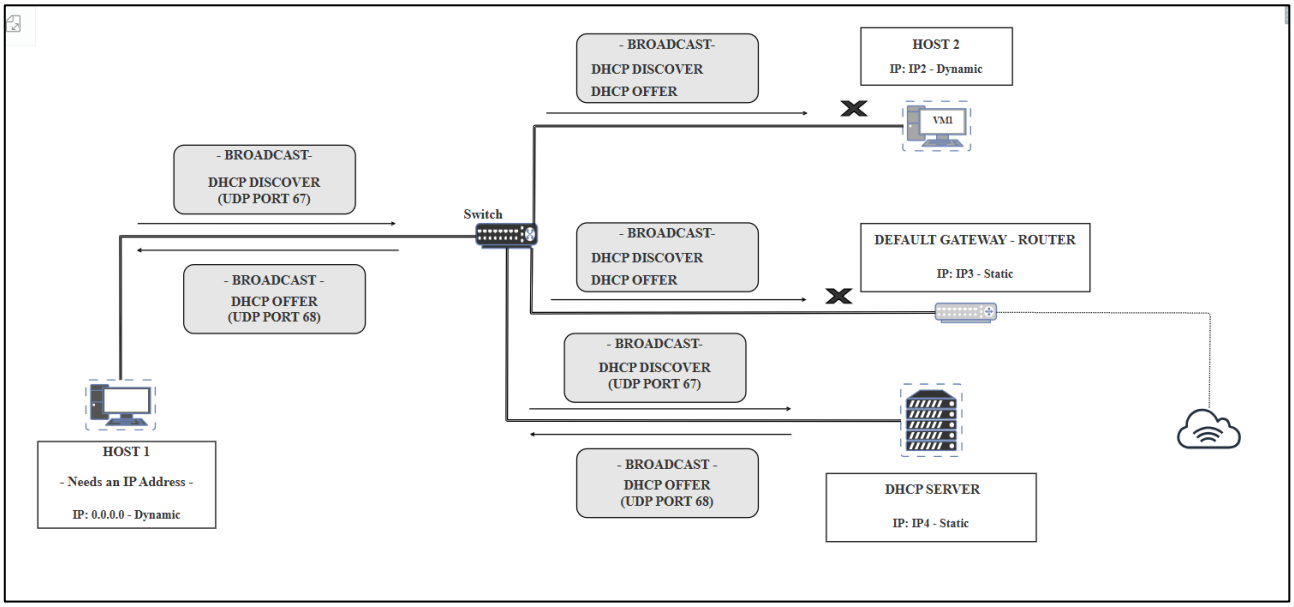
Discover. Τη στιγμή που μια συσκευή (DHCP Client) συνδέεται σε ένα δίκτυο, εκπέμπει αμέσως ένα πακέτο DHCP Discover σε όλες τις υπόλοιπες, προκειμένου να δηλώσει την αναζήτηση κατάλληλου διακομιστή, ο οποίος θα της εκχωρήσει τις απαραίτητες διευθύνσεις. Το πακέτο αυτό στέλνεται μέσω UDP (User Datagram Protocol) και συγκεκριμένα από το Port 67. Όταν φτάνει στις υπόλοιπες συσκευές του δικτύου αγνοείται από όλες εκτός από τους DHCP Servers.

Offer. Όταν ένας διακομιστής DHCP λάβει το παραπάνω πακέτο, απαντάει με αντίστοιχο Broadcast DHCP Offer, προσφέροντας ουσιαστικά στη συσκευή του πελάτη ένα σύνολο όλων των απαραίτητων δεδομένων, όπως θεσπίζεται από το πρωτόκολλο. Συγκεκριμένα, λαμβάνει το πακέτο Discover στη θύρα 67 και στέλνει το κατάλληλο Offer διευθύνσεων από την 68.

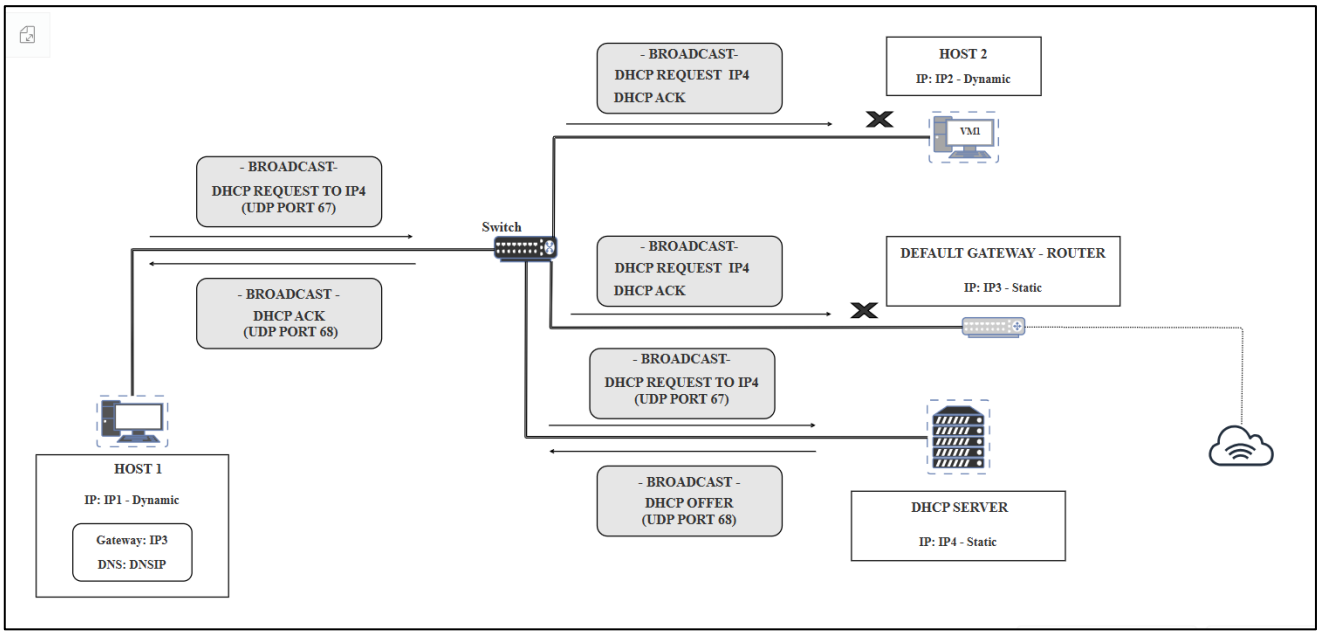
Request. Αφού λάβει την προσφορά στο Port 68, ο Client απαντά με εκπομπή πακέτου DHCP Request, το οποίο αντικατοπτρίζει την επίσημη αίτηση για αποδοχή και τελική ανάθεση των διευθύνσεων που προτάθηκαν από τον διακομιστή. Εάν προκύψουν παραπάνω από μια προσφορές (δηλαδή υπάρχουν δύο ή περισσότεροι διακομιστές στο δίκτυο) ο πελάτης συνήθως επιλέγει αυτή που θα καταφθάσει πρώτη, καθώς το πρωτόκολλο DHCP εκ φύσεως βασίζεται στην ταχύτητα ολοκλήρωσής του. Η διεύθυνση του διακομιστή που προτιμήθηκε βρίσκεται μέσα στο πακέτο Request, συνεπώς οι υπόλοιποι DHCP Servers του δικτύου που θα παραλάβουν το μήνυμα γνωρίζουν ότι πρέπει να αγνοήσουν τη συνέχεια της διαδικασίας.

Acknowledge. Στην τελική φάση του πρωτοκόλλου, ο διακομιστής που έκανε την προσφορά και έλαβε πίσω την αίτηση για ανάθεση διευθύνσεων από αυτόν, εκπέμπει κατάλληλο πακέτο επιβεβαίωσης της εν λόγω διαδικασίας. Το πακέτο λοιπόν DHCP ACK, οριστικοποιεί την εκχώρηση των διευθύνσεων, και μόλις αυτό καταφτάσει στον Client, τότε ο τελευταίος θα είναι σε θέση να τις χρησιμοποιήσει. Η χρήση των διευθύνσεων αυτών περιορίζεται χρονικά από το DHCP Lease, το οποίο είναι προκαθορισμένης διάρκειας και όταν λήξει επιβάλλεται η επανάληψη των βημάτων Request και Acknowledge για την ανανέωσή του.

Παρακάτω παρατίθεται η διαδικασία σε ενδεικτικό δίκτυο. Ο HOST 1 ζητάει ανάθεση διευθύνσεων από τον DHCP SERVER και λαμβάνει IP για τον ίδιο (IP1), το Default Gateway (IP3) καθώς και το DNS Server (DNSIP) του δικτύου.

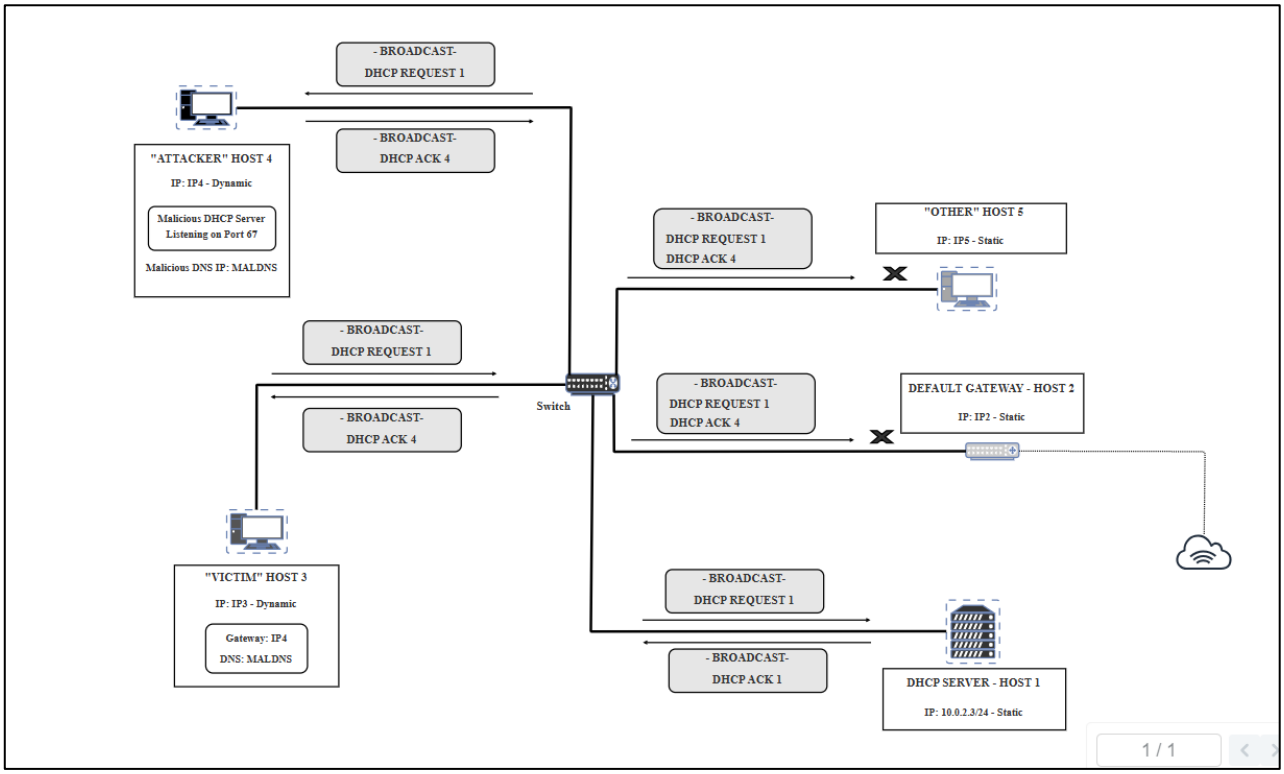


Εικόνα 24. DHCP Discover / Offer



Εικόνα 25. DHCP Request / Acknowledgement

Είναι εύκολο να παρατηρηθεί λοιπόν ότι η έλλειψη μηχανισμού επαλήθευσης της ταυτότητας των διακομιστών αποτελεί εν δυνάμει ευπάθεια στο συγκεκριμένο πρωτόκολλο. Σε μια επίθεση DHCP Spoofing, ο κακόβουλος χρήστης μπορεί να εγκαταστήσει έναν παράτυπο DHCP Server στο δίκτυο, ο οποίος θα ανταποκρίνεται σε Discover μηνύματα πιθανών συστημάτων - στόχων, απαντώντας με κακόβουλα Offers. Αυτά θα περιέχουν μια τυχαία αχρησιμοποίητη IP του δικτύου προκειμένου να ανατεθεί στο θύμα, σε συνδυασμό με τη διεύθυνση του επιτιθέμενου ως Default Gateway. Εάν το σύστημα – στόχος επιλέξει την κακόβουλη προσφορά, ο επιτιθέμενος θα έχει επιτύχει άλλη μια Man In the Middle Attack, μεταξύ του Router και του θύματος, εφόσον αυτό πλέον γνωρίζει ότι η διεύθυνση του Default Gateway είναι η κακόβουλη IP. Παράλληλα, το DHCP πακέτο αναθέτει και τις διευθύνσεις των DNS Server στα συστήματα, συνεπώς εγκαθιστώντας τους δικούς του, κακόβουλους διακομιστές DNS, ο επιτιθέμενος είναι σε θέση να διευρύνει περεταίρω το ρεπερτόριο των επιθέσεων που μπορεί να εκτελέσει (DNS Hijacking [25]). Όπως εξηγήθηκε και παραπάνω, η επιτυχία της επίθεσης βασίζεται κυρίως στη ταχύτερη αποστολή των κακόβουλων πακέτων DHCP Offer στο θύμα, από τον προκαθορισμένο διακομιστή του δικτύου. Ωστόσο, η επίθεση μπορεί να επιτευχθεί ακόμα και αν δεν επιλεχθεί το κακόβουλο πακέτο διευθύνσεων, εάν ο παράτυπος Server υπερβεί του προκαθορισμένου, στην αποστολή του Acknowledgement πακέτου. Και αυτό συμβαίνει διότι στο τελευταίο στάδιο της διαδικασίας όπου ο Client δέχεται το DHCP – ACK, δεν ελέγχει την διεύθυνση του αποστολέα και ορίζει στο σύστημά του τις διευθύνσεις που φέρει το εν λόγω πακέτο. Συνεπώς, ο επιτιθέμενος έχει δύο διακριτά χρονικά περιθώρια, προκειμένου να διεισδύσει στο σύστημα – θύμα. Το δεύτερο σκέλος της προσπάθειας αυτής αποτυπώνεται στην Εικόνα 26, για μια ενδεικτική τοπολογία δικτύου. Σημειώνεται ότι η επίθεση DHCP Starvation Attack [26] που συνηθίζεται να συνδυάζεται με την παραπάνω, δεν αναλύθηκε στην παρούσα εργασία, καθώς ένας απλός μηχανισμός ειδοποιήσεων όταν το DHCP Pool των διευθύνσεων εξαντλείται, λύνει εύκολα το πρόβλημα.



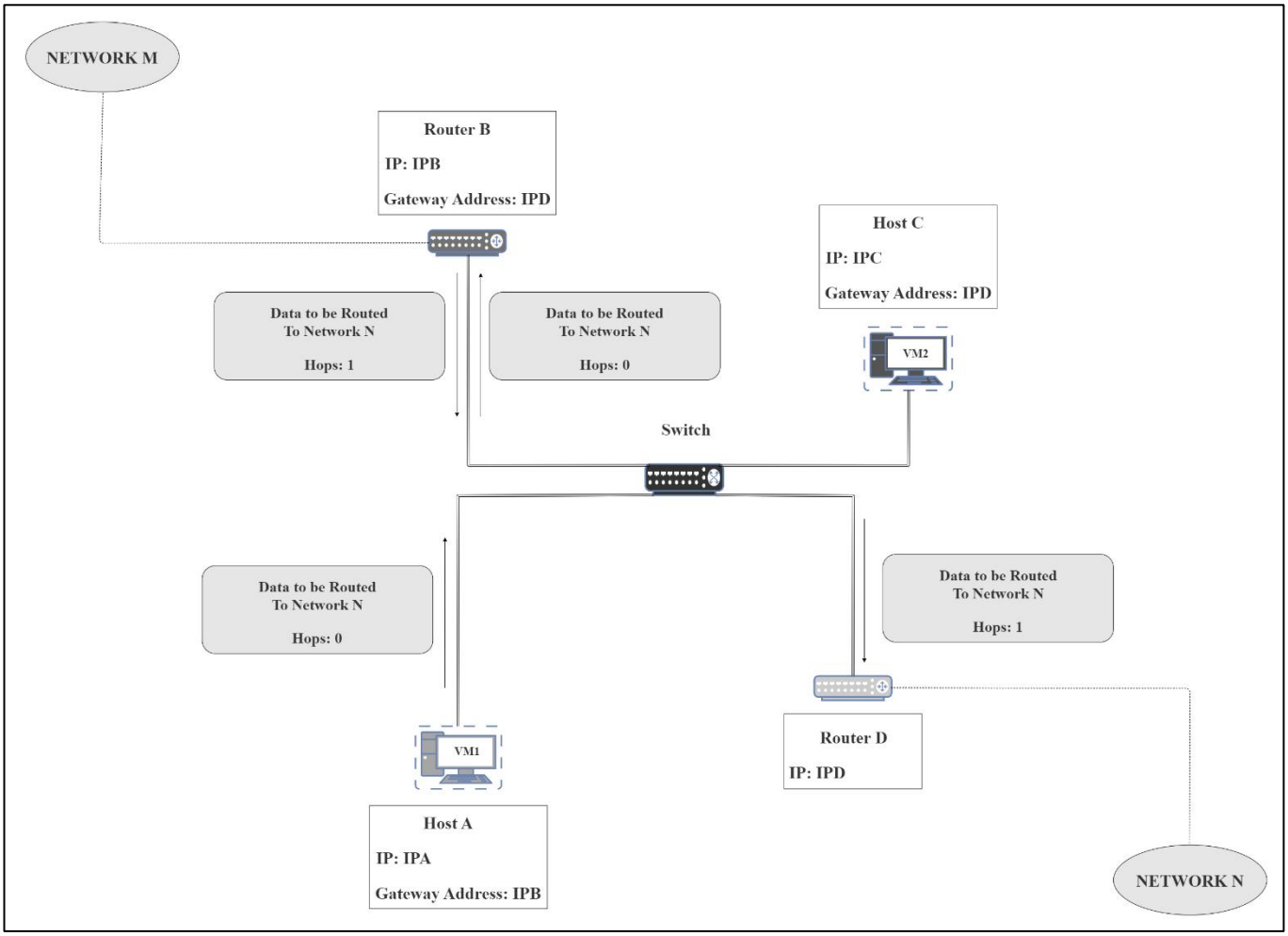
Εικόνα 26. Επισκόπηση DHCP Spoofing

ICMP REDIRECTION ATTACK

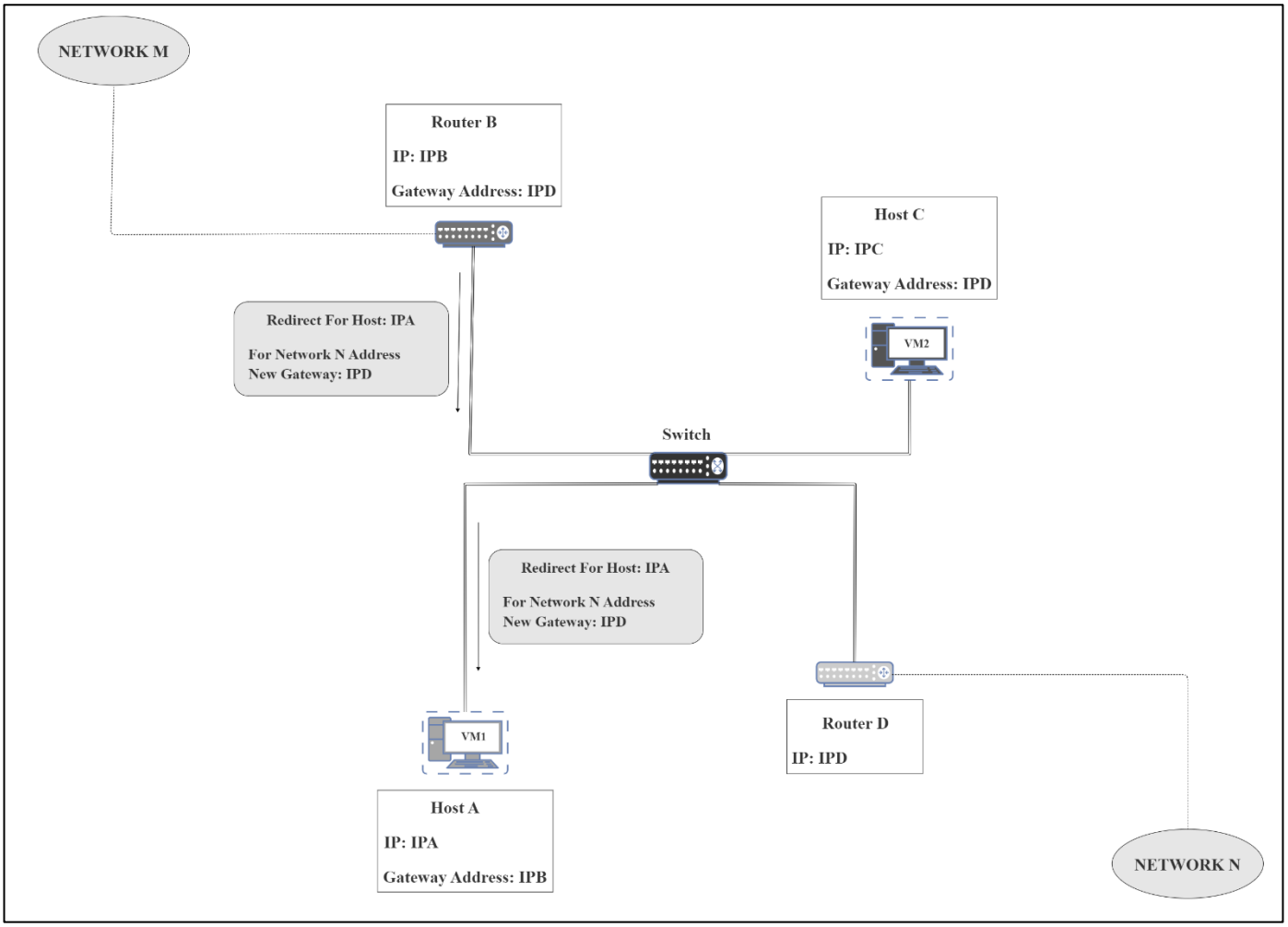
Το ICMP (Internet Control Message Protocol) αποτελεί πρωτόκολλο επιπέδου 3 (Network Layer), και χρησιμοποιείται για την αποστολή μηνυμάτων σφάλματος, λειτουργικών πληροφοριών και διαγνωστικών δεδομένων σχετικά με την επικοινωνία μεταξύ συσκευών. Σε αντίθεση με το TCP ή το UDP, το ICMP δεν χρησιμοποιείται για την ανταλλαγή δεδομένων μεταξύ εφαρμογών, αλλά υποστηρίζει την ορθή λειτουργία των δικτύων, αναφέροντας τα σχετικά προβλήματα μέσω διαγνωστικών ελέγχων, και βοηθώντας στην επίλυσή τους. Τα πακέτα ανακατεύθυνσης λοιπόν (Redirect), αποτελούν συγκεκριμένο τύπο του πρωτοκόλλου και χρησιμοποιούνται από τους δρομολογητές για να ενημερώνουν τους Hosts όταν υπάρχει μια πιο αποτελεσματική διαδρομή δρομολόγησης πακέτων προς ένα προορισμό.

Όταν ένας δρομολογητής λαμβάνει ένα πακέτο από κάποιον αποστολέα, το οποίο πρέπει πρώτα να προωθηθεί σε έναν άλλο δρομολογητή ενός του ίδιου υποδικτύου, αντί απευθείας στον προορισμό του, τότε διαπιστώνει το πλεονασματικό βήμα στη διαδρομή και στέλνει πίσω στον αποστολέα ένα πακέτο

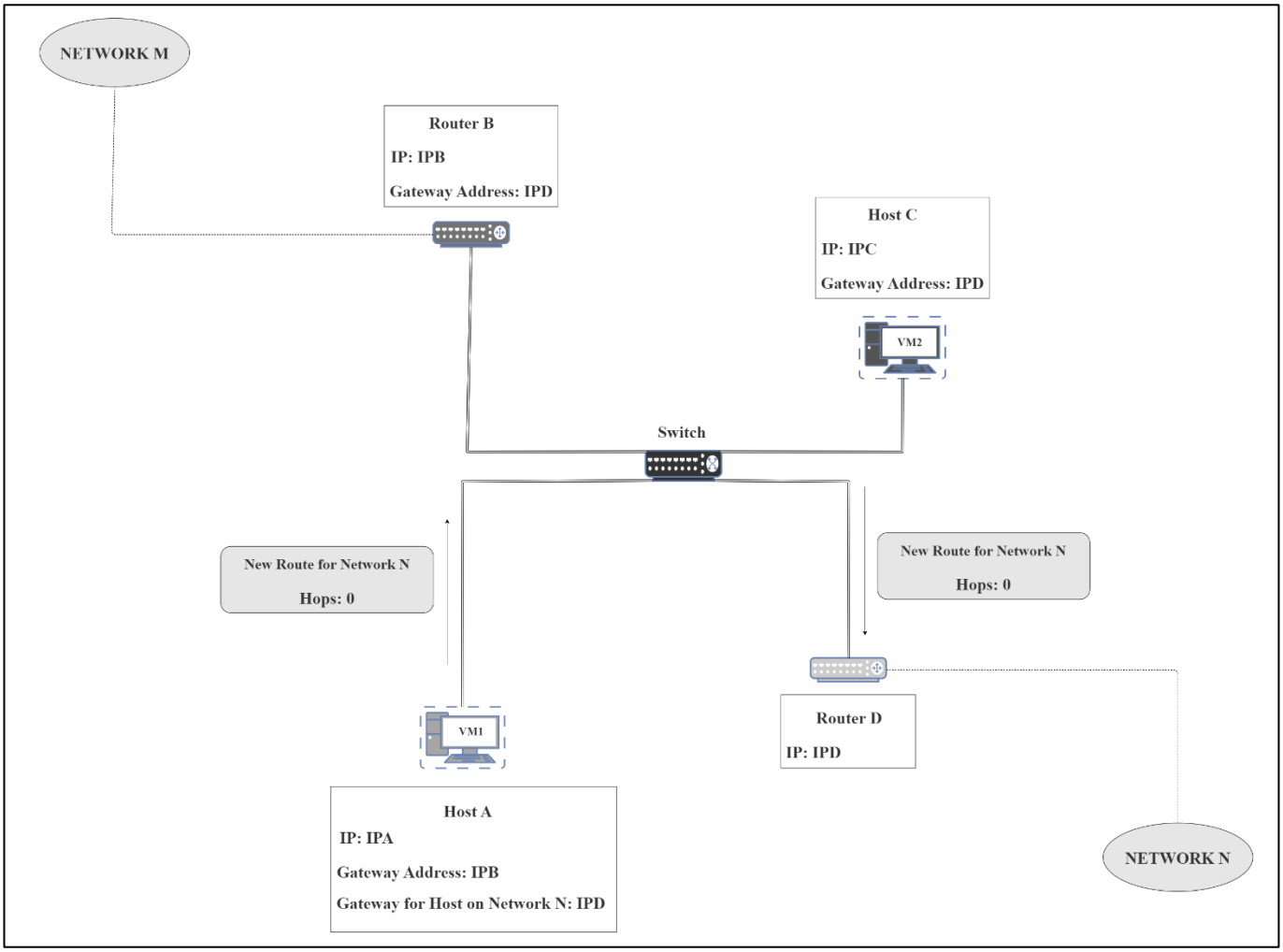
ICMP Redirect, πρώτου προωθήσει το προηγούμενο πακέτο στο δεύτερο δρομολογητή. Το Redirect ενημερώνει τον Host ότι πρέπει να ανανεώσει τον πίνακα δρομολόγησής του, προκειμένου να χρησιμοποιείται η αποτελεσματικότερη διαδρομή πακέτων για οποιαδήποτε μελλοντική προσπάθεια επικοινωνίας με τον ίδιο προορισμό. Ύστερα από την παραλαβή του πακέτου Redirection, ο Host ανανεώνει το Routing Table του συστήματος του και χρησιμοποιεί πλέον τον δεύτερο δρομολογητή για τον παραπάνω προορισμό. Υπενθυμίζεται ότι ένας πίνακας δρομολόγησης (Routing Table) είναι δομή δεδομένων που υπάρχει σε όλους τους Hosts, και περιέχει πληροφορίες σχετικά με τις διαδρομές προς διάφορους δικτυακούς προορισμούς, όπως συγκεκριμένες διευθύνσεις IP ή υποδίκων. Συγκεκριμένα, καταγράφονται οι διευθύνσεις των προεπιλεγμένων πυλών (Default Gateway) που χρησιμοποιούνται ως Next – Hop για την δρομολόγηση στους προαναφερόμενους δικτυακούς προορισμούς. Η αναπαράσταση της διαδικασίας παρουσιάζεται στις παρακάτω Εικόνες. Συγκεκριμένα στην 27 φαίνεται το πλεονασματικό βήμα (Hop) της διαδρομής των πακέτων από τον Host A, προς μια διεύθυνση του Network N, ενώ στην 28 η αποστολή του πακέτου ICMP Redirect (For Host) του Router B προς τον Host A. Τέλος, παρουσιάζεται η νέα διαδρομή των πακέτων προς τον προορισμό που βρίσκεται στο Network N, με Gateway αυτή τη φορά το Router D.



Εικόνα 27. Πλεονασματική Διαδρομή Δεδομένων



Εικόνα 28. Αποστολή Πακέτου ICMP Redirect



Εικόνα 29. Νέα Διαδρομή Πακέτων

Όπως τα πρωτόκολλα ARP και DHCP, έτσι και αυτό εμφανίζει ευπάθειες όσον αφορά την έλλειψη μηχανισμού πιστοποίησης του δρομολογητή καθώς στην παραπάνω διαδικασία θεωρείται δεδομένη η εμπιστοσύνη μεταξύ του Host και του Router. Σε μια επίθεση ICMP Redirection, ο κακόβουλος Host στέλνει ψεύτικα πακέτα ανακατεύθυνσης ICMP σε ένα θύμα, εξαναγκάζοντάς το να ενημερώσει τον πίνακα δρομολόγησής του με αποτέλεσμα να χρησιμοποιεί την IP του επιτιθέμενου ως προεπιλεγμένη πύλη για συγκεκριμένους προορισμούς. Γίνεται λόγος δηλαδή για άλλη μια έκφανση των Man In the Middle Attacks, τα αποτελέσματα της οποίας είναι γνωστά και συζητήθηκαν παραπάνω (Αναχαίτιση / Τροποποίηση / Αποκλεισμός της κυκλοφορίας ή περεταίρω ανακατεύθυνση σε κακόβουλους προορισμούς).

DENIAL OF SERVICE – DOS ATTACKS

Η επίθεση άρνησης παροχής υπηρεσιών (DoS) αποσκοπεί στο να καταστήσει έναν στόχο (π.χ. διακομιστή, δίκτυο ή εφαρμογή) μη διαθέσιμο στους νόμιμους χρήστες, κατακλύζοντάς τον με υπερβολική κίνηση, για κατανάλωση πόρων ή εκμετάλλευση ευπαθειών. Πρωταρχικός σκοπός τους αποτελεί η διακοπή των υπηρεσιών που παρέχονται από το θύμα, προκαλώντας αναταραχή στο σύστημα και πλήρη υποβάθμιση των επιδόσεων. Λόγω της μεγάλης γκάμας υποκατηγοριών της επίθεσης, επιλέχθηκαν τρεις βασικοί τύποι "πλημυρών" με βάση τον αντίστοιχο αριθμό πρωτοκόλλων που εκμεταλλεύονται.

1) **TCP - SYN Flood** Η επίθεση αυτή εκμεταλλεύεται την προαναφερθείσα τριμελή χειραψία TCP Handshake, όταν αυτή χρησιμοποιείται για την παροχή υπηρεσιών Web. Ο επιτιθέμενος λοιπόν στέλνει έναν μεγάλο αριθμό SYN πακέτων προς τοπικό Apache Server (διακομιστής - θύμα), ζητώντας τη δημιουργία συνδέσεων. Ο Server με τη σειρά του αναλύει τα πακέτα και απαντά με αντίστοιχα SYN-ACK, εξαντλώντας σταδιακά τους υπολογιστικούς πόρους και το διαθέσιμο εύρος ζώνης.

2) **ICMP - Ping Request Flood** Σε αυτό το σενάριο, ο επιτιθέμενος κατακλύζει το σύστημα στόχο με πακέτα Ping Request, αναγκάζοντας το να απαντήσει με τα προβλεπόμενα Ping Replies. Σκοπός της επίθεσης παραμένει η εξαθλίωση πόρων και εύρους ζώνης.

3) **UDP - DNS Request Flood** Εκμεταλλεύεται η δομή Request - Reply που αποκτάει το πρωτόκολλο UDP όταν χρησιμοποιείται για διαδικασίες παροχής υπηρεσιών DNS. Ο επιτιθέμενος "πλημυρίζει" τοπικό διακομιστή Bind9 με πακέτα UDP Request (ή αλλιώς DNS Query) εξαναγκάζοντας τον να τα αναλύσει, και να προβεί σε αποστολή κατάλληλων UDP Reply (ή DNS Querry Response). Τα πακέτα αυτά περιέχουν την απάντηση του ελέγχου για το Domain Name που ζητήθηκε ως προς την IP του, ή κάποιο διαγνωστικό λάθος σε περίπτωση αποτυχίας της αντιστοίχισης. Όπως και στους παραπάνω, έτσι και σε αυτόν τον τύπο επίθεσης αναμένεται η υπερκατανάλωση της CPU του θύματος, καθώς και του εισερχόμενου / εξερχόμενου εύρους ζώνης (Bandwidth).

Evasion Techniques

Οι τεχνικές υπεκφυγής είναι μέθοδοι που χρησιμοποιούν οι επιτιθέμενοι για να αποφύγουν την ανίχνευση από συστήματα ασφαλείας, όπως τείχη προστασίας, συστήματα ανίχνευσης εισβολών (IDS) ή συστήματα πρόληψης εισβολών (IPS). Μια γνωστή τεχνική υπεκφυγής που χρησιμοποιήθηκε στην παρούσα εργασία είναι η χρήση Spoofed IP διευθύνσεων. Ο επιτιθέμενος πλαστογραφεί τη διεύθυνση IP προέλευσης στις επικεφαλίδες των κακόβουλων πακέτων, κάνοντάς τα να φαίνονται σαν να προέρχονται από διαφορετική συσκευή. Μπορούν λοιπόν να κρύψουν την ταυτότητά τους, να παρακάμψουν το φιλτράρισμα βάσει IP ή να ξεγελάσουν τον στόχο ώστε να ανταποκριθεί σε άλλο σύστημα. Εφαρμόζεται σε συνδυασμό με τις επιθέσεις ICMP Redirection και Denial of Service. Επίσης, κατά τη διάρκεια των σαρώσεων με το εργαλείο Nmap χρησιμοποιείται ακόμα μια τεχνική υπεκφυγής, αυτή των Decoy IP Addresses. Σε αυτή τη μέθοδο, οι επιτιθέμενοι σαρώνουν το στόχο από πολλαπλές πλαστές διευθύνσεις (δολώματα) συνδυασμό με την πραγματική τους IP, καθιστώντας δυσκολότερο για τα συστήματα ασφαλείας να εντοπίσουν την πραγματική πηγή της επίθεσης μέσα στην αχανή κυκλοφορία που δημιουργείται από το Scan. Όπως και παραπάνω, οι διευθύνσεις Decoy αναπαριστούν είτε γνωστές, είτε τυχαίες διευθύνσεις του Subnet.

2

ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ

2.1 Δημιουργία Περιβάλλοντος Προσομοίωσης

Στην παρούσα Ενότητα θα πραγματοποιηθεί εκτενής αναφορά στον τρόπο υλοποίησης του συστήματος ασφάλειας που περιγράφηκε στην προηγούμενη, ξεκινώντας από τη δημιουργία του δικτύου που μελετάται. Για την εικονικοποίηση (virtualization) και την προσομοίωση της δράσης του IDS/IPS κατά τη διάρκεια μιας επίθεσης από μια κακόβουλη σε μια προστατευόμενη "μηχανή", χρησιμοποιήθηκε το VirtualBox [27]. Το τελευταίο αποτελεί εφαρμογή εικονικοποίησης που επιτρέπει στους χρήστες να διαχειρίζονται πολλαπλές υπολογιστικές "μηχανές" (Virtual Machines – VMs) από μια μόνο συσκευή υλικού (Host Machine). Το σημαντικότερο χαρακτηριστικό των VMs είναι η δυνατότητα υποστήριξης πολλαπλών ειδών λειτουργικών συστημάτων (OS), θέτοντας έτσι τις βάσεις για τη δημιουργία εικονικών δικτύων προσομοίωσης ανίχνευσης και αντιμετώπισης επιθέσεων. Τα μόνα πρακτικά όρια είναι η CPU, η RAM και ο χώρος Hard Drive που μπορούν να διατεθούν από τους χρήστες.

Στο παρών πείραμα, μέσω συσκευής Host (Laptop τυπικών επιδόσεων) διατέθηκαν οι κατάλληλοι πόροι υλικού και δημιουργήθηκαν τρείς VMs, Attacker, IDS/IPS και Victim1 VM. Μια σύντομη επισκόπηση των μηχανών παρατίθεται στις παρακάτω εικόνες.

Host Machine	
Operating System	Windows 10 Home, Version 22H2 (64 - bit)
CPU	4 Physical Cores
RAM	16 GB
Description	Μοιράζεται με το VirtualBox τους παραπάνω υλικούς πόρους για τη σύσταση του δικτύου

Εικόνα 30. Επισκόπηση του Host Machine

IDS/IPS Virtual Machine	
Operating System	Ubuntu 22.04.4 LTS, Codename jammy (64 - bit)
CPU	2 Physical Cores Dynamically Allocated
RAM	2 GB
Description	Ελέγχει την κίνηση, ανιχνεύει τις επιθέσεις στα προστατευόμενα συστήματα και τις αντιμετωπίζει. Φέρει το λογισμικό IDS/IPS με την τεχνολογία DSD

Εικόνα 31. Επισκόπηση του IDS/IPS Virtual Machine

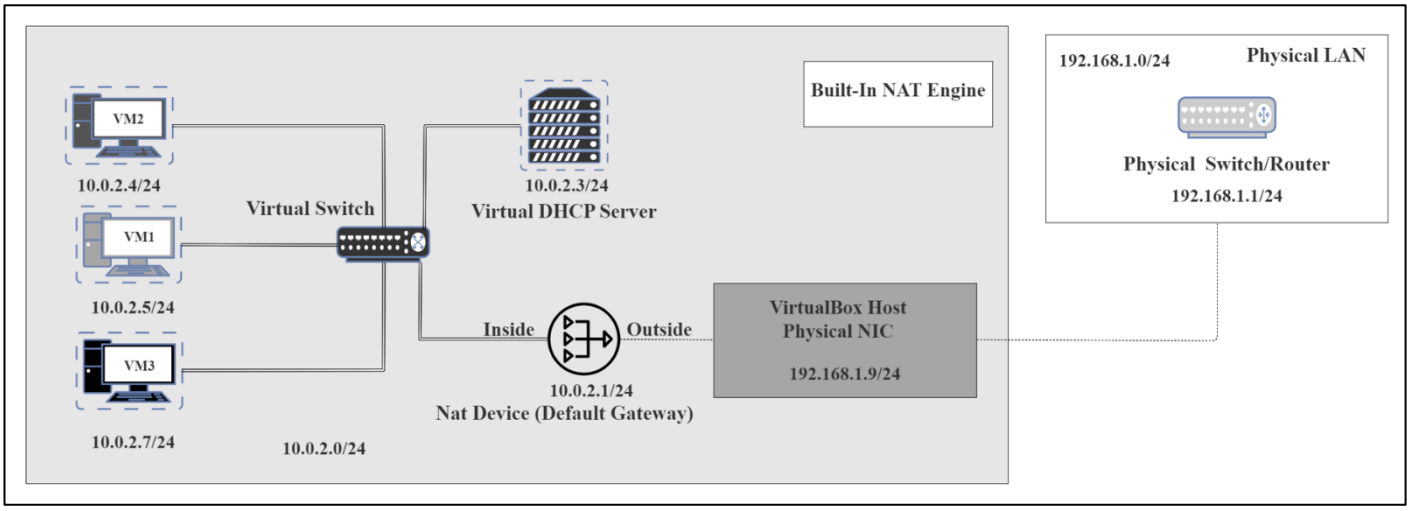
Attacker Virtual Machine	
Operating System	Kali Linux 2024.2, Codename kali - rolling (64 - bit)
CPU	2 Physical Cores Dynamically Allocated
RAM	2 GB
Description	”Κακόβουλη” συσκευή που εκτελεί τις επιθέσεις που αναλύθηκαν εις βάρους προστατευόμενης συσκευής.

Εικόνα 32. Επισκόπηση του Attacker Virtual Machine

Victim1 Virtual Machine	
Operating System	Ubuntu 22.04.4 LTS, Codename jammy (64 - bit)
CPU	2 Physical Cores Dynamically Allocated
RAM	2 GB
Description	Συσκευή που λειτουργεί ως στόχος επιθέσεων, φιλοξενώντας τις κατάλληλες υπηρεσίες και λογισμικό για την προσομοίωση πραγματικών δικτυακών σεναρίων και τη δοκιμή του IDS/IPS.

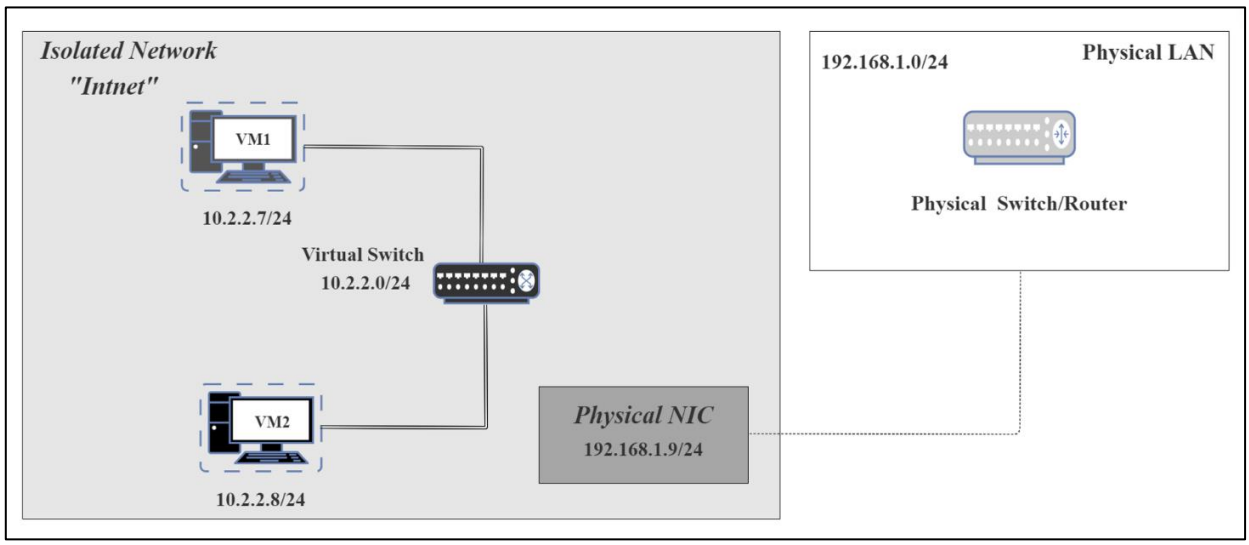
Εικόνα 33. Επισκόπηση Victim1 Virtual Machine

Αφού δημιουργήθηκαν τα 3 παραπάνω VMs πραγματοποιήθηκαν οι απαραίτητες ρυθμίσεις εικονικής δικτύωσης, προκειμένου να ακολουθούνται τα πρότυπα που τέθηκαν στην Εικόνα 11 της προηγούμενης Ενότητας. Όσον αφορά το πρωτεύον δίκτυο, για να προσομοιάζει ένα ρεαλιστικό VLAN, οι εικονικές κάρτες δικτύου (Virtual Interfaces) των παραπάνω Virtual Machine παραμετροποιήθηκαν με ίδια ρύθμιση, NAT Network. Σε κάθε VM εκχωρείται μια ιδιωτική (Private) IP εντός του δικτύου NAT που δημιουργείται, και το VirtualBox χειρίζεται τη "μετάφραση" αυτών στη δημόσια IP του Host Machine, για την επίτευξη εξωτερικής επικοινωνίας. Αυτή η ρύθμιση επιτρέπει στις μηχανές να έχουν πρόσβαση στο διαδίκτυο, απομονώνοντάς τα ταυτόχρονα από την Host Machine και άλλες φυσικές συσκευές. Για την προαναφερθείσα μέθοδο μετάφρασης των ιδιωτικών IP, χρησιμοποιείται ενσωματωμένος VirtualBox δρομολογητής NAT (Network Address Translation). Αυτός μεταφράζει την εσωτερική κυκλοφορία των VMs χρησιμοποιώντας την φυσική κάρτα δικτύου του Host Machine, προκειμένου αυτή εν συνεχεία να προωθηθεί στο διαδίκτυο ή άλλα εξωτερικά δίκτυα. Παράλληλα, στο δίκτυο NAT ενσωματώνεται μεταγωγέας επιπέδου 2, προκειμένου να διευθύνει την επικοινωνία βάσει των διευθύνσεων MAC που αποκτούν τα VMs, καθώς και διακομιστής DHCP (η λειτουργία του εξηγήθηκε παραπάνω). Στην Εικόνα 34 παρατίθεται η αρχιτεκτονική του NAT Network, όπως προκύπτει από την παραπάνω ρύθμιση. Σημειώνεται ότι το δίκτυο αυτό προσομοιάζει το Main Network της Εικόνας 11.



Εικόνα 34. NAT Network.

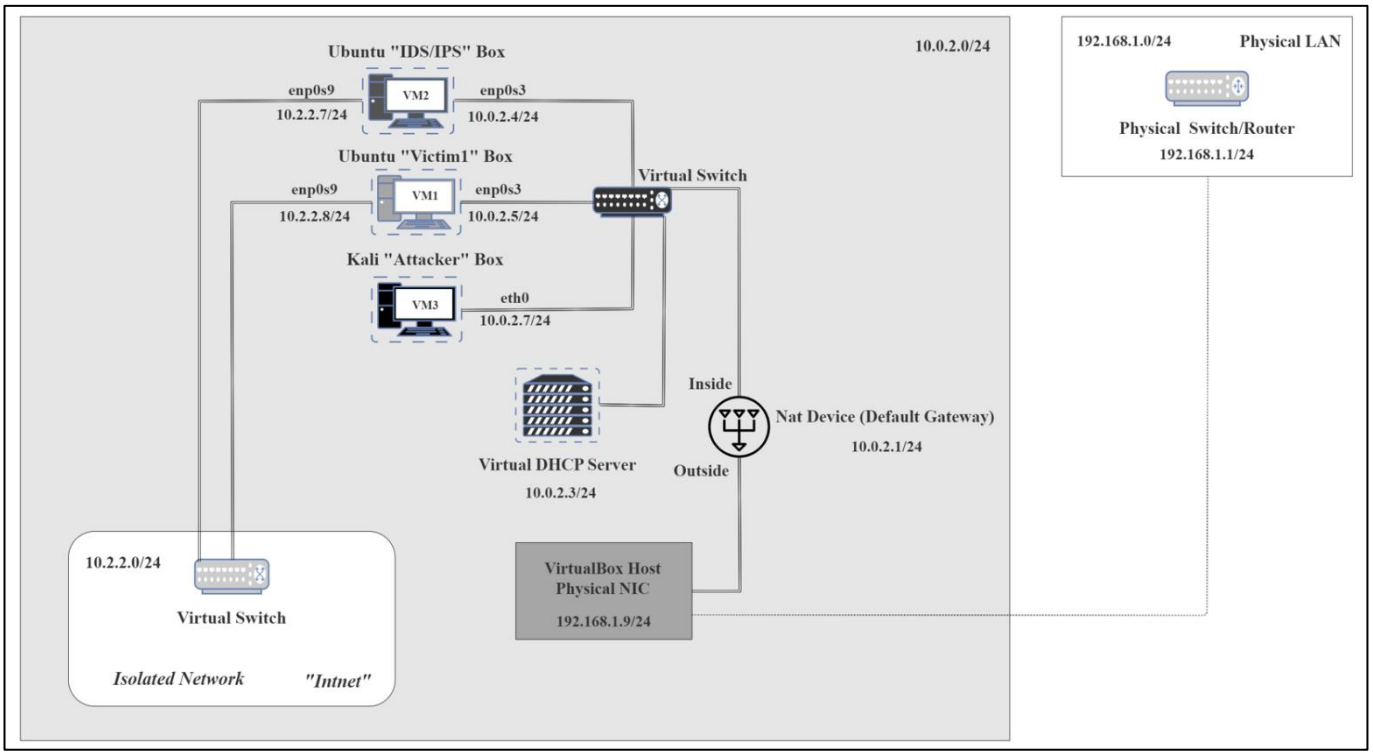
Προχωρώντας στην ενσωμάτωση του ασφαλούς διαύλου επικοινωνίας μεταξύ του IDS/IPS (VM2) και του Victim1 (VM1), δημιουργείται δεύτερο δίκτυο χρησιμοποιώντας αυτή τη φορά τη ρύθμιση Internal Network του VirtualBox, στις δεύτερες εικονικές κάρτες δικτύων των παραπάνω μηχανών. Όπως προκύπτει και από την ονομασία του (Εσωτερικό Δίκτυο), μέσω αυτού οι VM μπορούν να επικοινωνούν μεταξύ τους, αλλά είναι πλήρως απομονωμένες από τη Host Machine, αλλά και από οποιοδήποτε εξωτερικό δίκτυο, συμπεριλαμβανομένου και του παραπάνω. Σημειώνεται ότι στο δίκτυο χρησιμοποιείται μόνο ένας ενσωματωμένος μεταγωγέας επιπέδου 2, ο οποίος επιτελεί τις ίδιες λειτουργίες με αυτόν του NAT Network.



Εικόνα 35. Internal Network

Ενσωματώνοντας τα δύο δίκτυα (10.0.2.0/24 και 10.2.2.0/24) σε μια συνολική τοπολογία δικτύου που χρησιμοποιήθηκε ως βάση για την προσομοίωση της λειτουργίας του IDS/IPS Software, προκύπτει η παρακάτω.

Lab Setup



Εικόνα 36. Lab Setup

Προκειμένου να προσομοιωθεί η λειτουργία Port Mirroring του Virtual Switch που διαχειρίζεται το δίκτυο 10.0.2.0/24, ενεργοποιήθηκε η λειτουργία χωρίς κριτήρια του VirtualBox (ή αλλιώς Promiscuous Mode) στην εικονική διεπαφή enp0s3 του IDS/IPS. Το VM πλέον μπορεί να “βλέπει” και να επεξεργάζεται όλη την κυκλοφορία στο συνδεδεμένο δίκτυο, και όχι μόνο αυτή που απευθύνεται στη δική του διεύθυνση MAC, όπως ακριβώς σε μια συνθήκη Port Mirroring φυσικού μεταγωγέα σε αντίστοιχη συσκευή

Για την απομακρυσμένη πρόσβαση στο Victim1 κατά τη διάρκεια μιας επίθεσης, και προκειμένου να σταλθεί και να εκτελεστεί με ασφάλεια το κατάλληλο Shield, χρησιμοποιήθηκε το εργαλείο OpenSSH [28]. Πρόκειται για λογισμικό ανοιχτού κώδικα, που είναι ήδη ενσωματωμένο στην έκδοση Ubuntu του Victim VM. Με αυτόν τον τρόπο το Victim1 και το IDS/IPS αποκτούν μια σύνδεση Server – Client, με το πρώτο να ”ακούει” για συνδέσεις στο Port 23 μέσω του ενεργού OpenSSH Deamon [28] και το δεύτερο να

ταυτοποιείται με τη μέθοδο Public Key Authentication, αυτοματοποιώντας περεταίρω τη διαδικασία. Εν συντομία, ύστερα από την ενεργοποίηση του Deamon στη μηχανή Victim1 *(sudo systemctl enable sshd)* εκτελέστηκαν τα παρακάτω βήματα.

- *ssh-keygen -t ed25519 -f ~/.ssh/victim1* Εντολή δημιουργίας των δύο απαιτούμενων SSH κλειδιών (Private – Public) χρησιμοποιώντας τον γνωστό αλγόριθμο ed25519 [28] και αποθήκευσή τους στο directory ~/.ssh/victim1. Εκτελείται στη μηχανή του Client (IDS/IPS VM).

- *ssh-copy-id -i .ssh/victim1.pub manos@10.2.2.8* Εντολή αποστολής του δημόσιου κλειδιού - victim1.pub στον Server - Victim1 VM. Το απαιτούμενο Username της προστατευόμενης μηχανής είναι manos ενώ όπως φαίνεται και στην Εικόνα 36, η διεύθυνση IP της διεπαφής enp0s9 (ή αλλιώς Hostname) είναι 10.2.2.8.

Πλέον η μηχανές είναι σε θέση να συνδεθούν δίχως τη χρήση κωδικού πρόσβασης. Διαμορφώνοντας το Client Configuration File που διατίθεται από το εργαλείο στο Directory *~/.ssh/config* όπως παρακάτω, αρκεί μια απλή εντολή για την έναρξη της σύνδεσης: *SSH Victim1* (Εικόνα 37).

Host victim1
HostName 10.2.2.8
IdentityFile ~/.ssh/victim1
User manos

```
manos@manos-VirtualBox:~/Desktop$ ssh victim1
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

66 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***
Last login: Wed Nov 20 17:28:59 2024 from 10.2.2.7
manos@manosvictim1-VirtualBox:~$ exit
logout
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 37. Σύνδεση SSH στο Victim1 VM

2.2 Ανάπτυξη Λογισμικού

Σε αυτή την υποενότητα θα πραγματοποιηθεί παρουσίαση του λογισμικού που αναπτύχθηκε, σύμφωνα με τις σχεδιαστικές βάσεις που τέθηκαν από τη θεωρητική ανάλυση της προηγούμενης Ενότητας. Όντας ιδανική στην αυτοματοποίηση επαναλαμβανόμενων διεργασιών, χρησιμοποιήθηκε η διαδικασία του Bash Scripting. Το Bash (ή αλλιώς Bourne Again Shell) είναι ένας διερμηνέας γραμμής εντολών ή Unix Shell που χρησιμοποιείται ευρέως στο λειτουργικό σύστημα GNU/Linux [29], ενώ ένα αρχείο Bash (Script) περιλαμβάνει σύνολο εντολών Shell που θα εκτελεστούν αυτόματα, δίχως ανθρώπινη παρέμβαση. Το βασικό Bash Script της παρούσας εργασίας είναι το *Mainscript.sh*, το οποίο αποτελεί τη "ραχοκοκαλιά" του συστήματος ασφάλειας. Εκτελώντας το, ο διαχειριστής ασφάλειας μπορεί να παρατηρεί στο Terminal του IDS/IPS VM την εξέλιξη μιας επίθεσης, μέσω των βημάτων που εκτελούνται από το σύστημα για την ανίχνευση και την αντιμετώπισή της. Παρακάτω παρατίθεται το πρώτο μέρος του Script, όπου εισάγονται οι απαραίτητες μεταβλητές του δικτύου που θα χρησιμοποιηθούν από τις επερχόμενες μονάδες.

```
  GNU nano 6.2
#!/bin/bash

##############################
#Set up the Network Variables
##############################

ids_ips_ip="10.0.2.4"
ids_enp0s9_ip="10.2.2.7"
victim1_enp0s9_ip="10.2.2.8"
victim1_ip="10.0.2.5"
default_gateway_ip="10.0.2.1"

home_network="10.0.2.0/24"

interface="enp0s3"

router_mac="52:54:00:12:35:00"
victim1_mac="08:00:27:c0:f8:21"
ids_ips_mac="08:00:27:98:39:56"

dns1="192.168.1.1"
dns2="1.1.1.1"

emergency_victim1="10.0.2.50/24"

dhcp_server="10.0.2.3"


##############################
#Functions
##############################

center_text() {

    local term_width=$(tput cols)

    local text="$1"

    local bold_colored_text="\e[1;33m$text\e[0m"

    local text_length=${#text}

    local padding=$(( (term_width - text_length) / 2 ))

    printf "%${padding}s" ""

    echo -e "$bold_colored_text"

}
```

Εικόνα 38. Ρύθμιση Μεταβλητών Δικτύου στη Mainscript.sh

Η Function Center_text () χρησιμοποιείται για αισθητικούς λόγους καθώς όταν εφαρμόζεται σε κείμενο όπως το "Starting Up…" του προσδίδει την παρακάτω μορφή, κατά την εκτέλεση του Script.

```
manos@manos-VirtualBox:~/Desktop$ ./mainscript.sh

                                                    Starting Up...
```

Εικόνα 39. Εκτέλεση του Script

2.2.1 Μονάδα Ανίχνευσης

Η μονάδα ανίχνευσης επιθέσεων (Intrusion Detection Module) που αναλύθηκε στο θεωρητικό υπόβαθρο υλοποιείται με την χρήση του εργαλείου ανοικτού κώδικα Snort. Το Snort, μεταξύ άλλων, λειτουργεί ως σύστημα ανίχνευσης εισβολών στο δίκτυο (NIDS) που παρακολουθεί και αναλύει την υφιστάμενη κυκλοφορία για ύποπτη δραστηριότητα. Θεωρείται από τα πιο δημοφιλή εργαλεία στην κοινότητα της κυβερνοασφάλειας, καθώς προσφέρει έγκυρες και γρήγορες ειδοποιήσεις σε περιπτώσεις παραβίασης της ασφάλειας. Χρησιμοποιεί έναν συνδυασμό ανίχνευσης βάσει υπογραφών, ανάλυσης πρωτοκόλλων και ανίχνευσης ανωμαλιών για τον εντοπισμό των απειλών, με αποτέλεσμα να είναι εξαιρετικά ευέλικτο και προσαρμόσιμο σε διαφορετικά είδη αναγκών. Όπως αναλύθηκε παραπάνω, στο συγκεκριμένο πείραμα το Snort λειτουργεί κατά βάση ως Signature Based Detection System καθώς βασίζεται σε ένα σύνολο κανόνων για τον ορισμό υπογραφών ανίχνευσης. Οι κανόνες του Snort χωρίζονται σε δύο λογικά τμήματα, την κεφαλίδα (Header) και τις επιλογές (Options). Η επικεφαλίδα περιέχει τη ενέργεια του κανόνα (alert σε αυτή την περίπτωση), το πρωτόκολλο, τις διευθύνσεις IP πηγής και προορισμού καθώς και πληροφορίες για τις αντίστοιχες θύρες. Το τμήμα επιλογών κανόνα περιέχει κατάλληλα μηνύματα ειδοποίησης και πληροφορίες σχετικά με τα μέρη του πακέτου που πρέπει να ελεγχθούν για να καθοριστεί αν πρέπει να αναληφθεί η ενέργεια που αναγράφεται στην κεφαλίδα [30].

Αφού λοιπόν το Snort (έκδοση 2.9) εγκαταστάθηκε στο IDS/IPS VM συμπληρώθηκαν οι κατάλληλες μεταβλητές δικτύου στο αρχείο */etc/snort/snort.conf* με τις διευθύνσεις IP των προστατευόμενων Hosts, όπως αυτές προκύπτουν από την Εικόνα 36. Οι παραμετροποιήσεις φαίνονται παρακάτω.

```
  GNU nano 6.2                                                    /etc/snort/snort.conf2 *
# /etc/snort/snort.$interface.conf (where '$interface' is the name of your
# network interface) and adjust the valuse there.
#
# The Debian init.d script is defined in such a way
# that you can run multiple instances.

#####################################################
# Step #1: Set the network variables.  For more information, see README.variables
#####################################################

# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET [10.0.2.1,10.0.2.4,10.0.2.5,10.0.2.3]

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

#DHCP SERVER CONFIGURATION
ipvar DHCP_SERVERS [10.0.2.3]
```

Εικόνα 40. Παραμετροποιήσεις αρχείου Snort.conf

Το Snort συλλέγει (Sniff) την κυκλοφορία μέσω της βιβλιοθήκης Libcap [30], η οποία δίνει τη δυνατότητα μεταφοράς των πακέτων από την εικονική διεπαφή του IDS/IPS VM στις μονάδες ανίχνευσης όπου θα μπορέσει το Snort να τα επεξεργαστεί. Η διαδικασία περιγράφεται από την μονάδα Packet Acquisition του Block Diagram (Εικόνα 14). Αφού επεξεργαστούν από τη Decoder & Preprocessor Module (η οποία θα αναλυθεί στη συνέχεια), το Payload των πακέτων περνάει στην Detection Engine η οποία δουλεύει παράλληλα με το Ruleset. Χρησιμοποιήθηκε το αρχείο *etc/snort/rules/local.rules* για τη συγγραφή εξατομικευμένων κανόνων που σκοπεύουν στο μικρότερο δυνατό χρόνο απόκρισης του συστήματος σε περίπτωση παραβίασης της ασφάλειας. Οι προ-κατασκευασμένοι κανόνες που διατίθενται από την κοινότητα του Snort [31] απενεργοποιήθηκαν και αντ' αυτών δημιουργήθηκε το παρακάτω Ruleset. Οι μεταβλητές *$HOME_NET* και *$DHCP_SERVERS* προκύπτουν από την Εικόνα 40.

1) *alert tcp !$HOME_NET any -> $HOME_NET any (msg:"Potential Nmap Xmas Scan"; flow:stateless; flags:FPU; threshold: type both, track by_src, count 500, seconds 5; priority:1; sid:1001;)*

2) *alert tcp !$HOME_NET any -> $HOME_NET any (msg:"Potential Nmap Null Scan"; flow:stateless; ack:0; flags:0; threshold: type both, track by_src, count 500, seconds 5; priority:1; sid:1002;)*

3) *alert tcp !$HOME_NET any -> $HOME_NET any (msg:"Potential Nmap Fin Scan"; flow:stateless; flags:F; threshold: type both, track by_src, count 500, seconds 5; priority:1; sid:1003;)*

4) *alert tcp $HOME_NET any -> !$HOME_NET any (msg:"Potential Nmap Stealth/Full Scan"; flow:stateless; flags:RA; threshold: type both, track by_src, count 500, seconds 5; sid:1004; rev:1;)*

5) *alert udp any any -> $HOME_NET 80 (msg:"Possible UDP Flood DDOS Attempt"; flow: stateless; threshold: type both, track by_src, count 10000, seconds 2; sid:1005; rev: 1;)*

6) *alert tcp any any -> $HOME_NET 80 (msg:"Possible TCP Flood DDOS Attempt"; flags: S; flow: stateless; threshold: type both, track by_src, count 10000, seconds 2; sid:1006; rev: 1;)*

7) *alert icmp any any -> $HOME_NET 80 (msg: "Possible ICMP Flood DDOS Attempt"; itype: 8; threshold: type both, track by_src, count 10000, seconds 2; sid: 1010; rev: 1;)*

8) *alert tcp !$HOME_NET any -> $HOME_NET 22 (msg: "SSH Attempt (Request To Connect to Server)"; flow:stateless; content:"|53 53 48|",fast_pattern; priority:1; sid:1008;)*

9) *alert icmp any any -> $HOME_NET any (msg: "Potential ICMP Redirection Attempt"; content:!"|52 54 00 12 35 00|"; fast_pattern; itype: 5; icode: 1; threshold: type limit, track by_src, count 1 , seconds 10; rev:1; sid: 1019;)*

10) *alert udp any 67 -> any 68 (msg:"DHCP SPOOFING"; content:"|63 82 53 63|"; fast_pattern:only; content:!"|0a 00 02 01|"; content:!"|c0 a8 01 01|"; sid:1007; rev:5;)*

Οι κανόνες 1 έως 3 καλύπτουν τα σενάρια των επιθέσεων Xmas, Null και Fin Scan. Παρατηρείται ότι και οι 3 αφορούν κυκλοφορία TCP, ενώ η Detection Engine αναζητεί ως διεύθυνση αποστολέα οποιαδήποτε IP δεν ανήκει στο σύνολο *HOME_NET*. Τα Port αποστολέα και παραλήπτη μπορούν επίσης να είναι οποιαδήποτε (*any*). Η ρύθμιση *flow:stateless* αναγκάζει τη μηχανή ανίχνευσης να ελέγχει το κάθε πακέτο που δέχεται μεμονωμένα, χωρίς να βασίζεται στην κατάσταση της σύνδεσης TCP μεταξύ του αποστολέα και του παραλήπτη (established ή not established). Βασικό διαχωριστικό στοιχείο των κανόνων είναι η ύπαρξη των Flags, τα οποία σε κάθε περίπτωση τίθενται όπως προκύπτει από τους τύπους των σαρώσεων που αναλύθηκαν. Συνολικά, το σύστημα ανίχνευσης πυροδοτείται και παράγει μια ειδοποίηση (*alert*) με το κατάλληλο μήνυμα (*msg*) όταν οποιοδήποτε TCP πακέτο από "ξένο" αποστολέα σταλθεί στις διευθύνσεις του HOME_NET σε οποιαδήποτε Ports, και περιέχει ένα εκ των τριών συνδυασμό σημαιών (FPU, F και No Flags Set). Με την επιλογή του *threshold: type both, track by_src, count 500, seconds 5,* το Snort παράγει μια ειδοποίηση για κάθε IP προέλευσης που πυροδοτεί αυτόν τον κανόνα εάν μέσα σε 5 δευτερόλεπτα συμπληρωθούν 500 πακέτα πυροδότησης. Για το υπόλοιπο χρονικό διάστημα και μέχρι την λήξη των 5', το Snort αγνοεί τυχών πρόσθετα πακέτα της ίδιας IP προέλευσης. Δεδομένου ότι by default το Nmap σαρώνει τις θύρες 1 έως 1000 του θύματος, οι παραπάνω κανόνες θα παράγουν μια ειδοποίηση

ανά τύπο Scan. Για τα σενάρια των Stealth και Full σαρώσεων, το σύστημα καταλαβαίνει ότι κάποια προστατευόμενη συσκευή δέχεται ένα εκ των δύο Scan μέσω του κανόνα 4. Αυτός παράγει το αντίστοιχο Alert, όταν η προστατευόμενη συσκευή που θα δεχτεί τα κακόβουλα πακέτα SYN, στείλει πίσω στον επιτιθέμενο έναν μεγάλο όγκο αντίστοιχων Reset. Το παραπάνω επιτυγχάνεται από την αντιστροφή των πιθανών διευθύνσεων αποστολέα και παραλήπτη στο Header και την προσθήκη των Flags ACK και RST. Παρά το γεγονός ότι αντίστοιχα πακέτα RST στέλνονται από το θύμα και κατά τις σαρώσεις XMAS FIN και NULL, στους κανόνες 1-3 προστίθεται η επιλογή *priority:1* η οποία τους προσδίδει τη μέγιστη δυνατή προτεραιότητα κατά τον έλεγχο στη Detection Engine.

Οι κανόνες 5 έως 7 αφορούν τους τύπους πλημμυρών που αναλύθηκαν σε μια επίθεση Denial of Service. Αναζητούν πακέτα κυκλοφορίας *udp, tcp* (*flags: S* – SYN) και *icmp* (*itype: 8* – Ping Request) από οποιαδήποτε διεύθυνση και θύρα υπερβούν τις τιμές *threshold* 10.000 πακέτων ανά 2 δευτερόλεπτα. Από δοκιμή καταπόνησης (Stress Testing) που πραγματοποιήθηκε στο Victim1 VM, προέκυψε ότι τα παραπάνω επίπεδα κυκλοφορίας αλλοιώνουν αισθητά τη λειτουργικότητά του. Σε αυτό το σημείο τοποθετήθηκε και το κατώφλι πυροδότησης του Snort, με τις παραπάνω διακριτές ειδοποιήσεις.

Ο κανόνας 8 καταγράφει μια ειδοποίηση κάθε φορά που εξωτερική συσκευή επιχειρεί να εδραιώσει συνεδρία SSH με αντίστοιχο διακομιστή του προστατευόμενου συνόλου. Συγκεκριμένα, η μηχανή αναζήτησης αναλύει το Payload των TCP πακέτων που απευθύνονται στη θύρα 22 μιας προστατευόμενης μηχανής, πραγματοποιώντας αντιστοίχιση περιεχομένου (*content:"|53 53 48|"*). Οι δεκαεξαδικοί αυτοί χαρακτήρες σε κωδικοποίηση ASCII αντιστοιχούν σε “SSH”, και παρατηρούνται στα προαναφερόμενα πακέτα συνεδρίας. Προσθέτοντας τη λειτουργία *fast_pattern* αυξάνεται η ταχύτητα ταυτοποίησης και ενεργοποίησης του κανόνα, ο οποίος προφανώς ειδοποιεί για πιθανές SSH Password Authentication Brute Force επιθέσεις. Σημειώνεται ότι ο κανόνας δεν επιχειρεί κάποια επιβεβαίωση της επίθεσης, καθώς η διεργασία αυτή πραγματοποιείται στην επερχόμενη μονάδα Attack Verification.

Ομοίως με παραπάνω, ο κανόνας 9 δεν επιχειρεί να επιβεβαιώσει κάποια επίθεση ICMP Redirection, αλλά καταγράφει μια ειδοποίηση όταν οποιαδήποτε μη εξουσιοδοτημένη διεύθυνση στείλει αντίστοιχο πακέτο (Redirect For Host - *itype: 5; icode*) στο προστατευόμενο σύνολο. Συγκεκριμένα, πυροδοτείται σε πακέτα που δεν έχουν σταλθεί από την προβλεπόμενη συσκευή - Virtual Router, ελέγχοντας το Payload για την απουσία της διεύθυνση MAC του (*content:!"|52 54 00 12 35 00|"; fast_pattern*). Προκειμένου να αποφευχθεί η συγκέντρωση μεγάλου όγκου διπλότυπων ειδοποιήσεων, εφαρμόζεται το *threshold: type*

limit προκειμένου να καταγράφεται μόνο το πρώτο πακέτο που παραβιάζει τον ίδιο κανόνα, σε ένα διάστημα 10 δευτερολέπτων.

Τέλος, ο κανόνας 10 του Ruleset ενεργοποιείται όταν οποιοσδήποτε μη εξουσιοδοτημένος διακομιστής DHCP επιχειρήσει να στείλει κάποιο πακέτο (Offer ή Ack) σε συσκευή του προστατευόμενου συνόλου. Η τελευταία στον κανόνα αναπαρίσταται με *any*, καθώς κατά τη διάρκεια της διαδικασίας δεν κατέχει κάποια διεύθυνση IP, και στα πακέτα Broadcast εμφανίζεται ως "0.0.0.0". Συνολικά, ο κανόνας ελέγχει κυκλοφορία UDP και συγκεκριμένα αυτή από τις θύρες 67 και 68 του αποστολέα και παραλήπτη αντίστοιχα. Αυτό που διαχωρίζει τα πακέτα DHCP από οποιονδήποτε άλλο τύπο κυκλοφορίας UDP είναι η ύπαρξη του DHCP Magic Cookie [32], για αυτό και επισημαίνεται με *fast_pattern:only*. Αφού πραγματοποιηθεί η απαραίτητη αντιστοίχιση με το Cookie 63 82 53 63, η μηχανή αναζήτησης ελέγχει εάν απουσιάζουν από το Payload οι δύο βασικές τιμές διευθύνσεων ενός DHCP μηνύματος, αυτή του Default Gateway (*0a 00 02 01* – 10.0.2.1) και του Primary DNS Server (*c0 a8 01 01 – 192.168.1.1*). Εάν έστω και μια από τις δύο αποκλίνει από τις συνηθισμένες τιμές τότε το σύστημα παράγει μια ειδοποίηση.

Ένα βασικό χαρακτηριστικό μιας Detection Engine που δουλεύει με συγκεκριμένο Ruleset, είναι η αδυναμία διαχείρισης και ανάλυσης πακέτων που υπάγονται σε πρωτόκολλα κάτω από το επίπεδο 3 του OSI (IP – Based Traffic). Συνεπώς είναι ανέφικτη η συγγραφή συγκεκριμένου κανόνα που να αποτυπώνει την υπογραφή μιας ARP Spoofing επίθεσης και να παράγει τις κατάλληλες ειδοποιήσεις όταν εντοπίζει ύποπτα Level 2 Frames. Το πρόβλημα αυτό λοιπόν έρχεται να επιλύσει η μονάδα Decoder & Preprocessor, και συγκεκριμένα ο " Arpspoof " Preprocessor, ο οποίος συμπληρώνει την απαιτούμενη "ορατότητα" στο Snort – IDS, έχοντας τη δυνατότητα ανάλυσης των μηνυμάτων ARP. Ρυθμίζοντας το αρχείο */etc/snort/snort.conf* όπως παρακάτω, τίθενται τα ζευγάρια διευθύνσεων IP – MAC των συσκευών που θα παρακολουθεί το σύστημα για ενδεχόμενες επιθέσεις ARP Spoofing. Όταν λοιπόν σε αυτή τη μονάδα ελέγχου ανιχνευθεί οποιοδήποτε Frame ARP – Reply που θα παραβιάζει τη δομή των παρακάτω ζευγαριών, το σύστημα παράγει alert, συνοδευόμενο από το μήνυμα *"ARP Cache Overwrite Attack"*.

 # ARP spoof detection. For more information, see the Snort Manual - Configuring Snort - Preprocessors - ARP Spoof Preprocessor

preprocessor arpspoof_detect_host: 10.0.2.1 52:54:00:12:35:00

preprocessor arpspoof_detect_host: 10.0.2.4 08:00:27:98:39:56

preprocessor arpspoof_detect_host: 10.0.2.5 08:00:27:c0:f8:21

alert (msg: "ARP Cache Overwrite Attack"; sid:4; gid:112;)

Η εντολή που εκκινεί το εκτελέσιμο αρχείο του Snort προκειμένου αυτό να ξεκινήσει την ανάλυση της υφιστάμενης κυκλοφορίας, είναι η εξής:

sudo snort -q -l /var/log/snort -i $interface -A fast -c /etc/snort/snort.conf2 -b όπου:

-q. Καταστέλλει αχρείαστο Output, όπως ορισμένες πληροφορίες χρόνου εκτέλεσης, καθιστώντας την έξοδο πιο καθαρή.

-l /var/log/snort. Καθορίζει το Directory του αρχείου καταγραφής των Snort Logs από τις αντίστοιχες ειδοποιήσεις. Όταν το Snort ανιχνεύσει επίθεση για πρώτη φορά, δημιουργεί στο παραπάνω Directory ένα αρχείο Plain Text που ονομάζεται alert και κάθε φορά που πρέπει να αποθηκεύσει νέα Logs επιθέσεων, τροποποιεί το εν λόγω Document. Ενδεικτικά παρακάτω παρουσιάζεται screenshot του αρχείου ύστερα από πολλαπλές δοκιμαστικές επιθέσεις. Αποτελεί συνοπτικό αποδεικτικό στοιχείο από τον κανόνα που παραβιάστηκε, και ελέγχεται τακτικά από τη μονάδα αντιμετώπισης Attack Verification & Classification για πιθανώς νέες τροποποιήσεις.

-i $interface. Καθορίζεται η διεπαφή δικτύου που θα παρακολουθεί το Snort. Στο συγκεκριμένο πείραμα, αυτή είναι η enp0s3 και συνδέει το IDS/IPS VM με το κύριο δίκτυο (10.0.2.0/24).

-A Fast. Καταγράφει όλες τις ειδοποιήσεις στο Alert File, και όχι στο Console του VM.

-c /etc/snort/snort.conf2. Προσδιορίζεται το βασικό αρχείο Configuration της διαδικασίας, που αναλύθηκε και παραπάνω. Σημειώνεται ότι το snort.conf**2** είναι το ίδιο και αποτελεί αντιγραφή του snort.conf.

-b. Δημιουργεί αρχείο δυαδικής μορφής (Binary Format) για την καταγραφή κακόβουλων πακέτων, τα οποία μπορούν αργότερα να αναλυθούν με το γνωστά εργαλεία όπως Tcpdump ή Wireshark.

Το παραπάνω εφαρμόζεται στο Mainscript.sh όπως φαίνεται στην Εικόνα 42. Καθώς το Snort τρέχει σε διαφορετικό Terminal από το Script, χρησιμοποιείται το εργαλείο wmctl για την αυτόματη διαχείριση του αναδυόμενου παραθύρου. Όταν ο διαχειριστής συμπληρώσει τον απαιτούμενο κωδικό και δημιουργηθεί διεργασία (“pid”) για το Snort, τότε το σύστημα είναι έτοιμο να λειτουργήσει ως NIDS.

```
1773 09/25-20:14:19.379174  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:54972 -> 10.0.2.5:22
1774 09/25-20:14:22.332494  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:44098 -> 10.0.2.4:22
1775 09/25-20:14:22.634317  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:44122 -> 10.0.2.4:22
1776 09/25-20:14:22.634530  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:44130 -> 10.0.2.4:22
1777 09/25-20:14:22.634530  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:44110 -> 10.0.2.4:22
1778 09/25-20:14:22.635114  [**] [1:1008:0] SSH Attempt (Request To Connect to Server) [**] [Priority: 1] {TCP} 10.0.2.7:44138 -> 10.0.2.4:22
1779 09/26-00:11:03.771720  [**] [1:1001:0] Potential Nmap Xmas Scan [**] [Priority: 1] {TCP} 10.0.2.44:59977 -> 10.0.2.5:3011
1780 09/26-00:11:03.771720  [**] [1:1001:0] Potential Nmap Xmas Scan [**] [Priority: 1] {TCP} 10.0.2.7:59977 -> 10.0.2.5:3011
1781 09/26-00:11:03.771882  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.5:9091 -> 10.0.2.7:59977
1782 09/26-00:44:05.833413  [**] [1:1001:0] Potential Nmap Xmas Scan [**] [Priority: 1] {TCP} 10.0.2.44:50959 -> 10.0.2.5:1152
1783 09/26-00:44:05.833464  [**] [1:1001:0] Potential Nmap Xmas Scan [**] [Priority: 1] {TCP} 10.0.2.7:50959 -> 10.0.2.5:1152
1784 09/26-00:44:05.833661  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.5:10012 -> 10.0.2.7:50959
1785 09/26-00:44:09.381259  [**] [1:1001:0] Potential Nmap Xmas Scan [**] [Priority: 1] {TCP} 10.0.2.49:57054 -> 10.0.2.5:6666
1786 09/26-00:57:49.472589  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.4:1322 -> 10.0.2.7:49529
1787 09/26-01:02:51.011436  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.4:2179 -> 10.0.2.7:64398
1788 09/26-01:02:51.688684  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.5:42510 -> 10.0.2.7:42570
1789 09/26-01:10:01.239977  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.5:1972 -> 10.0.2.7:61838
1790 09/26-01:10:02.094808  [**] [1:1004:1] Potential Nmap Stealth/Full Scan [**] [Priority: 0] {TCP} 10.0.2.4:8099 -> 10.0.2.7:54474
1791 09/26-01:42:33.096972  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1792 09/26-01:42:33.096973  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1793 09/26-01:42:33.116046  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1794 09/26-01:42:33.116047  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1795 09/26-01:42:34.128849  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1796 09/26-01:42:34.128850  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1797 09/26-01:42:35.139430  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1798 09/26-01:42:35.139432  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1799 09/26-01:43:30.765829  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1800 09/26-01:43:30.765843  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1801 09/26-01:43:31.776162  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1802 09/26-01:43:31.776163  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1803 09/26-01:43:32.786486  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1804 09/26-01:43:32.786487  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1805 09/26-01:43:33.796607  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1806 09/26-01:43:33.796608  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1807 09/26-01:43:34.809010  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1808 09/26-01:43:34.809011  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1809 09/26-01:47:31.600976  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1810 09/26-01:47:31.601039  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1811 09/26-01:47:32.615852  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1812 09/26-01:47:32.615856  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1813 09/26-01:47:33.626970  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1814 09/26-01:47:33.626979  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1815 09/26-01:47:34.637145  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1816 09/26-01:47:34.637151  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1817 09/26-01:47:35.647437  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1818 09/26-01:47:35.647441  [**] [112:4:0] ARP Cache Overwrite Attack [**]
1819 09/26-03:08:18.998046  [**] [1:1019:1] Potential ICMP Redirection Attempt [**] [Priority: 0] {ICMP} 10.0.2.7 -> 10.0.2.5
1820 09/26-03:11:34.626172  [**] [1:1019:1] Potential ICMP Redirection Attempt [**] [Priority: 0] {ICMP} 10.0.2.7 -> 10.0.2.5
1821 09/26-03:11:34.626173  [**] [1:1019:1] Potential ICMP Redirection Attempt [**] [Priority: 0] {ICMP} 10.0.2.1 -> 10.0.2.5
1822 09/26-03:14:21.595749  [**] [1:1019:1] Potential ICMP Redirection Attempt [**] [Priority: 0] {ICMP} 10.0.2.77 -> 10.0.2.5
```

Εικόνα 41. /var/log/snort/alert - Snort Alert Logs

```
manos@manos-VirtualBox: ~/Desktop
  GNU nano 6.2                                   mainscript.sh
##############################
#Snort NIDS Start Up
##############################

echo
echo
center_text "Starting Up..."
echo -e "\n\n"
sleep 2

gnome-terminal --title="snort" -- bash -c "cd && sudo snort -q -l /var/log/snort -i $interface -A fast -c /etc/snort/snort.conf2 -b; exec bash"

while true; do

    SNORT_PID=$(pidof snort)

    if [[ -n "$SNORT_PID" ]]; then

        window_id=$(wmctrl -l | grep "manos@manos-VirtualBox: ~/Desktop" | awk '{print $1}')

        wmctrl -i -a $window_id

        break

    fi

    sleep 1

done

sleep 1
center_text "-- Defensive System Initialized -- Waiting for Snort to Generate Alerts --"
echo
echo
```

Εικόνα 42. Εκκίνηση Snort στο Mainscript.sh

2.1.2 Μονάδα Αντιμετώπισης

Προχωρώντας από τη διαδικασία ανίχνευσης σε αυτή της αντιμετώπισης των επιθέσεων, η πρώτη αντίστοιχη μονάδα που συναντάται στην Εικόνα 14 (Block Diagram) είναι η Attack Verification & Classification. Η υλοποίηση της διαδικασίας αποτυπώνεται στη Mainscript.sh ως εξής.

```
#######################################
#Attack Verification & Classification
#######################################

file_path="/var/log/snort/alert"

initial_size=$(wc -l < "$file_path")

sudo inotifywait -e modify "$file_path" > /dev/null 2>&1
sleep 1

while true; do

    INOTIFYWAIT_PID=$(pidof inotfywait)

    if [[ -z "$INOTIFYWAIT_PID" ]]; then

       timestamp=$(date +"%d/%m %H:%M:%S")
       echo "Snort Detected Malicious Traffic at $timestamp" &&
       sleep 2
       new_lines=$(awk "NR > $initial_size" "$file_path")

       break

    fi

    sleep 1

done

if echo "$new_lines" |  grep -q "Nmap"; then

   source nmaphandler.sh

elif echo "$new_lines" |  grep -q "ARP"; then

   source arpspoofhandler.sh

elif  echo "$new_lines" |  grep -q "SSH Attempt"; then

   source bruteforcehandler.sh

elif echo "$new_lines" |  grep -q "Redirection"; then

   source redirectionhandler.sh

elif echo "$new_lines" |  grep -q "DDOS"; then

   source doshandler.sh

elif echo "$new_lines" |  grep -q "DHCP SPOOFING."; then

   source dhcpspoofhandler.sh

fi
```

Εικόνα 43. Attack Verification & Classification

Δεδομένου ότι το Snort σε περίπτωση ανίχνευσης επιθέσεων τροποποιεί ακαριαία το αρχείο alert με τις νεότερες ειδοποιήσεις, χρησιμοποιείται το εργαλείο inotifywait για τη συνεχή εποπτεία του. Ύστερα από μια τροποποίηση στο φάκελο, το εργαλείο αυτομάτως θα την εντοπίσει, θα την καταγράψει και ύστερα θα αναστείλει τη λειτουργία του. Το pid του πλέον θα είναι μηδέν, και το σύστημα θα εξέλθει από την ατέρμονη while loop, έχοντας πρώτα αποθηκεύσει στη μεταβλητή new_lines, όλες τις καταγραφές που προστέθηκαν στο alert file, σε διάστημα περίπου δύο δευτερολέπτων ύστερα από την αρχική. Το περιεχόμενο της μεταβλητής είναι γραμμές καταγραφών, όπως αυτές που παρατίθενται στην Εικόνα 41, και αναλύονται από την if statement προκειμένου η υφιστάμενη επίθεση να ταξινομηθεί σε μια κατηγορία που καλύπτει το παρόν σύστημα. Ο έλεγχος πραγματοποιείται μέσω του εργαλείου grep, το οποίο ταυτίζει το περιεχόμενο των γραμμών με τις λέξεις “Nmap”, “ARP”, “SSH Attempt”, “Redirection” “DDOS” και “DHCP SPOOFING”. Οι λέξεις αυτές αντιστοιχούν στα messages των κανόνων που καταγράφηκαν στο Ruleset της παραπάνω υποενότητας, και παραπέμπουν το σύστημα στον κατάλληλο handler της επίθεσης. Οι handlers αυτοί υλοποιούνται μέσω εμφωλευμένων Script, τα οποία περιέχουν τη χρήση γενικών function, αλλά και εξατομικευμένων ενεργειών και εκτελούνται προκειμένου να συνεχιστεί η διαδικασία αντιμετώπισης μιας επικείμενης επίθεσης. Προτού παρουσιαστούν τα περιεχόμενα των Handler Scripts, θα αναλυθούν τα βασικά functions που αναπτύχθηκαν στη Mainscript, και χρησιμοποιούνται για την ανάπτυξη των υπολοίπων μονάδων.

Ξεκινώντας από τη συνάρτηση **process_flag(),** σε αυτή πραγματοποιείται η ανάλυση πακέτων του αρχείου binary που θα δημιουργηθεί ύστερα από την πυροδότηση του Snort, με σκοπό την εξαγωγή των διευθύνσεων IP και MAC τόσο των κακόβουλων συσκευών όσο και των προστατευόμενων. Συγκεκριμένα εφαρμόζεται το εργαλείο ανάλυσης πακέτων Tcpdump στο πιο πρόσφατα δημιουργημένο αρχείο, και στη συνέχεια ακολουθούν οι διεργασίες άντλησης και ταξινόμησης των διευθύνσεων, προκειμένου να αποφευχθούν οι διπλότυπες καταγραφές. Το σύστημα βεβαιώνεται ότι στις 4 τελικές array θα βρίσκονται αποθηκευμένες μόνο οι τιμές των διευθύνσεων, αποκομμένες από ενδεχόμενες θύρες αποστολής ή παραλαβής, καθώς και αντιστοιχίες διευθύνσεων με Host Names που πραγματοποιεί αυτόματα το Tcpdump (πχ. 10.0.2.1 – Gateway). Ενδεικτικά, παρατίθεται το Output του εργαλείου ύστερα από ένα Xmas Scan ενώ σημειώνεται ότι το παρακάτω εναλλάσσεται κάθε φορά ανάλογα με το πρωτόκολλο που αναλύεται.

```
manos@manos-VirtualBox:~/Desktop$ sudo tcpdump -e -r /var/log/snort/snort.log.1732645732
reading from file /var/log/snort/snort.log.1732645732, link-type EN10MB (Ethernet), snapshot length 1514
20:29:07.724397 08:00:27:09:51:06 (oui Unknown) > 08:00:27:c0:f8:21 (oui Unknown), ethertype IPv4 (0x0800), length 60: 10.0.2.7.62180 > 10.0.2.5.1583: Flags [FPU], seq 2697874196, win 1024, urg 0, length 0
20:29:07.725312 08:00:27:c0:f8:21 (oui Unknown) > 08:00:27:09:51:06 (oui Unknown), ethertype IPv4 (0x0800), length 60: 10.0.2.5.2301 > 10.0.2.7.62180: Flags [R.], seq 0, ack 2697874197, win 0, length 0
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 44. Tcpdump Output σε Xmas Scan

```
  GNU nano 6.2                                                                    mainscript.sh
process_flag() {

    flag=$1
    attack_src_col=$2
    attack_dst_col=$3
    mal_mac_col=$4
    home_mac_col=$5

    echo
    echo
    echo "Anilyzing Snort Logs to Extract Crucial Information about the Attack"


    latest_log_file=$(ls -t /var/log/snort/snort.log.* | head -n 1) > /dev/null 2>&1
    tcpdump_output=$(sudo tcpdump -e -r "$latest_log_file" 2>/dev/null | grep -F "$flag" 2>/dev/null)


    while IFS= read -r attackline; do

            atckipsrc=$(echo "$attackline" | awk -v src_col=$attack_src_col '{print $src_col}' | cut -d':' -f1)
            atckipdest=$(echo "$attackline" | awk -v dst_col=$attack_dst_col '{print $dst_col}' | cut -d':' -f1)
            atckmacsrc=$(echo "$attackline" | awk -v malmac_col=$mal_mac_col '{print $malmac_col}')
            homemacdst=$(echo "$attackline" | awk -v homemac_col=$home_mac_col '{print $homemac_col}')

            source_ips_of_attack_unsorted+=("$atckipsrc")
            destination_ips_of_attack_unsorted+=("$atckipdest")
            source_macs_of_attack_unsorted+=("$atckmacsrc")
            home_macs_of_attack_unsorted+=("$homemacdst")

     done <<< "$tcpdump_output"

     source_ips_sorted=($(for sorted1 in "${source_ips_of_attack_unsorted[@]}"; do echo "${sorted1}"; done |  sort -u))
     destination_ips_sorted=($(for sorted2 in "${destination_ips_of_attack_unsorted[@]}"; do echo "${sorted2}"; done |  sort -u))
     source_macs_sorted=($(for sorted3 in "${source_macs_of_attack_unsorted[@]}"; do echo "${sorted3}"; done |  sort -u))
     home_macs_sorted=($(for sorted4 in "${home_macs_of_attack_unsorted[@]}"; do echo "${sorted4}"; done |  sort -u))
     length5=${#source_ips_sorted[@]}
     ipcombination_array_unconverted=("${source_ips_sorted[@]}" "${destination_ips_sorted[@]}")


     for ipconversion in "${ipcombination_array_unconverted[@]}"; do

         if [[ "$ipconversion" =~ ^_gateway ]]; then

            ipcombination_array+=("$default_gateway_ip")

         elif [[ "$ipconversion" =~ ^manos-VirtualBox ]]; then

              ipcombination_array+=("$ids_ips_ip")

         else
```

Εικόνα 45. Process_flag() Function [1/2]

```
  GNU nano 6.2                                                                    mainscript.sh
              ipcombination_array+=("$ids_ips_ip")

         else
              ipcombination_array+=("$ipconversion")

         fi


     done

     for converted in "${ipcombination_array[@]}"; do

         final_array_combined+=("$(echo "$converted" | cut -d'.' -f1-4)")

     done

     for ((l=0; l<length5; l++)); do

         final_source_unsorted+=("${final_array_combined[l]}")

     done


     for ((m=length5; m<${#final_array_combined[@]}; m++)); do

         final_destination_unsorted+=("${final_array_combined[m]}")

     done

     final_source_sorted=($(for sorted6 in "${final_source_unsorted[@]}"; do echo "${sorted6}"; done |  sort -u))
     final_destination_sorted=($(for sorted7 in "${final_destination_unsorted[@]}"; do echo "${sorted7}"; done |  sort -u))
     length1=${#source_macs_sorted[@]}
     length2=${#final_source_sorted[@]}
     length6=${#final_destination_sorted[@]}

     echo
     echo
     echo "Logs Check Came back with the Following:"
     echo
     echo "Source IP(s): ${final_source_sorted[@]}"
     echo
     echo "Destination -- Protected Network -- IP(s): ${final_destination_sorted[@]}"
     echo

}
```

Εικόνα 46. Process_flag() Function [2/2]

Κάθε φορά που καλείται η παραπάνω συνάρτηση, πρέπει να εκχωρείται ως input μια λέξη – κλειδί προκειμένου το σύστημα να εστιάζει στις σωστές γραμμές του Tcpdump Output, καθώς και οι θέσεις των ζητούμενων τιμών που θα αντληθούν και θα προστεθούν στα arrays. Αποτυπώνει τη μονάδα Source & Target Identification, και χρησιμοποιείται τοιουτοτρόπως από τον κάθε handler ξεχωριστά.

Προχωρώντας στις επόμενες συναρτήσεις της Mainscript, παρατίθενται οι **mac2ip_convertor()** και η **ip_unspoof()**. Όσον αφορά την πρώτη, δέχεται ως είσοδο ένα array που αποτελείται από τις MAC των γνωστών συσκευών, και τις μετατρέπει στις αντίστοιχες IP, με κατάλληλο μήνυμα εξόδου σε περίπτωση εισαγωγής άγνωστων διευθύνσεων. Η χρήση της είναι γενική, και δεν υπάγεται σε μια συγκεκριμένη μονάδα. Αντιστοίχως, η δεύτερη συνάρτηση δέχεται array κακόβουλων IP, και τις συγκρίνει με τις γνωστές διευθύνσεις του δικτύου, προκειμένου να βεβαιωθεί ότι δεν είναι Spoofed. Εάν δεν ταυτιστεί με καμία από αυτές, τότε εισάγεται στην not_spoofed_array. Η συγκεκριμένη συνάρτηση χρησιμοποιείται στα πλαίσια της μονάδας Evasion Countermeasure, και αποκλείει αυτόματα τις Spoofed IPs των επιτιθέμενων.

```
GNU nano 6.2
mac2ip_convertor() {
      protected_macs_notconverted=("$@")
      for macconvert in "${protected_macs_notconverted[@]}"; do
         if [ "$macconvert" = "$victim1_mac" ]; then
            protected_macs_converted+=("$victim1_ip")
         elif [ "$macconvert" = "$router_mac" ]; then
               protected_macs_converted+=("$default_gateway_ip")
         elif [ "$macconvert" = "$ids_ips_mac" ]; then
              protected_macs_converted+=("$ids_ips_ip")
         else
              echo "ERROR! Nothing Found For $macconvert"
         fi
      done
}

ip_unspoof() {
    ips_to_unspoof=("$@")
    for ipspoofed in "${ips_to_unspoof[@]}"; do
        if [[ "$ipspoofed" != "$ids_ips_ip" && \
              "$ipspoofed" != "$default_gateway_ip" && \
              "$ipspoofed" != "$dhcp_server" && \
              "$ipspoofed" != "$victim1_ip" ]]; then
                  not_spoofed_array+=("$ipspoofed")
        fi
    done
    length7=${#not_spoofed_array[@]}
}
```

Εικόνα 47. Mac2ip_convertor() & Ip_unspoof() Functions

Η function **network_scan()** χρησιμοποιείται κυρίως από την Evasion Countermeasure Module προκειμένου το σύστημα να ιχνηλατήσει τις κακόβουλες IP, εκτελώντας ένα ARP Scan. Θέτοντας τη μεταβλητή reverse_mode σε 1, για κάθε εισαγόμενη διεύθυνση MAC πραγματοποιείται σάρωση δικτύου μέσω μηνυμάτων ARP – Request προκειμένου η συσκευή του επιτιθέμενου να απαντήσει με την πραγματική της IP. Παράλληλα, παρέχονται οι απαιτούμενες πληροφορίες για την επιτυχία (ή μη) της διαδικασίας, καθώς και συγκεκριμένα output arrays για τις δύο αυτές περιπτώσεις (not_tracable για αποτυχία, scanned_ip_array για επιτυχία). Εάν η τιμή του reverse_mode τεθεί στο 2, τότε η συνάρτηση λειτουργεί με αντίστροφη λογική, δίνοντας ως είσοδο IPs και λαμβάνοντας στην έξοδο τις αντίστοιχες MACs.

```
  GNU nano 6.2                                                                  mainscript.sh
}


network_scan() {

    reverse_mode=$1
    shift
    interfacetype=$1
    shift
    mac_addresses_toscan=("$@")

    echo
    echo
    echo "Running a Scan to Map the Attacker's MAC(s) back to the Corresponding IP(s)"
    interface=$interfacetype
    network=$home_network
    not_traced=0

    for malmac in "${mac_addresses_toscan[@]}"; do

        scan_result=$(sudo arp-scan -I "$interface" "$network" | grep -i "$malmac" | awk -v field="$reverse_mode" '{print $field}')

        if [[ -z "$scan_result" ]]; then

             not_tracable+=("$malmac")
             ((not_traced++))

        else
               scanned_ip_array+=("$scan_result")
               reverse_ips+=("$malmac")

        fi

    done

    if [ "${#scanned_ip_array[@]}" -ne 0 ]; then

        echo
        echo "-> Finished Analyzing. Scan Results Indicate Malicious IP(s): ${scanned_ip_array[@]} <-"
        echo

    else
        echo
        echo "-> Scan Came Back with Nothing! <-"
        echo

    fi

    if [[ "$not_traced" -ne 0 ]]; then

       echo "-> Was not Able to Trace MAC(s): ${not_tracable[@]} <-"
       echo

    fi

}
```

Εικόνα 48. Network_scan() Function

Τέλος, η συνάρτηση **destination_ipcheck()** υπάγεται στη μονάδα Shield Deployment και είναι υπεύθυνη για την αυτόματη αποστολή και εκτέλεση του κατάλληλου Shield σε όλες τις παραβιασμένες συσκευές. Συγκεκριμένα, παρέχονται στο Input το όνομα της "ασπίδας" που επρόκειτο να σταλεί στα θύματα, καθώς και ένα combinedall_array που περιέχει λίστα των IP τους, καθώς και σύνολο περεταίρω διευθύνσεων και πληροφοριών που θα χρειαστούν για την εκτέλεση του Shield (πχ λίστα κακόβουλων διευθύνσεων προς αποκλεισμό). Αφού διαχωριστούν σωστά τα περιεχόμενα του παραπάνω array, το σύστημα καθορίζει τα Host Names των μηχανών που δέχονται την επίθεση (victim1 – Router – IDS/IPS) και χρησιμοποιεί το εργαλείο scp προκειμένου να αποστείλει το σωστό Script. Όπως ο Attack, έτσι και ο Shield Handler υλοποιούνται με τη μέθοδο των εμφωλευμένων Scripts, τα οποία θα αναλυθούν εκτενώς στη συνέχεια. Όσον αφορά την εκτέλεσή τους, η μηχανή IDS/IPS συνδέεται στο εκάστοτε θύμα μέσω SSH (όπως αναλύθηκε παραπάνω), τρέχει το Script, και στη συνέχεια διακόπτει τη συνεδρία.

```
GNU nano 6.2

destination_ipcheck() {

    separator=$1
    shift
    script_name=$1
    shift
    combinedall_array=("$@")

    for ((i=0; i<separator; i++)); do

        victims_array+=("${combinedall_array[i]}")

    done

    for ((i=separator; i<${#combinedall_array[@]}; i++)); do

        scriptspecific_array+=("${combinedall_array[i]}")

    done

    for protip in "${victims_array[@]}"; do

        if [ "$protip" = "$victim1_ip" ]; then

           attacked_machines+=("victim1")

        elif [ "$protip" = "$default_gateway_ip" ]; then

            attacked_machines+=("Router")

        elif [ "$protip" = "$ids_ips_ip" ]; then

            attacked_machines+=("IDS-IPS")

        else
              echo
              echo " !!  Detection Error For $protip !!"
              echo

        fi

    done

    echo "-- Protected-Netowrk Machine(s) Under Attack: ${attacked_machines[@]} --"
    echo
    echo
    echo
    echo

    for system in "${attacked_machines[@]}"; do

       if [[ "$system" != "Router" && "$system" != "IDS-IPS" ]]; then

          center_text "For $system..." &&
          (
          scp $script_name $system:/home/manos/Desktop > /dev/null  &&
          echo -e "\n" &&
          center_text "Security Script (Shield) Installed Successfully to the Target System" &&
          ssh -t $system "cd /home/manos/Desktop && ./$script_name ${scriptspecific_array[@]}"
          )

       fi

    done

}
```

Εικόνα 49. Destination_ipcheck() Function

Nmapscanhandler.sh

```
  GNU nano 6.2
#!/bin/bash

#Handler For Nmap Scans

echo
echo "Recon activity -- Potential Nmap Scan Detected!"
echo

sleep 6
sudo kill -15 $SNORT_PID
sleep 1


first_line=$(echo "$new_lines" | head -n 1)
scantype=0

#################################################################
#Further Classification and Source & Target Identification
#################################################################

if echo "$first_line" |  grep -q "Stealth/Full Scan"; then

   echo "Type: Nmap Stealth/Full Scan. Check the Log File for further Information."
   process_flag "Flags [R.]" "16" "14" "6" "16"
   ((scantype++))


elif echo "$first_line" |  grep -q "Fin"; then

     echo "Type: Nmap Fin Scan. Check the Log File for further Information."
     process_flag "Flags [F]" "14" "16" "2" "14"


elif echo "$first_line" |  grep -q "Null"; then

     echo "Type: Nmap Null Scan. Check the Log File for further Information."
     process_flag "Flags [none]" "14" "16" "2" "14"


elif echo "$first_line" |  grep -q "Xmas"; then

     echo "Type: Nmap Xmas Scan. Check the Log File for further Information."
     process_flag "Flags [FPU]" "14" "16" "2" "14"


else
```

Εικόνα 50. Nmaphandler.sh [1/2]

```
  GNU nano 6.2                                                                    nmaphandler.sh

else

     echo "Detection Error."
fi

#######################################################
#Evasion Countermeasure Filter and Shield Deployment
#######################################################

nmap_source_array=( "$ids_ips_ip" "$ids_enp0s9_ip" "$victim1_enp0s9_ip" "$victim1_ip" "$default_gateway_ip" )


if (( length2 > length1 )); then

    echo "Snort logs analysis also shows fewer source MACs compared to source IPs."
    echo "This indicates potential spoofed decoy IP addresses in the attack."
    echo "Will try to find the original Attacker's IP(s)"
    network_scan "1" "$interface" "${source_macs_sorted[@]}"
    echo

    if [[ ${#not_tracable[@]} -ne 0 ]]; then

       echo "System was unable to trace all malicious MACs back to the corresponding IPs. All identified"
       echo "source IPs will be utilized moving forward (Excluding known pre-Configured spoofed values)"
       echo
       ip_unspoof "${final_source_sorted[@]}"
       sleep 1
       nmap_source_array=( "${nmap_source_array[@]}" "${not_spoofed_array[@]}" )
       destination_ipcheck "$length6" "nmap.sh" "${final_destination_sorted[@]}" "${nmap_source_array[@]}"


    else

       nmap_source_array=( "${nmap_source_array[@]}" "${scanned_ip_array[@]}" )
       echo
       destination_ipcheck "$length6" "nmap.sh" "${final_destination_sorted[@]}" "${nmap_source_array[@]}"


    fi


elif (( length1 == length2 )); then

      nmap_source_array=( "${nmap_source_array[@]}" "${final_source_sorted[@]}" )
      echo
      destination_ipcheck "$length6" "nmap.sh" "${final_destination_sorted[@]}" "${nmap_source_array[@]}"

fi
```

Εικόνα 51. Nmaphandler.sh [2/2]

Όταν διαγνωστεί μια επίθεση, και κατανεμηθεί ως “Nmap Scan”, εκτελείται το Nmaphandler.sh προκειμένου να διεξαχθούν οι απαραίτητες διαδικασίες αντιμετώπισης του κακόβουλου σαρωτή (Εικόνα 41). Αφού ενημερώσει το διαχειριστή ως προς την κατηγορία της επίθεσης που διαγνώστηκε, το σύστημα αντιμετώπισης αδρανοποιείται για 6 δευτερόλεπτα προτού αναστείλει τη λειτουργία του Snort, με σκοπό το τελευταίο να έχει συλλέξει ικανοποιητικό όγκο δεδομένων για την επίθεσή. Η τεχνική αυτή εφαρμόζεται σε όλες τις επιθέσεις, χρησιμοποιώντας κυμαινόμενες τιμές χρονικών πλαισίων οι οποίες προέκυψαν πειραματικά. Στη συνέχεια, το σύστημα κατηγοριοποιεί περεταίρω την επίθεση ανάλογα με τους τύπους σαρώσεων που καλύπτονται από το σύστημα (Stealth/Full, Xmas, Fin, Null), χρησιμοποιώντας την προαναφερόμενη μέθοδο σύγκρισης των alert (first_line) με λέξεις – κλειδιά. Για κάθε περίπτωση, και εφόσον το Snort πλέον έχει τερματιστεί, στο Directory */var/log/snort*, θα έχει σχηματιστεί το Binary File της καταγραφής των κακόβουλων πακέτων, με αποτέλεσμα να χρησιμοποιείται από την process flag(). Όπως αναφέρθηκε και προηγουμένως, τα ορίσματα της συνάρτησης διαφέρουν τόσο στα Flags, όπου κάθε μεταβλητή διαχωρίζει έναν συγκεκριμένο τύπο πακέτου από τους υπόλοιπους, όσο και στα ορίσματα των θέσεων για τις διευθύνσεις που θα αντληθούν. Η σωστές θέσεις των IP και MAC που καλείται η συνάρτηση να αντλήσει από το Tcpdump Output, καθορίζεται σε κάθε περίπτωση πειραματικά. Συνεχίζοντας στη ροή του Nmaphandler Script, αφού έχουν προκύψει οι 4 ζητούμενες Arrays διευθύνσεων το σύστημα συγκρίνει τον αριθμό των κακόβουλων MAC διευθύνσεων, με τις αντίστοιχες IP, προκειμένου να εξετάσει την πιθανότητα χρήσης Decoy διευθύνσεων από τον επιτιθέμενο. Σε περίπτωση που ο έλεγχος δεν επαληθευτεί και προκύψουν λιγότερες MAC, η μονάδα Evasion Countermeasure τις χρησιμοποιεί ως όρισμα στη network_scan() με σκοπό να αντιστοιχηθούν με τις πραγματικές κακόβουλες IP που θα προκύψουν από την εκτέλεσή της. Τέλος, εάν για οποιοδήποτε λόγο (πχ. η συσκευή του επιτιθέμενου αποκλείει την κυκλοφορία που προέρχεται από το IDS/IPS) δεν καθίσταται δυνατή η ιχνηλάτηση όλων των διαθέσιμων διευθύνσεων, το σύστημα χρησιμοποιεί τη συνάρτηση ip_unspoof() προκειμένου να αποσπάσει τυχών γνωστές παραποιημένες (Spoofed) διευθύνσεις IP από το Array των κακόβουλων. Ανεξάρτητα από τις επιμέρους υποπεριπτώσεις, στην έκβαση της ροής καλείται η συνάρτηση destination_ipcheck() προκειμένου το Shield nmap.sh να μεταφερθεί και να εκτελεστεί στις μηχανές - θύματα που υπάγονται στο Array final_destination_sorted. Όσον αφορά το nmap_source_array, αποτελείται από διευθύνσεις γνωστών συσκευών που είναι απαραίτητες κατά την εκτέλεση του παραπάνω Script, καθώς και όλες τις κακόβουλες IP που έχουν προκύψει από την ανάλυση στα επιμέρους σενάρια.

Nmap.sh

```
  GNU nano 6.2
#!/bin/bash

#Shield For Nmap Scans

ids_nmap=$1
shift
ids_e9_nmap=$1
shift
victim1_e9_nmap=$1
shift
victim1_nmap=$1
shift
gw_nmap=$1
shift
nmap_source_array=("$@")

victim1="$victim1_nmap"

trusted_sources=( "$victim1_nmap" "$ids_nmap" "$ids_e9_nmap" "$victim1_e9_nmap" "$gw_nmap" )

flag=0

#####################################
#Block the Malicious IP Addresses
#####################################

if [ "${#nmap_source_array[@]}" -ne 0 ]; then

   echo
   echo
   echo "Banning Intruders IP(s)"
   echo

   for source_ip_nmap in "${nmap_source_array[@]}"; do

       sudo iptables -A INPUT -s "$source_ip_nmap" -j DROP &&
       sudo iptables -A OUTPUT -d "$source_ip_nmap" -j DROP &&
       sudo iptables -A FORWARD -d "$source_ip_nmap" -j DROP &&
       sudo iptables -A FORWARD -s "$source_ip_nmap" -j DROP

   done

   echo "Banned Successfully"
   echo
   echo

else
      echo "NO IP(s) TO BAN?"

fi


##############################################
#Open Port(s) Check and Filter Application
##############################################

apply_filter() {

    local port=$1
    local service=$2

    echo "-> Found Port $port (Service $service) Open <-"
    echo

    while true; do

        read -p "Do you want to filter this port for trusted connections only? (yes/no): " user_input

        if [ "$user_input" = "yes" ]; then

            echo
```

Εικόνα 52. Nmap.sh Shield [1/2]

```
  GNU nano 6.2
        if [ "$user_input" = "yes" ]; then

            echo
            echo "Applying filter on port $port..."
            echo

            for trusted_source in "${trusted_sources[@]}"; do
                sudo iptables -A INPUT -p tcp -s "$trusted_source" --dport "$port" -j ACCEPT
                sudo iptables -A OUTPUT -p tcp -d "$trusted_source" --sport "$port" -j ACCEPT
            done

            sudo iptables -A INPUT -p tcp --dport "$port" -j DROP
            sudo iptables -A OUTPUT -p tcp --sport "$port" -j DROP &&

            echo "Done. Make sure to Drop the Filter once the Threat is Mitigated"
            echo
            echo

            break

        elif [ "$user_input" = "no" ]; then

            echo
            echo "No action taken. Port stays unfiltered for potentially malicious Connections..."
            echo
            echo
            break


        else

            echo
            echo "Invalid input. Please type 'yes' or 'no'."
            echo

        fi

    done
}

echo "Searching for some Default Open Ports on the Machine..."
echo

sleep 1

nmap_output=$(sudo nmap -sT $victim1)

if echo "$nmap_output" | grep -q '22/tcp'; then

    flag=$((flag + 1))

    apply_filter 22 SSH

fi

if echo "$nmap_output" | grep -q '80/tcp'; then

    flag=$((flag + 1))

    apply_filter 80 HTTP

fi

if [ "$flag" -eq 0 ]; then

  echo
  echo "Found 0 Usually Open Ports."

fi
```

Εικόνα 53. Nmap.sh Shield [2/2]

Στο αρχικό στάδιο εκτέλεσης του Shield nmap.sh το σύστημα διαχωρίζει τις διευθύνσεις του Array που δόθηκαν ως όρισμα από την προηγούμενη μονάδα σε δύο αντίστοιχες έμπιστων (trusted_sources), και κακόβουλων (nmap_source_array) IP. Στη συνέχεια χρησιμοποιείται το Utility Iptables για την απόκλιση των πακέτων που υπάγονται σε οποιοδήποτε είδος επικοινωνίας με τις διευθύνσεις του επιτιθέμενου. Συγκεκριμένα, το εργαλείο λειτουργεί ως τείχος προστασίας των μηχανών, απορρίπτοντας όλη την εισερχόμενη, εξερχόμενη αλλά και προωθούμενη δικτυακή κυκλοφορία που ταυτίζεται με τις παραπάνω κακόβουλες διευθύνσεις. Τέλος, εφαρμόζεται σάρωση στην ίδια τη μηχανή – θύμα προκειμένου να ανιχνευθούν κάποιες βασικές ανοιχτές θύρες, όπως η 22 και η 80. Χρησιμοποιώντας τη συνάρτηση apply_filter() και με τη συγκατάθεση του διαχειριστή, εφαρμόζεται φίλτρο για τις θύρες που θα επιλεχθούν, επιτρέποντας τη διέλευση εισερχόμενης και εξερχόμενης κυκλοφορίας μόνο στις διευθύνσεις του trusted_sources Array. Παράλληλα τα πακέτα από οποιαδήποτε άλλη IP (όχι απαραιτήτως κακόβουλη) που απευθύνονται στα εν λόγω Ports απορρίπτονται από το σύστημα.

Bruteforcehandler.sh

```
  GNU nano 6.2                                                    bruteforcehandler.sh
#!/bin/bash

#Handler For Brute Force Attacks

##############################
#Further Attack Verification
##############################

first_line=$(echo "$new_lines" | head -n 1)
sleep 8

file_path_bruteforce="/var/log/snort/alert"
line_number=$(grep -nF "$first_line" "$file_path_bruteforce" | cut -d: -f1)
final_lines_bruteforce=$(awk "NR > $line_number" "$file_path_bruteforce" | wc -l)

if [ "$final_lines_bruteforce" -gt 3 ]; then

   echo
   echo " ^^^^ SSH Password Authentication Bruteforce Attempt ^^^^ "
   echo
   sudo kill -15 $SNORT_PID
   sleep 1

else

   echo "Detection Error: Insufficient lines for Bruteforce SSH Password Authentication detection."
   echo "Exiting program."
   exit 1

fi

#######################################################
#Source & Target Identification and Shield Deployment
#######################################################

process_flag "SSH" "14" "16" "2" "6"
sleep 1
destination_ipcheck "$length6" "bruteforce.sh" "${final_destination_sorted[@]}" "${final_source_sorted[@]}"
```

Εικόνα 54. Bruteforcehandler.sh

Προχωρώντας στο Script bruteforcehandler.sh, προτού το σύστημα ενημερώσει το διαχειριστή για τον τύπο της επίθεσης, αδρανοποιείται για 8 δευτερόλεπτα και στη συνέχεια ξανά ελέγχει το snort/alert File προκειμένου να διαπιστώσει εάν πρόκειται για πραγματική παραβίαση της ασφάλειας, ή για ψευδώς θετικό αποτέλεσμα. Όπως αναφέρθηκε και στην ανάλυση του Snort Ruleset, ο κανόνας 8 που καλύπτει απειλές αντίστοιχου τύπου, ειδοποιεί το σύστημα IPS όταν οποιαδήποτε διεύθυνση που δεν ανήκει στο προστατευόμενο σύνολο επιχειρήσει να εδραιώσει σύνδεση SSH με συσκευή εντός αυτού. Το κατώφλι των τριών αποπειρών σε οχτώ δευτερόλεπτα τέθηκε προκειμένου να διαχωρίσει μια κατάσταση Brute Force από τα ανεκτά επίπεδα κυκλοφορίας. Σε περίπτωση όπου ο έλεγχος επαληθευτεί, το σύστημα χρησιμοποιεί τη συνάρτηση process_flag(), για να εντοπίσει κυρίως τις IP των επιτιθέμενων και των θυμάτων. Με αυτές είναι σε θέση να καλέσει την destination_ipcheck(), δίνοντας ως όρισμα αυτή τη φορά το Shield bruteforce.sh. Σημειώνεται ότι η φύση της επίθεσης δεν επιτρέπει τη χρήση των Spoofed διευθύνσεων, με αποτέλεσμα να αποφεύγεται η χρήση της μονάδας Evasion Countermeasure.

Bruteforce.sh

Στο Script που αναλαμβάνει να θωρακίσει το θύμα μιας επίθεσης Brute Force, αρχικά ελέγχεται εάν ο επιτιθέμενος έχει ήδη παραβιάσει τη συσκευή, χρησιμοποιώντας το Linux Utility Who. Το τελευταίο μεταξύ άλλων, παρέχει πληροφορίες σχετικά με τους χρήστες που είναι συνδεδεμένοι με το σύστημα τη δεδομένη χρονική στιγμή, και είναι ιδανικό για τέτοιου είδους διεργασίες. Εάν ο έλεγχος επιβεβαιωθεί, τότε το σύστημα αποπέμπει γρήγορα τον εισβολέα (kill -9 $pid). Στη συνέχεια, τροποποιείται το βασικό Configuration αρχείο του SSH Deamon (Password Authentication – no), προκειμένου η συσκευή να απορρίπτει μελλοντικές απόπειρες συνδέσεων που βασίζονται σε κωδικό πρόσβασης. Τέλος, οι κακόβουλες διευθύνσεις που προέκυψαν από τις προηγούμενες μονάδες εισάγονται στους κανόνες απόρριψης του Iptables, συμβάλλοντας έτσι στην παρεμπόδιση της επίθεσης και τον αποκλεισμό του εισβολέα. Η διάρθρωση του Script παρουσιάζεται στις επόμενες δύο Εικόνες.

```
  GNU nano 6.2
#!/bin/bash

#Shield For SSH Authentication Bruteforce Attacks

source_ips_br=("$@")
SSHD_CONFIG="/etc/ssh/sshd_config"

##################################################
#Check Potential Present or Past Connections
##################################################

for ip in "${source_ips_br[@]}"; do

  echo
  echo "---> Processing Malicious IP: $ip <---"
  echo

  pts=$(who | grep "$ip" | awk '{print $2}')

  if [ -z "$pts" ]; then

     echo
     echo "Attacker is not currently connected to the Machine via SSH..."

  else

     echo
     echo "(!) The Attacker is conneted to the Machine via SSH now (!)"
     echo
     echo "Killing the malicious connection.."
     pid=$(ps aux | grep "sshd" | grep "$pts" | awk '{print $2}')
     sudo kill -9 $pid
     echo
     echo "Done. Attacker Kicked off the System!"
     echo

  fi

done
```

Εικόνα 55. Bruteforce.sh Shield [1/2]

```
##################################################
#Disable Password Authentication For SSH Connections
##################################################

echo
echo
echo "--- Disabling Password Authentication For SSH Connections ---"
sudo sed -i 's/^#*PasswordAuthentication .*/PasswordAuthentication no/' $SSHD_CONFIG
sudo systemctl restart sshd
sleep 1

if grep -q "^PasswordAuthentication no" "$SSHD_CONFIG"; then

    echo
    echo "Password authentication has been successfully disabled for SSH."
    echo
    echo

else

    echo
    echo "Failed to disable password authentication for SSH."
    echo
    echo

fi

#################################
#Block the Malicious IP Addresses
#################################

echo "--- Blocking Intruders IP(s) ---"

for malipbr in "${source_ips_br[@]}"; do

  sudo iptables -A INPUT -s $malipbr -j DROP &&
  sudo iptables -A OUTPUT -d $malipbr -j DROP &&
  sudo iptables -A FORWARD -d $malipbr -j DROP &&
  sudo iptables -A FORWARD -s $malipbr -j DROP &&

done

echo
echo "Intruder Blocked. Make sure you enable SSH Password Authentication (if needed) after the Attack is completely Mitigated."
```

Εικόνα 56. Bruteforce.sh Shield [2/2]

Arpspoofhandler.sh

```
  GNU nano 6.2                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                
arpspoofhandler.sh
#!/bin/bash

#Handler For ARP Spoofing Attacks

echo
echo
echo "--- Potential ARP Spoof Detected ---"
echo
sleep 6
sudo kill -15 $SNORT_PID
sleep 1

##################################
#Source & Target Identification
##################################

echo
echo
echo "Anilyzing Snort Logs to Extract Crucial Information about the Attack"

process_flag "ARP" "6" "17" "2" "6" > /dev/null 2>&1

echo
echo
echo "Logs Check Came back with the Following:"
echo
echo "Source MAC(s): ${source_macs_sorted[@]}"
echo
echo "Destination -- Protected Network -- MAC(s): ${home_macs_sorted[@]}"
echo

###########################
#Attack Specific Actions
###########################

mac2ip_convertor "${home_macs_sorted[@]}"
sleep 1
length4=${#protected_macs_converted[@]}

for maccheck in "${home_macs_sorted[@]}"; do

    if [ "$maccheck" == "$router_mac" ]; then

            echo "!Router's Cache is Probably Poisoned! Preparing System to act as the Default Gateway if needed."
            sudo sysctl -w net.ipv4.ip_forward=1 > /dev/null 2>&1
            sudo iptables -t nat -A POSTROUTING -s $victim1_ip -o $interface -j MASQUERADE
            echo
            echo
            echo "--- System is Ready ---"
            break

    fi

done

network_scan "1" "$interface" "${source_macs_sorted[@]}"
sleep 1
length3=${#scanned_ip_array[@]}

#####################
#Shield Deployment
#####################

unified_arpspoof=( "${scanned_ip_array[@]}" "${source_macs_sorted[@]}" )
unified_arpspoof=( "$interface" "$ids_ips_mac" "$victim1_mac" "$router_mac" "$victim1_ip" "$ids_ips_ip" "$default_gateway_ip" "$length3" "${unified_arpspoof[@]}" )

destination_ipcheck "$length4" "arpspoof.sh" "${protected_macs_converted[@]}" "${unified_arpspoof[@]}"
```

Εικόνα 57. Arpspoofhandler.sh

Όσον αφορά τη διαχείριση επιθέσεων ARP Spoofing από τον αντίστοιχο handler, αφού αναστείλει τη λειτουργία του Snort χρησιμοποιεί τη συνάρτηση process_flag() δίνοντας έμφαση αυτή τη φορά στις διευθύνσεις MAC των αποστολέα και παραλήπτη. Το τελευταίο συμβαίνει διότι το Tcpdump περιορίζεται σε ανάλυση πακέτων επιπέδου 2 (frames), τα οποία δεν διαθέτουν αρκετές πληροφορίες σχετικά με τις διευθύνσεις IP των εμπλεκόμενων μηχανών. Συμπερασματικά, χρειάζεται κάποιος μηχανισμός αυτόματης μετάφρασης των MAC σε IP τόσο των επιτιθέμενων όσο και των θυμάτων. προκειμένου να είναι σε θέση να χρησιμοποιηθούν αργότερα από την destination_ipcheck(). Τη λύση στο πρόβλημα δίνει η χρήση της συνάρτησης mac2ip_convertor() για τις διευθύνσεις MAC των θυμάτων, προκειμένου αυτή να επιστρέψει τις ανάλογες IP. Παράλληλα, για αυτές των επιτιθέμενων καλείται η network_scan() προκειμένου να

συντελέσει στο ίδιο αποτέλεσμα. Η μονάδα Attack Specific Actions χρησιμοποιείται στο συγκεκριμένο handler, ως έλεγχος στις – μεταφρασμένες πλέον – IP των θυμάτων. Εάν διαπιστωθεί ότι σε μια Full Duplex επίθεση η μια εκ των δύο επηρεασμένων συσκευών είναι το Router (έχει δηλαδή υποστεί ARP Cache Poisoning), η μηχανή IDS/IPS προετοιμάζεται προκειμένου να λειτουργήσει ως προσωρινό Default Gateway του ετέρου θύματος. Σκοπός αυτού είναι να αποφευχθεί το άμεσο κανάλι επικοινωνίας μεταξύ των δύο, το οποίο έχει παραβιαστεί και παρακολουθείται από τον εισβολέα, με τον περιορισμό ότι δεν είναι δυνατή η διόρθωσης των καταγραφών στην ARP Cache ενός Virtual Box Router. Στο IDS/IPS λοιπόν, ενεργοποιείται η προώθηση δικτυακής κυκλοφορίας (ip_forward=1) και παράλληλα ρυθμίζεται το Iptables προκειμένου να εφαρμόζεται Network Address Translation για κάθε πακέτο που πρόκειται να στείλει εκ μέρους του δεύτερου θύματος. Τροποποιώντας (masquerade) τη διεύθυνση IP αποστολέα και εισάγοντας τη δική του, το IDS/IPS θα αποτελεί μεσάζοντα στην επικοινωνία του με το Router, διασφαλίζοντας έτσι ότι δεν θα χρησιμοποιηθεί η “Poisoned” καταγραφή στην μνήμη του δρομολογητή. Τέλος, προτού ενεργοποιηθεί η συνάρτηση destination_ipcheck() δημιουργείται το Array unified_arpspoof που θα χρησιμοποιηθεί ως όρισμά της , και περιέχει τις κακόβουλες MACs και IPs, καθώς και ορισμένες τιμές δικτύου που είναι απαραίτητες για τη λειτουργία του ομώνυμου Shield.

Arpspoof.sh

Στο συγκεκριμένο Script, αρχικά διαχωρίζονται οι κακόβουλες MAC και IP διευθύνσεις σε ισάριθμα Arrays, όπως αυτές προέκυψαν από τις εξόδους του Handler (ips_arpspoof_array macs_arpspoof_array). Στη συνέχεια, ενεργοποιείται η συνάρτηση check_arp_tables(), η οποία δέχεται ως Input το σύνολο των διευθύνσεων που υπάρχουν τη δεδομένη στιγμή στην ARP Cache του θύματος, και τις ελέγχει για τυχών αποκλίσεις από τις αναμενόμενες τιμές. Εάν για οποιοδήποτε ζευγάρι διαγνωστεί ύποπτος συνδυασμός, τότε η Cache τροποποιείται εκ νέου, με σκοπό την αντικατάσταση της κακόβουλης MAC με την αναμενόμενη τιμή της. Βασικός άξονας για την επίτευξη του ελέγχου αποτελεί η Associative Array expected_macs, η οποία δημιουργείται χρησιμοποιώντας τις γνωστές διευθύνσεις των συσκευών του δικτύου που αντλήθηκαν νωρίτερα από τον Handler. Η δομή της είναι πανομοιότυπη με τα υφιστάμενα ζευγάρια τιμών στον Arpspoof Preprocessor, συσχετίζοντας τις IPs (Keys) με τις αντίστοιχες MACs (Values) των προστατευόμενων μηχανών. Το σύστημα επιστρέφει στην έξοδο κατάλληλο διαγνωστικό μήνυμα για κάθε αναμενόμενη (ή μη) αντιστοιχία διευθύνσεων, και αναπροσαρμόζει τις καταγραφές της ARP Cache καταλλήλως (sudo ip neigh replace "$ip" lladdr "$expected_mac" dev $interface_arpspoof).

```
  GNU nano 6.2
#!/bin/bash

#Shield For ARP Spoofing Attacks

interface_arpspoof=$1
shift
idsmac_arpspoof=$1
shift
victim1mac_arpspoof=$1
shift
gwmac_arpspoof=$1
shift
victim1_arpspoof=$1
shift
ids_arpspoof=$1
shift
gw_arpspoof=$1
shift
separator_arpspoof=$1
shift
combinedall_arpspoof=("$@")

for ((j=0; j<separator_arpspoof; j++)); do
     ips_arpspoof_array+=("${combinedall_arpspoof[j]}")
done

for ((k=separator_arpspoof; k<${#combinedall_arpspoof[@]}; k++)); do
     macs_arpspoof_array+=("${combinedall_arpspoof[k]}")
done

#################################################################
#Check For Inconsistencies in the ARP Cache Refit If Needed
#################################################################

indexOf() {
        local element="$1"
        shift
        local array=("$@")

        for i in "${!array[@]}"; do
            if [ "${array[$i]}" == "$element" ]; then
                echo "$i"
                return 0
            fi
        done

        echo "-1"
}

check_arp_table() {

    echo
    echo "Checking the System's ARP Cache Based on the Pre - Configured Values.."
    echo
    echo
    current_ip_macs=("$@")

    declare -A expected_macs
    expected_macs["$ids_arpspoof"]="$idsmac_arpspoof"
    expected_macs["$gw_arpspoof"]="$gwmac_arpspoof"
    expected_macs["$victim1_arpspoof"]="$victim1mac_arpspoof"

    ip_array=()
    mac_array=()
```

Εικόνα 58. Arpspoof.sh Shield [1/3]

```
GNU nano 6.2                                                                    arpspoof.sh
    mac_array=()

    for ((i=0; i<${#current_ip_macs[@]}; i+=2)); do

    ip_without_parentheses=$(echo "${current_ip_macs[i]}" | tr -d '()')
    ip_array+=("$ip_without_parentheses")
    mac_array+=("${current_ip_macs[i+1]}")

    done

    for ip in "${!expected_macs[@]}"; do
          expected_mac="${expected_macs[$ip]}"
          index=$(indexOf "$ip" "${ip_array[@]}")

            if [ "$index" -ne -1 ]; then

                if [ "${mac_array[$index]}" == "$expected_mac" ]; then

                    echo "---- Expected Match found for $ip - ${mac_array[$index]}. No action needed on this Pair ----"
                    echo
                    echo

                else

                    echo "!! Mismatch for $ip: ${mac_array[$index]} (Expected: $expected_mac) !!"
                    mismatch_mac_array+=("${mac_array[$index]}")
                    echo
                    echo

                    if [ "$ip" == "$gw_arpspoof" ]; then

                       ##############################################################
                       #Router's Cache is Poisoned - Redirect Traffic through IDS/IPS
                       ##############################################################

                       echo "--> Special handling for the Gateway's IP - $ip <---"
                       echo
                       echo "Redirecting all traffic for this machine through the IDS/IPS to avoid the routers Poisoned Cache..."
                       echo
                       sudo iptables -A INPUT -s $gw_arpspoof -j DROP
                       sudo iptables -A OUTPUT -d $gw_arpspoof -j DROP
                       sudo iptables -A FORWARD -s $gw_arpspoof -j DROP
                       sudo iptables -A FORWARD -d $gw_arpspoof -j DROP
                       sudo ip route del default via $gw_arpspoof dev $interface_arpspoof
                       sudo ip route add default via $ids_arpspoof dev $interface_arpspoof
                       echo "Done. Make sure you revert back to normal Network Configurations when the attack is completely mitigated (Network Congestion)"
                       echo
                       echo
                       echo "Refitting the System's ARP Cache Entries"
                       echo
                       sudo ip neigh replace "$ip" lladdr "$expected_mac" dev $interface_arpspoof
                       echo "Malicious Mac for $ip Dropped and Changed back to to $expected_mac"
                       echo
                       echo

                    else

                       echo "Handling $ip..."
                       echo
                       echo "Refitting the System's ARP Cache Entries"
                       echo
                       sudo ip neigh replace "$ip" lladdr "$expected_mac" dev $interface_arpspoof
                       echo "Malicious Mac for $ip Dropped and Changed back to to $expected_mac"
                       echo
                       echo

                    fi
                fi
```

Εικόνα 59 Arpspoof.sh Shield [2/3]

```
                    fi
                fi


            else

                    echo "---- IP $ip not found in current ARP Cache. No action needed ----"
                    echo
                    echo
            fi
    done

}

current_ipmacs=($(arp -a -i $interface_arpspoof | awk '/ether/ {print $2, $4}'))
sleep 1
check_arp_table "${current_ipmacs[@]}"

##########################################
#Block the Malicious IP and MAC Addresses
##########################################

if [ "${#ips_arpspoof_array[@]}" -ne 0 ]; then

     echo "Blocking known Intruders IP(s).."
     echo

     for intruderspotip in "${ips_arpspoof_array[@]}"; do

          sudo iptables -A INPUT -s "$intruderspotip" -j DROP
          sudo iptables -A OUTPUT -d "$intruderspotip" -j DROP
          sudo iptables -A FORWARD -s "$intruderspotip" -j DROP
          sudo iptables -A FORWARD -d "$intruderspotip" -j DROP

     done

           echo "IP(s) Blocked"
           echo
           echo

fi

final_malicious=("${mismatch_mac_array[@]}" "${macs_arpspoof_array[@]}")
final_malicious_sorted=($(printf "%s\n" "${final_malicious[@]}" | sort -u))

echo "Blocking known  Intruder's Mac Address..."

for maliciousmacsfinal in "${final_malicious_sorted[@]}"; do

   sudo arptables -A INPUT --src-mac "$maliciousmacsfinal" -j DROP
   sudo arptables -A OUTPUT --dst-mac "$maliciousmacsfinal" -j DROP

done

echo
echo "MAC(s) Blocked."
echo
echo
echo
```

Εικόνα 60. Arpspoof.sh. Shield [3/3]

Όπως προαναφέρθηκε, σε περίπτωση που ο δρομολογητής εμπεριέχεται στο σύνολο παραβιασμένων συσκευών της επίθεσης, ακολουθείται ειδικός χειρισμός όσον αφορά τις ρυθμίσεις δικτύου του ετέρου θύματος. Αρχικά χρησιμοποιώντας το Utility Iptables αποκόπτεται οποιοδήποτε είδος άμεσης ανταλλαγής πακέτων μεταξύ του τελευταίου και του Router, προκειμένου να αποφευχθεί η χρήση του παραβιασμένου καναλιού επικοινωνίας που παρακολουθείται από τον εισβολέα. Στη συνέχεια, για να εξασφαλισθεί ασφαλής πρόσβαση στο Internet υπό τις παραπάνω, αντίξοες συνθήκες, πραγματοποιούνται συγκεκριμένες αναπροσαρμογές στον πίνακα δρομολόγησης του θύματος (Routing Table). Διαγράφεται η προεπιλεγμένη διαδρομή προς εξωτερικά δίκτυα μέσω του Router και αντικαθίσταται από μια αντίστοιχη μέσω της μηχανή IDS/IPS. Η ρύθμιση αυτή, σε συνδυασμό με αυτή που πραγματοποιήθηκε νωρίτερα κατά την εκτέλεση του Handler, εξασφαλίζει την αποφυγή της Poisoned ARP Cache του Router, καθώς πλέον το IDS/IPS λειτουργεί ως Default Gateway των θυμάτων. Τέλος, οι κακόβουλες διευθύνσεις IP που βρίσκονται στην ips_arpspoof_array προστίθενται στους κανόνες απόρριψης του Iptables για όλα τα είδη κυκλοφορίας. Παράλληλα, οι αντίστοιχες MAC που προέκυψαν από τον έλεγχο αναντιστοιχιών, αλλά και αυτές που

“έστειλαν” τα κακόβουλα πακέτα, συγκεντρώνονται στην final_malicious και αφού ταξινομηθούν ως προς τις διπλότυπες καταγραφές, εισάγονται στους κανόνες απόρριψης του Arptables για την εισερχόμενη και εξερχόμενη κυκλοφορία. Το τελευταίο λειτουργεί παρόμοια με το Iptables, αλλά είναι ειδικά σχεδιασμένο για τον χειρισμό πακέτων ARP.

Redirectionhandler.sh

```
  GNU nano 6.2
#!/bin/bash

#Handler For ICMP Redirection Attacks

echo
echo "ICMP Redirect Packet(s) Detected *** Could be Malicious ***"
echo
sleep 6
sudo kill -15 $SNORT_PID
sleep 1

##################################
#Source & Target Identification
##################################

process_flag "redirect" "14" "16" "2" "22"
sleep 1
malicious_ips=("${home_macs_sorted[@]}")

#######################################################
#Evasion Countermeasure Filter and Shield Deployment
#######################################################

echo
echo "Checking source IP(s) against pre-Configured Network Variables"
echo "and running a Network Scan to sort out the Spoofed IP(s)"
echo
ip_unspoof "${final_source_sorted[@]}"
sleep 1
redirection_source_array=( "$ids_ips_ip" "$default_gateway_ip" )


if [[ ${#not_spoofed_array[@]} -ne 0 ]]; then

   network_scan "2" "$interface" "${not_spoofed_array[@]}" > /dev/null 2>&1
   sleep 1

   if [[ ${#not_tracable[@]} -ne 0 ]]; then

      echo "Unable to pinpoint every spoofed IP with certainty!! All identified source IPs"
      echo "will be utilized moving forward (Excluding pre-Configured spoofed values)"
      echo

      redirection_source_array=( "${redirection_source_array[@]}" "${not_spoofed_array[@]}" )

      destination_ipcheck "$length6" "icmpredirect.sh" "${final_destination_sorted[@]}" "${redirection_source_array[@]}"

   else

      echo "Finished searching. Will utilize Source IP(s): ${reverse_ips[@]}"
      echo

      redirection_source_array=( "${redirection_source_array[@]}" "${scanned_ip_array[@]}" )

      destination_ipcheck "$length6" "icmpredirect.sh" "${final_destination_sorted[@]}" "${redirection_source_array[@]}"

   fi


else

   echo "All Source IP(s) match known network IP(s). Definitely spoofed!"
   echo

   destination_ipcheck "$length6" "icmpredirect.sh" "${final_destination_sorted[@]}" "${redirection_source_array[@]}"

fi
```

Εικόνα 61. Redirectionhandler.sh

Στην αρχή της εκτέλεσης του Redirectionhandler καλείται η process_flag(), αυτή τη φορά δίνοντας έμφαση στην άντληση IP των επιτιθέμενων και προστατευόμενων μηχανών, αλλά και των κακόβουλων διευθύνσεων Gateway που οι εισβολείς συμπεριλαμβάνουν στα συγκεκριμένα πακέτα. Οι τιμές αυτές ενδέχεται να συμπίπτουν, δηλαδή η IP του αποστολέα να είναι ίδια με αυτή που επιχειρεί να εισάγει στους πίνακες δρομολόγησης των θυμάτων ως Gateway για συγκεκριμένους προορισμούς, ωστόσο στο ανάλογο Shield (Icmpredirect.sh) πραγματοποιείται έλεγχος διαγραφής των διπλότυπων διευθύνσεων. Αφού λοιπόν έχουν προκύψει οι κακόβουλες διευθύνσεις αποστολέα (final_source_sorted), εισάγονται ως όρισμα στη συνάρτηση φιλτραρίσματος ip_unspoof() προκειμένου να συγκριθούν με τις τιμές γνωστών διευθύνσεων του δικτύου, και κυρίως του Router, καθώς τα πακέτα τύπου ICMP Redirection στέλνονται κυρίως από εξουσιοδοτημένους δρομολογητές. Εάν ο έλεγχος επιστρέψει όλες τις διευθύνσεις αποστολέα ως σίγουρα πλαστογραφημένες (Spoofed IP Addresses), τότε το σύστημα προχωράει στην πρόωρη εκτέλεση της destination_ipcheck(), χωρίς συγκεκριμένο Input κακόβουλων IP αποστολέα. Στην αντίθετη έκβαση του παραπάνω ελέγχου, και για τις IP που πέρασαν από το φίλτρο (not_spoofed_array), το σύστημα για να επιβεβαιώσει ότι πρόκειται για πραγματικές διευθύνσεις, τις χρησιμοποιεί ως εκ νέου όρισμα στη network_scan(), επιχειρώντας αντίστροφη σάρωση (“2”). Επιβεβαιώνεται ότι οι διευθύνσεις αποστολέα δεν είναι πλαστογραφημένες (Spoofed) μόνο στην περίπτωση όπου όλες ανταποκριθούν στο Scan με την αντίστοιχη MAC τους. Σημειώνεται ότι σε οποιαδήποτε έκβαση της βασικής ροής σεναρίων, η συνάρτηση destination_ipcheck(), μεταξύ άλλων, θα καλεστεί με τιμές εισόδου τις διευθύνσεις του IDS/IPS και του Router καθώς είναι απαραίτητες για την εκτέλεση του Icmpredirect.sh Script (redirection_source_array).

Icmpredirect.sh

Η πρώτη ενέργεια του συστήματος κατά τη λειτουργία του Shield στα θύματα, είναι η ανάλυση του πίνακα δρομολόγησής τους (ip route show cache). Όλα τα συστήματα Linux διατηρούν κρυφές μνήμες συγκεκριμένων καταγραφών δρομολόγησης σε εξωτερικούς προορισμούς μέσω αντίστοιχων Gateways. Οι μοναδικοί Hosts του δικτύου που είναι εξουσιοδοτημένοι να διεκπεραιώνουν ενέργειες δρομολόγησης πακέτων από τις υπόλοιπες μηχανές είναι το Router και το IDS/IPS. Συνεπώς, το σύστημα έχοντας λάβει από τον προηγούμενο Handler τις διευθύνσεις τους, τις συγκρίνει με αυτές που προέκυψαν ως Gateways εξωτερικών προορισμών από την ανάλυση του Routing Table των θυμάτων. Όταν βρεθεί κάποια IP που αποκλίνει από τις παραπάνω τιμές, αποθηκεύεται στο Array malicious_gws. Ακολουθεί αποκλεισμός των παραπάνω χρησιμοποιώντας ξανά το Iptables, ενώ όπως εξηγήθηκε και παραπάνω, στην περίπτωση όπου προκύψουν διαφορετικές διευθύνσεις αποστολέα, προστίθενται και αυτές στους κανόνες απόκλισης του

εργαλείου. Τέλος, στα θύματα πραγματοποιείται ολική εκκαθάριση της κρυφής μνήμης δρομολογήσεων (sudo ip route flush cache), καθώς και αναδιαμόρφωση των παραμέτρων δικτύωσης στο Kernel τους, προκειμένου να απορρίπτεται πρώιμα οποιοδήποτε εισερχόμενο πακέτο τύπου ICMP Redirection.

```
  GNU nano 6.2
#!/bin/bash

#Shield For ICMP Redirection Attacks

ips=$1
shift
router=$1
shift
source_ips_of_redirection=("$@")

#######################################################################
#Routing Cache Check and Block Potentially Malicious IP Addresses
#######################################################################

echo
echo "--> Checking System's Routing Cache <--"
echo
echo

while IFS= read -r cache_line; do

    destinations=$(echo "$cache_line" | awk -F'[: ]' '{print $1}')
    gateways=$(echo "$cache_line" | awk -F'[: ]' '{print $3}')
    destination_arrays+=("$destinations")
    gateway_array+=("$gateways")

done <<< "$(ip route show cache | grep 'via')"

gateway_array_sorted=($(for gw in "${gateway_array[@]}"; do echo "${gw}"; done | sort -u))

echo "Check Reported back with.."
echo
echo "Destination IP(s) - ${destination_arrays[@]} -"
echo "Are being redirected to Gateway(s): ${gateway_array_sorted[@]}"
echo


for pot_mal_gw in "${gateway_array_sorted[@]}"; do

    if [[ "$pot_mal_gw" != "$ips" && "$pot_mal_gw" != "$router" ]]; then

        malicious_gws+=("$pot_mal_gw")

    fi

done


if [ ${#malicious_gws[@]} -eq 0 ]; then

    echo "---- Traffic is being Redirected to Secure destination IP(s) ---- "
    echo "---- No action needed (Potential Network setup misconfiguration!) ---- "
    echo "Exiting program."
    exit 1
    echo
    echo

else

    sudo iptables -A INPUT -p icmp --icmp-type redirect -j DROP

    for mal_gw in "${malicious_gws[@]}"; do

        echo "-> $mal_gw is not a trusted Gateway address! Blocking it now <-"

        sudo iptables -A INPUT -s $mal_gw -j DROP &&
        sudo iptables -A OUTPUT -d $mal_gw -j DROP &&
        sudo iptables -A FORWARD -d $mal_gw -j DROP &&
        sudo iptables -A FORWARD -s $mal_gw -j DROP

    done
    echo
    echo "Intruder(s) Blocked!"
```

Εικόνα 62. Icmpredirect.sh Shield [1/2]

```
    echo
    echo "Intruder(s) Blocked!"
    echo

fi


for moreip in "${source_ips_of_redirection[@]}"; do

  if [[ ! " ${malicious_gws[@]} " =~ " ${moreip} " ]]; then

    more_ips_array+=("$moreip")

  fi

done


if [ ${#more_ips_array[@]} -ne 0 ]; then

    echo
    echo "Also Blocking the sender's IP(s) as they do not match the malicious"
    echo "IP(s) injected into the system's routing cache."

    for ip_more in "${more_ips_array[@]}"; do

      sudo iptables -A INPUT -s $ip_more -j DROP &&
      sudo iptables -A OUTPUT -d $ip_more -j DROP &&
      sudo iptables -A FORWARD -d $ip_more -j DROP &&
      sudo iptables -A FORWARD -s $ip_more -j DROP

    done
    echo
    echo "Blocked!"
    echo

fi

#######################################################################
#Flush the Routing Cache and Configure the Kernel to Drop Redirect Packets
#######################################################################

echo
echo "### Flushing Routing Cache and Configuring Kernel to Drop ICMP Redirection Packets From all Insecure Sources ###"
echo "Make sure to change this Configuration once the threat is Mitigated"

sudo ip route flush cache
sudo sysctl -w net.ipv4.conf.all.accept_redirects=0 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.default.accept_redirects=0 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.enp0s3.accept_redirects=0 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.lo.accept_redirects=0 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.all.secure_redirects=1 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.default.secure_redirects=1 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.enp0s3.secure_redirects=1 > /dev/null 2>&1 &&
sudo sysctl -w net.ipv4.conf.lo.secure_redirects=1 > /dev/null 2>&1
echo
echo "Done. Traffic will be redirected to the Default Gateway."
```

Εικόνα 63. Icmpredirect.sh Shield [2/2]

Doshandler.sh

```
GNU nano 6.2                                                                    doshandler.sh
#!/bin/bash

#Handler For DoS Attacks

##############################################################################
#Attack Specific Action - Potentially Power Off Protected System Under Attack
##############################################################################

dos_function() {

     victims_dos=("$@")

     while true; do

        read -p "** Recommended ** Do you want to shut down the System(s) under Attack? (yes/no): " user_input

        if [ "$user_input" == "yes" ]; then
           echo
           echo

           for victimdos in "${victims_dos[@]}"; do

               if [ "$victimdos" = "$victim1_ip" ]; then

                  system_to_shutdown+=("victim1")

               else

                   echo "Detection Error!!"
                   echo

               fi

           done

           for systemdos in "${system_to_shutdown[@]}"; do

               echo "Shutting Down System $systemdos"
               ssh $systemdos 'sudo shutdown -h now'
               sleep 5

           done

           break

        elif [ "$user_input" == "no" ]; then

             echo
             echo
             echo " ----> Anti-DOS Mode Enabled. Will try to handle the Flood <----"
             echo
             echo "Will block all traffic from/to identified malicious IPs and implement Rate Limiting for the corresponding protocols,"
             echo "to smooth out the bandwidth usage and alleviate the CPU(s). These configurations will help prevent further strain,"
             echo "allowing the Victim System(s) to maintain stability and prevent further resource exhaustion until the attack subsides."
             echo
             echo

             break

        else
                echo
```

Εικόνα 64. Doshandler.sh [1/3]

```
GNU nano 6.2
        else
                echo
                echo "Invalid input. Please type 'yes' or 'no'."
                echo

        fi

     done

}

echo
echo
echo "*** Denial Of Service (DOS) Attempt ***"
echo

sleep 8
sudo kill -15 $SNORT_PID
sleep 1
counter=""

##############################################################
#Further Classification and Source & Target Identification
##############################################################

if echo "$new_lines" | grep -q "TCP"; then

    counter="TCP"
    echo "--- Type: TCP SYN Flood Attack ---"


elif echo "$new_lines" | grep -q "UDP"; then

      counter="UDP"
      echo "--- Type: UDP Request Flood Attack ---"


elif echo "$new_lines" | grep -q "ICMP"; then

      counter="ICMP"
      echo "--- Type: ICMP Echo - Request Flood Attack ---"


else
      echo "Detection Error! Exiting.."
      exit 1

fi


process_flag "length" "14" "16" "2" "6"
sleep 1
dos_source_array=( "$counter" )
echo
echo
dos_function "${final_destination_sorted[@]}"
sleep 1
```

Εικόνα 65. Doshandler.sh [2/3]

```
GNU nano 6.2
fi


process_flag "length" "14" "16" "2" "6"
sleep 1
dos_source_array=( "$counter" )
echo
echo
dos_function "${final_destination_sorted[@]}"
sleep 1

#######################################################
#Evasion Countermeasure Filter and Shield Deployment
#######################################################

echo
echo " >>>  Will try to find the original Attacker's IP(s) <<< "
echo
echo
echo "Checking source IP(s) against pre-Configured Network Variables"
echo "and running a Network Scan to sort out the Spoofed IP(s)"
echo

ip_unspoof "${final_source_sorted[@]}"
sleep 1

network_scan "1" "$interface" "${source_macs_sorted[@]}" > /dev/null 2>&1
sleep 1

if [[ ${#not_tracable[@]} -ne 0 ]]; then

   unified_dos_source=( "${scanned_ip_array[@]}" "${not_spoofed_array[@]}" )
   dos_sorted=($(for sorted10 in "${unified_dos_source[@]}"; do echo "${sorted10}"; done |  sort -u))

   if [ "${#dos_sorted[@]}" -ne 0 ]; then

      echo "Unable to pinpoint every spoofed IP with certainty!! All identified source IPs"
      echo "will be utilized moving forward (Excluding pre-Configured spoofed values)"
      echo
      dos_source_array=( "$counter" "${dos_sorted[@]}" )
      destination_ipcheck "$length6" "ddos.sh" "${final_destination_sorted[@]}" "${dos_source_array[@]}"

   else

      echo "Unable to pinpoint any Source IPs!! Only Rate Limiting"
      echo "for the $counter Protocol will be utilized moving forward"
      echo
      destination_ipcheck "$length6" "ddos.sh" "${final_destination_sorted[@]}" "${dos_source_array[@]}"

   fi


else

   echo "Finished Analyzing. Will utilize Malicious IP(s): ${scanned_ip_array[@]}"
   echo
   dos_source_array=( "$counter" "${scanned_ip_array[@]}" )
   destination_ipcheck "$length6" "ddos.sh" "${final_destination_sorted[@]}" "${dos_source_array[@]}"

fi
```

Εικόνα 66. Doshandler.sh [3/3]

Στην περίπτωση διαχείρισης μιας επίθεσης Denial of Service, το σύστημα συγκρίνει αρχικά τις λέξεις – κλειδιά “TCP”, “UDP” και “ICMP” με τα περιεχόμενα της μεταβλητής new_lines, και κατατάσσει τον τύπο πλημμύρας που δέχονται τα θύματα, σύμφωνα με τα παραπάνω πρωτόκολλα. Αφού η process_flag() επιστρέψει τις κατάλληλες διευθύνσεις IP αποστολέα και παραλήπτη, οι τελευταίες χρησιμοποιούνται ως όρισμα στη συνάρτηση Dos_function(), στα πλαίσια της λειτουργίας της μονάδας Attack Specific Actions. Βασικός της σκοπός είναι η απενεργοποίηση των παραβιασμένων συσκευών, σε περίπτωση που το επιλέξει ο διαχειριστής. Συγκεκριμένα, το σύστημα συνδέεται στα θύματα χρησιμοποιώντας SSH, και άμεσα εκτελεί εντολή αναστολής λειτουργίας (sudo shutdown -h now) με σκοπό την αναχαίτηση της επίθεσης σε πρώιμο στάδιο. Αντιθέτως εάν το σύστημα πρέπει να παραμείνει ενεργοποιημένο, και εφόσον επιλεχθεί από το διαχειριστή, ενεργοποιείται η λειτουργία “Anti – Dos” κατά την οποία θα πραγματοποιηθεί προσπάθεια διαχείρισης της χειμαρρώδους κυκλοφορίας, διατηρώντας μια σχετική σταθερότητα στο φόρτο των υπολογιστικών πόρων της συσκευής, αλλά και του διαθέσιμου εύρους ζώνης. Αφού πραγματοποιηθεί

παρόμοιος με τον προαναφερθέντα μηχανισμός ανίχνευσης των Spoofed IP και παράλληλης αναζήτησης των πραγματικών διευθύνσεων επιτιθέμενου, οι τιμές που ενδεχομένως προκύψουν αποθηκεύονται στην dos_source_array. Ανεξαρτήτως των σεναρίων εκτέλεση της destination_ipcheck(), σε όλες τις περιπτώσεις αποθηκεύεται στο παραπάνω Array η τιμή ενός Counter, η οποία αναπαριστά τα τρία σενάρια πλημμυρών που μελετώνται (TCP, UDP, ICMP), και χρησιμοποιείται αργότερα από την Ddos.sh Shield

Ddos.sh

```
  GNU nano 6.2
#!/bin/bash

#Shield For DoS Attacks

counter="$1"
shift
unified_array_mal=("$@")

dos_source_check=0

##########################################################################################
#Block the Malicious IP Addresses and Implement Rate Limiting for Specified Protocol
##########################################################################################

sudo nft add table ip filter
sudo nft add chain ip filter input { type filter hook input priority 0 \; }
sudo nft add chain ip filter output { type filter hook output priority 0 \; }


if [ ${#unified_array_mal[@]} -ne 0 ]; then

   ((dos_source_check++))

   for ip_to_ban in "${unified_array_mal[@]}"; do


       sudo nft add rule ip filter input ip saddr "$ip_to_ban" drop
       sudo nft add rule ip filter output ip daddr "$ip_to_ban" drop


   done

fi


if [ "$counter" == "UDP" ]; then

     sudo nft add rule ip filter input ip protocol udp limit rate 100/second burst 50 packets drop
     sudo nft add rule ip filter output ip protocol udp limit rate 100/second burst 50 packets drop

elif [ "$counter" == "TCP" ]; then

     sudo nft add rule ip filter input ip protocol tcp limit rate 100/second drop
     sudo nft add rule ip filter output ip protocol tcp limit rate 100/second drop


elif [ "$counter" == "ICMP" ]; then

     sudo nft add rule ip filter input ip protocol icmp limit rate 100/second drop
     sudo nft add rule ip filter output ip protocol icmp limit rate 100/second drop

fi


if [[ "$dos_source_check" -ne 0 ]]; then

     echo
     echo
     echo "*** Attacker was Blocked and Rate Limiting for $counter Traffic was applied Successfully ***"

else
     echo
     echo
     echo "*** Rate Limiting was applied Successfully for $counter Traffic ***"
fi
```

Εικόνα 67. Ddos.sh Shield

Ο συγκεκριμένος μηχανισμός θωράκισης των θυμάτων σε μια επίθεση Denial of Service στοχεύει στην καλύτερη δυνατή διαχείριση της εισερχόμενης, αλλά και εξερχόμενης κυκλοφορίας, δεδομένης της έλλειψης ισχυρών υλικών πόρων (πχ. Could Server Providers). Ειδικότερα, για τις κακόβουλες διευθύνσεις που προκύπτουν από τον προηγούμενο Handler εφαρμόζεται απαγόρευση εισερχόμενης και εξερχόμενης κυκλοφορίας, ενώ για ενδεχομένως άγνωστες IPs επιτιθέμενων υλοποιείται μηχανισμός περιορισμού ρυθμού (Rate Limiting). Οι κανόνες απόρριψης αφορούν όλα τα είδη πακέτων, ενώ ο περιορισμός ρυθμού κυκλοφορίας εφαρμόζεται στα πρωτόκολλα της επίθεσης, και βασίζεται στην τιμή που θα αποδοθεί στη μεταβλητή Counter. Τα πακέτα που υπερβαίνουν την τιμή κατωφλίου που επιλέχθηκε (100 Packets/Sec) θα απορρίπτονται αυτόματα από το τείχος προστασίας. Τέλος το εργαλείο που χρησιμοποιήθηκε για τη σύσταση του τείχους προστασίας είναι το Nftables Framework, το οποίο αποτελεί εξέλιξη του παλαιότερου Iptables. Προσφέρει βελτιωμένη απόδοση και μειωμένη πολυπλοκότητα, τα οποία είναι ιδιαίτερα πολύτιμα χαρακτηριστικά σε μια επίθεση όπου το θύμα πασχίζει να διαθέσει πόρους σε οποιαδήποτε υπολογιστική διεργασία.

Dhcpspoofhandler.sh

Ο τελευταίος Handler των επιθέσεων που μελετήθηκαν αφορά τα σενάρια DHCP Spoofing. Αρχικά με την εκτέλεση της process_flag() καθορίζονται οι IP των διακομιστών DHCP που έστειλαν τα κακόβουλα πακέτα, αλλά και οι διευθύνσεις MAC των παραληπτών. Υπενθυμίζεται ότι ο κανόνας 10 του Snort Ruleset που πυροδοτεί τη συγκεκριμένη διαδικασία δεν καθορίζει συγκεκριμένο τύπο διευθύνσεων αποστολέα και παραλήπτη, καθώς ο πρώτος θα μπορούσε να πλαστογραφήσει την IP του, ενώ ο δεύτερος βρίσκεται ακόμα στη φάση ανάθεσης (IP 0.0.0.0). Συνεπώς, οι ύποπτες IP της process_flag() θα περάσουν από τη συνάρτηση ip_unspoof() κυρίως για τη σύγκρισή τους με τον εξουσιοδοτημένο DHCP Server του δικτύου. Παράλληλα οι MAC της πρώτης συνάρτησης θα αποτελέσουν όρισμα για την mac2ip_convertor(), με σκοπό να προκύψουν μόνο οι διευθύνσεις που αφορούν τους προστατευόμενους Hosts, βάσει της μοναδικής MAC τους. Εάν και τα δύο Arrays εξόδου ύστερα από την ολοκλήρωση των παραπάνω συναρτήσεων (not_spoofed_array και protected_macs_converted), είναι μηδενικά, τότε πρόκειται για False Positive ειδοποίηση, και το πρόγραμμα τερματίζεται πρόωρα. Διαφορετικά, η διαδικασία συνεχίζεται με την εκτέλεση της create_network_config() για κάθε επιβεβαιωμένο θύμα του δικτύου. Η συνάρτηση αυτή αναπτύχθηκε στα πλαίσια της μονάδας Attack Specific Actions, καθώς δημιουργεί ένα εξατομικευμένο αρχείο ρυθμίσεων Netplan για κάθε παραβιασμένο Host (01-network-manager-all.yaml.$systemdhcp). Το

Netplan είναι το προεπιλεγμένο πρόγραμμα διαμόρφωσης δικτύου στα σύγχρονα συστήματα Ubuntu, και στη παρούσα συνάρτηση θα χρησιμοποιηθεί ως αντικατάσταση για τις ενδεχομένως κακόβουλες παραμετροποιήσεις, που εφαρμόστηκαν από τον μη εξουσιοδοτημένο διακομιστή DHCP. Αναλυτικότερα, όσον αφορά τις διευθύνσεις του κυρίως δικτύου (interfacedhcp) το σύστημα αναθέτει στα θύματα την προσωπική τους Static IP και Subnet Mask (victimdhcp), η οποία είναι τύπου Emergency και εισάγεται από το διαχειριστή στο τμήμα διαμόρφωσης μεταβλητών δικτύου της Mainscript.sh (#Set up the Network Variables). Δίνονται επίσης οι τιμές διευθύνσεων του πιστοποιημένων Gateway (router) καθώς των DNS Server (dns1 – dns2). Τέλος, εφόσον το παραπάνω αρχείο Netplan θα αντικαταστήσει και τις υφιστάμενες ρυθμίσεις του δευτερεύοντος δικτύου (enp0s9), προστίθενται σε αυτό οι κατάλληλες IP προκειμένου ο ασφαλής δίαυλος επικοινωνίας μεταξύ των θυμάτων και του IDS/IPS να παραμείνει ενεργοποιημένος. Το αρχείο υλοποιείται αυτόματα σε γλώσσα Yaml, ονοματίζεται κατάλληλα, και στη συνέχεια προωθείται στις παραβιασμένες μηχανές για μελλοντική χρήση του από το ομώνυμο Shield. Για την ασφαλή αποστολή του χρησιμοποιείται το Scp, όπως συμβαίνει και για τη μεταφορά των Scripts από την destination_ipcheck(). Η τελευταία καλείται στο κλείσιμο του Handler, έχοντας ως όρισμα το Array των ύποπτων διακομιστών, καθώς και τις τιμές IP των Router και DNS1/DNS2.

Dhcpspoofing.sh

Στο συγκεκριμένο Shield, το σύστημα ελέγχει τις IP των Gateway και DNS Servers του θύματος, συγκρίνοντάς τις με τις διευθύνσεις που προέκυψαν από την dhcpspoofing_array του Handler. Για κάθε IP που δεν συμπίπτει με κάποια αντίστοιχη εξουσιοδοτημένου Host, ενημερώνει κατάλληλα το διαχειριστή και στη συνέχεια την προσθέτει σε διακριτά Arrays (mal_gws, maldns_servers). Στη συνέχεια, και αφού το έχει επιλέξει ο διαχειριστής, το σύστημα διαγράφει όλες τις υφιστάμενες ρυθμίσεις δικτύου της κύριας διεπαφής (ip addr flush dev $interfacedhcp), καθώς και το DHCP Lease που έχει συνάψει το θύμα με τον επιτιθέμενο Server (dhclient -r > /dev/null 2, rm /var/lib/dhcp/dhclient.leases). Τη θέση τους πλέον είναι έτοιμο να πάρει το yaml αρχείο που δημιουργήθηκε νωρίτερα, αλλά προτού τεθεί σε λειτουργία και τερματιστεί η σύνδεση με το εκάστοτε θύμα, ρυθμίζονται κατάλληλα οι κανόνες του εργαλείου Iptables. Οι διευθύνσεις αποστολέων των DHCP μηνυμάτων καθώς και αυτές των Gateway και DNS Servers που προέκυψαν από τον παραπάνω έλεγχο, συγκεντρώνονται, ταξινομούνται ως προς τη διαγραφή διπλότυπων IP και προστίθεται στους κανόνες απόρριψης εισερχόμενης, εξερχόμενης και προωθούμενης κυκλοφορίας.

```
  GNU nano 6.2
#!/bin/bash

#Handler For DHCP Spoofing Attacks

##########################################################################################
#Attack Specific Action - Deploy a Static IP Configuration File to the Victim
##########################################################################################

create_network_config() {


    interfacedhcp=$1
    victimdhcp=$2
    dns1=$3
    dns2=$4
    router=$5
    systemdhcp=$6
    idssecuredhcp=$7


    cat <<EOL > "01-network-manager-all.yaml.$systemdhcp"
network:
  version: 2
  renderer: networkd
  ethernets:
    $interfacedhcp:
      addresses:
        - $victimdhcp
      nameservers:
        addresses: [$dns1,$dns2]
      routes:
        - to: default
          via: $router

    enp0s9:
      addresses:
        - $idssecuredhcp/24
EOL

    echo "Netplan Configuration file (01-network-manager-all.yaml.$systemdhcp) created for $systemdhcp"
    echo "to pass emergency""static IP address $victimdhcp. Sending it now!"
    echo
    scp 01-network-manager-all.yaml.$systemdhcp $systemdhcp:/home/manos/Desktop > /dev/null
    echo ">> Done. Will use this File later if needed. <<"
    echo
    echo

}

#################################################################
#Source & Target Identification - Further Attack Verification
#################################################################

echo
echo
echo "*** Detected DHCP Offer Packet From Untrusted Source ***"
echo
echo "--- Potential Rogue DHCP Server ---"

sleep 3
sudo kill -15 $SNORT_PID
sleep 1
```

Εικόνα 68. Dhcpspoofhandler.sh [1/2]

```
  GNU nano 6.2                                                                    dhcpspoofha
echo
echo "*** Detected DHCP Offer Packet From Untrusted Source ***"
echo
echo "--- Potential Rogue DHCP Server ---"

sleep 3
sudo kill -15 $SNORT_PID
sleep 1

process_flag "length" "14" "16" "2" "6" > /dev/null 2>&1
sleep 1
macZip_convertor "${home_macs_sorted[@]}" > /dev/null 2>&1
sleep 1
ip_unspoof "${final_source_sorted[@]}" > /dev/null 2>&1
sleep 1

if [ ${#not_spoofed_array[@]} -eq 0 ] && [ ${#protected_macs_converted[@]} -eq 0 ]; then

   echo "False Positive Alert! Exiting.."
   exit 1

fi


length_dhcp=${#protected_macs_converted[@]}

echo
echo
echo "Anilyzing Snort Logs to Extract Crucial Information about the Attack"
echo
echo
echo "Logs Check Came back with the Following:"
echo
echo "Source IP(s): ${not_spoofed_array[@]}"
echo
echo "Destination -- Protected Network -- IP(s): ${protected_macs_converted[@]}"
echo
echo


for macdhcp in "${protected_macs_converted[@]}"; do

    if [ "$macdhcp" == "$victim1_ip" ]; then

        create_network_config "$interface" "$emergency_victim1" "$dns1" "$dns2" "$default_gateway_ip" "victim1" "$victim1_enp0s9_ip"

    else
        echo "DETECTION ERROR!"

    fi

done

#####################
#Shield Deployment
#####################

dhcpspoofing_array=( "$interface" "$default_gateway_ip" "$dns1" "$dns2" "${not_spoofed_array[@]}" )
destination_ipcheck "$length_dhcp" "dhcpspoofing.sh" "${protected_macs_converted[@]}" "${dhcpspoofing_array[@]}"
```

Εικόνα 69. Dhcpspoofhandler.sh [2/2]

```
  GNU nano 6.2
interfacedhcp=$1
shift
gateway_dhcp=$1
shift
dns1_dhcp=$1
shift
dns2_dhcp=$1
shift
dhcp_malservers=("$@")

##########################################
#Check The Default GWs and DNS Servers
##########################################

mal_gws=()
maldns_servers=()

echo
echo "--- Checking the Gateway and DNS Servers IPs ---"
echo

while IFS= read -r gatewayline; do

  defaultgw=$(echo "$gatewayline" | awk '{print $3}')

  defaultgw_array+=("$defaultgw")

done < <(ip route | grep '^default')

for gw in "${defaultgw_array[@]}"; do
  if [[ "$gw" != "$gateway_dhcp" ]]; then
    echo "!! Untrusted Gateway Found: $gw"
    echo
    mal_gws+=("$gw")
  fi
done
if [ "${#mal_gws[@]}" -eq 0 ]; then

   echo
   echo "<< Check came back with No Untrusted Gateways >>"
   echo
fi

while IFS= read -r linedns; do
  if [[ "$linedns" != "$dns1_dhcp" && "$linedns" != "$dns2_dhcp" ]]; then
  echo "!! Untrusted DNS Server Found: $linedns"
  echo
  maldns_servers+=("$linedns")
  fi
done < <(sudo resolvectl status $interfacedhcp | grep "DNS Servers" | awk '{for(i=3;i<=NF;i++) print $i}')

if [ "${#maldns_servers[@]}" -eq 0 ]; then

   echo
   echo "<< Check came back with No Untrusted DNS Servers >>"
   echo
fi
```

Εικόνα 70. Dhcpspoofing.sh Shield [1/2]

```
  GNU nano 6.2
##########################################
#Apply the Static IP Configuration File
##########################################

echo
while true; do
    read -p "Do you want to apply the Static IP Configuration file that was created earlier for this System? (yes/no): " user_input
    if [ "$user_input" = "yes" ]; then

        echo
        echo "Flushing all Malicious Configurations on the Interface First.."
        sudo ip addr flush dev $interfacedhcp &&
        sudo dhclient -r > /dev/null 2>&1 &&
        sudo rm /var/lib/dhcp/dhclient.leases &&
        echo
        echo "Done! Static IP is redy to be Applied"
        echo
        echo
        break

    elif [ "$user_input" = "no" ]; then

         echo
         echo "No Action Taken! Exiting.."
         break
         exit 1

    else

        echo
        echo "Invalid input. Please type 'yes' or 'no'."
        echo

    fi
done

filename=$(ls | grep "01-network-manager-all.yaml") &&
filename_cut=$(echo "$filename" | cut -d '.' -f 1,2) &&
sudo mv /home/manos/Desktop/$filename /home/manos/Desktop/$filename_cut
cd &&
sudo mv /home/manos/Desktop/$filename_cut /etc/netplan

##########################################
#Block all the Malicious IP Addresses
##########################################

echo "Blocking all Malicious Identified IPs.."

mal_ipsdhcp=( "${mal_gws[@]}" "${maldns_servers[@]}" "${dhcp_malservers[@]}" )
mal_ips_sorted=($(for dhcpspoofip in "${mal_ipsdhcp[@]}"; do echo "${dhcpspoofip}"; done | sort -u))

for mal in "${mal_ips_sorted[@]}"; do

sudo iptables -A INPUT -s $mal -j DROP &&
sudo iptables -A OUTPUT -d $mal -j DROP &&
sudo iptables -A FORWARD -d $mal -j DROP &&
sudo iptables -A FORWARD -s $mal -j DROP

done

echo
echo "Blocked!"
echo
echo
echo "Applying new Neplan Network Configurations and Exiting"
echo
echo "Make sure to enable DHCP Services (If Needed) when the attack is completely Mitigated!"
sudo netplan apply > /dev/null 2>&1
```

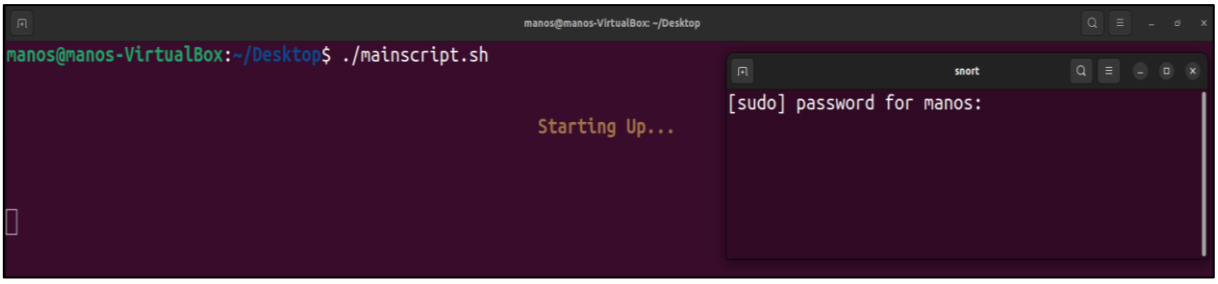
Εικόνα 71. Dhcpspoofing.sh Shield [2/2]

3

ΛΕΙΤΟΥΡΓΙΑ ΚΑΙ ΑΠΟΤΕΛΕΣΜΑΤΑ

Σε αυτό το Κεφάλαιο θα αναλυθεί η λειτουργία του συστήματος ασφάλειας (IDS/IPS) που σχεδιάστηκε και υλοποιήθηκε στα προηγούμενα. Για την προσομοίωση κάθε τύπου επίθεσης στο δίκτυο της Εικόνας 36 (Lab Setup), εξηγούνται οι απαιτούμενες συνθήκες υπό τις οποίες το Kali Linux “Attacker” VM (10.0.2.7) θα στέλνει την κακόβουλη κυκλοφορία προς τους υπόλοιπους Hosts του VLAN. Παράλληλα θα αξιολογηθούν τα αποτελέσματα των ενεργειών του IDS/IPS βάσει της αναφοράς που προκύπτει στην κονσόλα του, ύστερα από κάθε σενάριο επίθεσης. Στο συγκεκριμένο δίκτυο οι συσκευές που θεωρητικά ανήκουν στο προστατευόμενο σύνολο είναι οι Virtual Router (10.0.2.1), Victim1 (10.0.2.5) VM, καθώς και Virtual DHCP Server (10.0.2.). Ωστόσο, όπως έχει αποσαφηνισθεί στα προηγούμενα Κεφάλαια, η ενεργή δράση του IDS/IPS (10.0.2.4) VM στο σκέλος της αντιμετώπισης επιθέσεων περιορίζεται μόνο σε Linux OS όπως του Victim1 VM (Ubuntu). Σημειώνεται ότι το σύστημα δίνει τη δυνατότητα κλιμάκωσης όσον αφορά τη μελλοντική προσθήκη περισσότερων Victim, αλλά και Attacker μηχανών.

Το σύστημα ασφάλειας τίθεται σε λειτουργία, εφόσον ενεργοποιηθεί η συνάρτηση Mainscript.sh του IDS/IPS (./mainscript.sh). Στην αρχή της εκτέλεσής της δημιουργείται νέο παράθυρο για την παράλληλη αρχικοποίηση του Snort, και προτρέπει το διαχειριστή να εισάγει τον κωδικό πρόσβασης της μηχανής. Ύστερα, το λογισμικό θα βρίσκεται πλέον σε φάση ανίχνευσης επιθέσεων, και παράλληλα σε ετοιμότητα να αντιδράσει κατασταλτικά σε πραγματικό χρόνο.



Εικόνα 72. Αρχικοποίηση Συστήματος IDS/IPS

3.1 Nmap Scans

Συνθήκες Δικτύου

Για την προσομοίωση του σεναρίου επιθέσεων Nmap Scan(s), επιλέχθηκε το Victim1 ως στόχος των σαρώσεων. Διατηρεί ανοιχτά και μη – φιλτραρισμένα τα Ports 22 (SSH) και 80 (HTTP), για συνδέσεις TCP με οποιονδήποτε Host. Παράλληλα, στο Attacker VM, εγκαταστάθηκε το εργαλείο Nmap και στη συνέχεια πραγματοποιήθηκαν οι σαρώσεις που συνοψίζονται στην παρακάτω Εικόνα.

Scan Type	Terminal Command	Output
XMAS	sudo nmap -sX 10.0.2.5	22/tcp open | filtered ssh 88/tcp open | filtered http
FIN	sudo nmap -sF 10.0.2.5	22/tcp open | filtered ssh 88/tcp open | filtered http
NULL	sudo nmap -sN 10.0.2.5	22/tcp open | filtered ssh 88/tcp open | filtered http
STEALTH	sudo nmap -sS 10.0.2.5	22/tcp open ssh 88/tcp open http
FULL	sudo nmap -sT 10.0.2.5	22/tcp open ssh 88/tcp open http
DECOY	sudo nmap {scan type} -D {Decoy IP1} {Decoy IP2} 10.0.2.5	Same as the Selected Scan Type

Εικόνα 73. Σύνοψη σαρώσεων Nmap

Οι διαφορές στα Outputs μεταξύ των Stealth / Full και Xmas / Fin / Null Scan προκύπτουν από τη φύση των ίδιων των σαρώσεων. Όταν το Victim1 απαντήσει στα TCP SYN πακέτα με αντίστοιχα SYN-ACK για τις θύρες 22 και 80, τότε το Nmap είναι σε θέση να επιβεβαιώσει ότι είναι ανοιχτές και Unfiltered. Στις περιπτώσεις όμως όπου ο Attacker δεν λάβει κανένα πακέτο για τα αντίστοιχα Ports, το Nmap δεν δύναται

να ανιχνεύσει την ύπαρξη κάποιου φίλτρου. Ενδεικτικά, στην παρακάτω Εικόνες παρατίθενται στιγμιότυπα από τις καταγραφές πακέτων μέσω του εργαλείου ανάλυσης Wireshark, τη στιγμή που ο επιτιθέμενος έχει μόλις πραγματοποιήσει σαρώσεις τύπου Xmas και Full αντίστοιχα στο Victim1.

	Time	Source	Destination	Protocol	Length	Info
7	0.104782707	10.0.2.7	10.0.2.5	TCP	54	46309 → 5900 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
8	0.104816228	10.0.2.7	10.0.2.5	TCP	54	46309 → 113 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
9	0.104824193	10.0.2.7	10.0.2.5	TCP	54	46309 → 80 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
10	0.104830786	10.0.2.7	10.0.2.5	TCP	54	46309 → 995 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
11	0.104837553	10.0.2.7	10.0.2.5	TCP	54	46309 → 1025 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
12	0.104844368	10.0.2.7	10.0.2.5	TCP	54	46309 → 8888 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
13	0.104851553	10.0.2.7	10.0.2.5	TCP	54	46309 → 3306 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
14	0.104860243	10.0.2.7	10.0.2.5	TCP	54	46309 → 993 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
15	0.104867625	10.0.2.7	10.0.2.5	TCP	54	46309 → 21 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
16	0.104892659	10.0.2.7	10.0.2.5	TCP	54	46309 → 443 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
17	0.105391876	10.0.2.5	10.0.2.7	TCP	60	5900 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
18	0.105392094	10.0.2.5	10.0.2.7	TCP	60	113 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
19	0.105392119	10.0.2.5	10.0.2.7	TCP	60	995 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
20	0.105392143	10.0.2.5	10.0.2.7	TCP	60	1025 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
21	0.105392166	10.0.2.5	10.0.2.7	TCP	60	8888 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
22	0.105392189	10.0.2.5	10.0.2.7	TCP	60	3306 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
23	0.105392210	10.0.2.5	10.0.2.7	TCP	60	993 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
24	0.105392231	10.0.2.5	10.0.2.7	TCP	60	21 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
25	0.105420342	10.0.2.5	10.0.2.7	TCP	60	443 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
26	0.107608805	10.0.2.7	10.0.2.5	TCP	54	46309 → 139 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
27	0.107635917	10.0.2.7	10.0.2.5	TCP	54	46309 → 22 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
28	0.107644107	10.0.2.7	10.0.2.5	TCP	54	46309 → 445 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
29	0.107651172	10.0.2.7	10.0.2.5	TCP	54	46309 → 1720 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
30	0.107658670	10.0.2.7	10.0.2.5	TCP	54	46309 → 8080 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
31	0.107665950	10.0.2.7	10.0.2.5	TCP	54	46309 → 110 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
32	0.107675397	10.0.2.7	10.0.2.5	TCP	54	46309 → 23 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
33	0.107683411	10.0.2.7	10.0.2.5	TCP	54	46309 → 256 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
34	0.107708695	10.0.2.7	10.0.2.5	TCP	54	46309 → 587 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
35	0.107716899	10.0.2.7	10.0.2.5	TCP	54	46309 → 1723 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
36	0.107727006	10.0.2.7	10.0.2.5	TCP	54	46309 → 135 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
37	0.107736821	10.0.2.7	10.0.2.5	TCP	54	46309 → 53 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
38	0.107747844	10.0.2.7	10.0.2.5	TCP	54	46309 → 554 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
39	0.107760313	10.0.2.7	10.0.2.5	TCP	54	46309 → 199 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
40	0.107774110	10.0.2.7	10.0.2.5	TCP	54	46309 → 111 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
41	0.107786174	10.0.2.7	10.0.2.5	TCP	54	46309 → 25 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
42	0.107796340	10.0.2.7	10.0.2.5	TCP	54	46309 → 143 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
43	0.107809635	10.0.2.7	10.0.2.5	TCP	54	46309 → 3389 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
44	0.108040877	10.0.2.5	10.0.2.7	TCP	60	139 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
45	0.108041128	10.0.2.5	10.0.2.7	TCP	60	445 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
46	0.108041173	10.0.2.5	10.0.2.7	TCP	60	1720 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
47	0.108041215	10.0.2.5	10.0.2.7	TCP	60	8080 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
48	0.108041256	10.0.2.5	10.0.2.7	TCP	60	110 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
49	0.108041294	10.0.2.5	10.0.2.7	TCP	60	23 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
50	0.108041333	10.0.2.5	10.0.2.7	TCP	60	256 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
51	0.108041378	10.0.2.5	10.0.2.7	TCP	60	587 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
52	0.108103757	10.0.2.5	10.0.2.7	TCP	60	1723 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
53	0.108103897	10.0.2.5	10.0.2.7	TCP	60	135 → 46309 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0

ame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0, id 0 | 0000 ff ff ff ff ff ff 08 00 27 09 51 06 0

wireshark_eth09EMNX2.pcapng | Packets: 2010

Εικόνα 74. Καταγραφή Wireshark για Xmas Scan

27	0.067274633	10.0.2.7	10.0.2.5	TCP	74	35088 → 143 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
28	0.067292114	10.0.2.7	10.0.2.5	TCP	74	41190 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
29	0.067308285	10.0.2.7	10.0.2.5	TCP	74	43274 → 80 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
30	0.067328629	10.0.2.7	10.0.2.5	TCP	74	48578 → 995 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
31	0.067361368	10.0.2.7	10.0.2.5	TCP	74	48422 → 1720 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
32	0.067380776	10.0.2.7	10.0.2.5	TCP	74	50436 → 443 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
33	0.067397257	10.0.2.7	10.0.2.5	TCP	74	54620 → 25 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
34	0.067421868	10.0.2.7	10.0.2.5	TCP	74	43102 → 256 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
35	0.067429076	10.0.2.5	10.0.2.7	TCP	60	113 → 39314 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
36	0.067473579	10.0.2.5	10.0.2.7	TCP	60	199 → 55332 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
37	0.067473627	10.0.2.5	10.0.2.7	TCP	60	143 → 35088 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
38	0.067473671	10.0.2.5	10.0.2.7	TCP	74	22 → 41190 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3221184063 TSecr=663518541 WS=128
39	0.067473711	10.0.2.5	10.0.2.7	TCP	74	80 → 43274 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3221184063 TSecr=663518541 WS=128
40	0.067482031	10.0.2.7	10.0.2.5	TCP	74	41626 → 8080 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
41	0.067493979	10.0.2.7	10.0.2.5	TCP	66	41190 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=663518541 TSecr=3221184063
42	0.067503582	10.0.2.7	10.0.2.5	TCP	74	41020 → 445 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518541 TSecr=0 WS=128
43	0.067509203	10.0.2.7	10.0.2.5	TCP	66	43274 → 80 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=663518541 TSecr=3221184063
44	0.067555133	10.0.2.5	10.0.2.7	TCP	60	995 → 48578 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
45	0.067555223	10.0.2.5	10.0.2.7	TCP	60	1720 → 48422 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
46	0.067555279	10.0.2.5	10.0.2.7	TCP	60	443 → 50436 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
47	0.067555326	10.0.2.5	10.0.2.7	TCP	60	25 → 54620 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
48	0.067565917	10.0.2.7	10.0.2.5	TCP	66	41190 → 22 [RST, ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=663518541 TSecr=3221184063
49	0.067612034	10.0.2.7	10.0.2.5	TCP	66	43274 → 80 [RST, ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=663518541 TSecr=3221184063
50	0.067613181	10.0.2.5	10.0.2.7	TCP	60	256 → 43102 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
51	0.067658271	10.0.2.7	10.0.2.5	TCP	74	58148 → 1025 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
52	0.067703721	10.0.2.5	10.0.2.7	TCP	60	8080 → 41626 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
53	0.067703835	10.0.2.5	10.0.2.7	TCP	60	445 → 41020 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
54	0.067741025	10.0.2.7	10.0.2.5	TCP	74	36192 → 110 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
55	0.067773903	10.0.2.7	10.0.2.5	TCP	74	53990 → 53 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
56	0.067795076	10.0.2.7	10.0.2.5	TCP	74	59908 → 8888 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
57	0.067811209	10.0.2.7	10.0.2.5	TCP	74	55176 → 139 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
58	0.067829426	10.0.2.7	10.0.2.5	TCP	74	47592 → 3389 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
59	0.067846589	10.0.2.7	10.0.2.5	TCP	74	34590 → 9103 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
60	0.067873544	10.0.2.7	10.0.2.5	TCP	74	42684 → 4045 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
61	0.067901605	10.0.2.7	10.0.2.5	TCP	74	33670 → 7625 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
62	0.067920894	10.0.2.5	10.0.2.7	TCP	60	1025 → 58148 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
63	0.067921009	10.0.2.5	10.0.2.7	TCP	60	110 → 36192 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
64	0.067923469	10.0.2.7	10.0.2.5	TCP	74	35132 → 10621 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
65	0.067941062	10.0.2.7	10.0.2.5	TCP	74	41088 → 32782 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
66	0.067973948	10.0.2.7	10.0.2.5	TCP	74	58116 → 2042 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
67	0.068001812	10.0.2.7	10.0.2.5	TCP	74	38536 → 2605 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=663518542 TSecr=0 WS=128
68	0.068023977	10.0.2.5	10.0.2.7	TCP	60	53 → 53990 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

rame 24: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0 | 0000 08 00 27 09 51 06 08 00 27 c0 f8 21 08 00 45 00 ··'·Q··· '··!··E·

wireshark_eth0D34UX2.pcapng | Packets: 2008 · Displayed: 2008 (100.0%) · Dropped: 0 (0.0%)

Εικόνα 75. Καταγραφή Wireshark για Full Scan

Για τη σάρωση τύπου Xmas παρατηρείται με κίτρινο χρώμα το πακέτο που στάλθηκε στη θύρα 80, για την οποία το Victim1 (όπως και την 22 που δεν φαίνεται στην εικόνα) δεν απαντάει με κάποιο TCP RST-ACK. Ανάλογη λογική ακολουθείται και για τις Fin και Null, με μοναδική διαφοροποίηση τα Flags που τίθενται από το σαρωτή. Όσον αφορά το Full Scan της επόμενης εικόνας, με κίτρινο χρώμα διακρίνεται η πλήρης TCP χειραψία μεταξύ Attacker και Victim1 για τη θύρα 80, ενώ για την 22 παρατηρείται το αντίστοιχο ACK του επιτιθέμενου. Τα πακέτα υπ' αριθμό 48 και 49 αναδεικνύουν τον απότομο τερματισμό των εν λόγω συνδέσεων από το μέρος του επιτιθέμενου. Διαφοροποιείται από το Stealth Scan στο τελευταίο μέρος της χειραψίας, όπου ο σαρωτής αντί να ολοκληρώσει το Three Way Handshake στέλνει ανάλογα μηνύματα τύπου RST, με σκοπό την άμεση διακοπή της σύνδεσης. Ο μετρητής καταγραφών του Wireshark σε κάθε περίπτωση μετράει περίπου δύο χιλιάδες πακέτα, τα οποία σύμφωνα με τους κανόνες 1-4 του Snort Ruleset συνιστούν συνθήκες επίθεσης

Λειτουργία του IDS/IPS και Αποτελέσματα

Στην Εικόνα 75 παρατίθεται η αναφορά που προκύπτει ύστερα από μια απλή Stealth σάρωση. Το σύστημα IDS/IPS ανίχνευσε την κακόβουλη κυκλοφορία και την κατέταξε ως Nmap Scan. Στη συνέχεια, αφού καθόρισε τον τύπο σάρωσης (Stealth/Full), προχώρησε στον εντοπισμό της κακόβουλης διεύθυνσης (Attacker - 10.0.2.7) αλλά και της παραβιασμένης μηχανής (Victim1 - 10.0.2.5). Ο έλεγχος στην Evasion Countermeasure μονάδα απέκλεισε το ενδεχόμενο ύπαρξης Decoy IP διευθύνσεων με αποτέλεσμα το σύστημα να συνεχίσει με την εκτέλεση του Shield στο Victim1. Εκεί, ανιχνεύθηκαν τα προαναφερθέντα Unfiltered Ports και με την επιλογή (yes) του διαχειριστή στις προτάσεις του συστήματος, εφαρμόστηκαν τα ανάλογα φίλτρα. Η επιτυχημένη παρέμβαση του IDS/IPS αποδεικνύεται από τις καταγραφές των κανόνων στο Iptables του Victim, όπως παρουσιάζεται στην Εικόνα 76, αλλά και από το Output του Nmap εάν επιχειρηθεί δεύτερο Scan ("All 1000 scanned ports on 10.0.2.5 are in ignored states"). Επιπροσθέτως, στην Εικόνα 77 παρουσιάζεται η αναφορά του IDS/IPS ως αποτέλεσμα μιας σάρωσης τύπου Null. Ο Attacker για το συγκεκριμένο σενάριο έχει χρησιμοποιήσει δύο πρόσθετες Decoy IP σάρωσης (sudo nmap -sN -D 10.0.2.55 10.0.2.1 10.0.2.5), ενώ παράλληλα έχει αποκλείσει την εισερχόμενη κυκλοφορία ARP πακέτων από τη MAC του IDS/IPS VM (sudo arptables -A INPUT --src-mac 08:00:27:98:39:56 -j DROP). Όπως φαίνεται λοιπόν και από την εν λόγω αναφορά, το σύστημα εντόπισε τις κακόβουλες IP, απέκλεισε την 10.0.2.1 ως επιβεβαιωμένη Spoofed (διεύθυνση του Router) και επιχείρησε να εντοπίσει την αληθινή εκ των 10.0.2.55 και 10.0.2.7, βάσει της διαθέσιμης MAC (08:00:27:98:51:06). Λόγω του αποκλεισμού

που προηγήθηκε από τον επιτιθέμενο, η σάρωση δεν επιστρέφει κάποια έξοδο, με αποτέλεσμα να χρησιμοποιηθούν όλες οι IP αποστολέα (εκτός φυσικά της 10.0.2.1). Τα αποτελέσματα της εφαρμογής του φίλτρου μόνο για το Port 80 (εφόσον επιλέχθηκε κατάλληλα) φαίνονται στους παραπάνω κανόνες Iptables.

```
manos@manos-VirtualBox:~/Desktop$ ./mainscript.sh

                                                    Starting Up...

                        -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 02/12 03:27:53
Recon activity -- Potential Nmap Scan Detected!
Type: Nmap Stealth/Full Scan. Check the Log File for further Information.

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:
Source IP(s): 10.0.2.7
Destination -- Protected Network -- IP(s): 10.0.2.5

-- Protected-Netowrk Machine(s) Under Attack: victim1 --


                                                    For victim1...

                        Security Script (Shield) Installed Successfully to the Target System

Banning Intruders IP(s)
Banned Successfully

Searching for some Default Open Ports on the Machine...
-> Found Port 22 (Service SSH) Open <-
Do you want to filter this port for trusted connections only? (yes/no): yes
Applying filter on port 22...
Done. Make sure to Drop the Filter once the Threat is Mitigated

-> Found Port 80 (Service HTTP) Open <-
Do you want to filter this port for trusted connections only? (yes/no): yes
Applying filter on port 80...
Done. Make sure to Drop the Filter once the Threat is Mitigated

Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 76. Generated Report για Stealth Scan

```
manos@manosvictim1-VirtualBox:~/Desktop$ sudo iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  10.0.2.7             0.0.0.0/0
ACCEPT     tcp  --  10.0.2.5             0.0.0.0/0            tcp dpt:22
ACCEPT     tcp  --  10.0.2.4             0.0.0.0/0            tcp dpt:22
ACCEPT     tcp  --  10.2.2.7             0.0.0.0/0            tcp dpt:22
ACCEPT     tcp  --  10.2.2.8             0.0.0.0/0            tcp dpt:22
ACCEPT     tcp  --  10.0.2.1             0.0.0.0/0            tcp dpt:22
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp dpt:22
ACCEPT     tcp  --  10.0.2.5             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.0.2.4             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.2.2.7             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.2.2.8             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.0.2.1             0.0.0.0/0            tcp dpt:80
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp dpt:80

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  0.0.0.0/0            10.0.2.7
DROP       all  --  10.0.2.7             0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  0.0.0.0/0            10.0.2.7
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.5             tcp spt:22
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.4             tcp spt:22
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.7             tcp spt:22
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.8             tcp spt:22
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.1             tcp spt:22
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp spt:22
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.5             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.4             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.7             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.8             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.1             tcp spt:80
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp spt:80
```

Εικόνα 77. Iptables Ruleset του Victim1 [1/2]

```
manos@manos-VirtualBox:~/Desktop$ ./mainscript.sh

                                                    Starting Up...

                        -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 02/12 04:06:54

Recon activity -- Potential Nmap Scan Detected!

Type: Nmap Null Scan. Check the Log File for further Information.

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:

Source IP(s): 10.0.2.55 10.0.2.7

Destination -- Protected Network -- IP(s): 10.0.2.5

Snort logs analysis also shows fewer source MACs compared to source IPs.
This indicates potential spoofed decoy IP addresses in the attack.
Will try to find the original Attacker's IP(s)

Running a Scan to Map the Attacker's MAC(s) back to the Corresponding IP(s)

-> Scan Came Back with Nothing! <-

-> Was not Able to Trace MAC(s): 08:00:27:09:51:06 <-

System was unable to trace all malicious MACs back to the corresponding IPs. All identified
source IPs will be utilized moving forward (Excluding known pre-Configured spoofed values)

-- Protected-Netowrk Machine(s) Under Attack: victim1 --


                                                    For victim1...

                        Security Script (Shield) Installed Successfully to the Target System

Banning Intruders IP(s)

Banned Successfully

Searching for some Default Open Ports on the Machine...

-> Found Port 22 (Service SSH) Open <-

Do you want to filter this port for trusted connections only? (yes/no): no

No action taken. Port stays unfiltered for potentially malicious Connections...

-> Found Port 80 (Service HTTP) Open <-

Do you want to filter this port for trusted connections only? (yes/no): yes

Applying filter on port 80...

Done. Make sure to Drop the Filter once the Threat is Mitigated

Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 78. Generated Report για Null Scan χρησιμοποιώντας Decoy IP Addresses

```
manos@manosvictim1-VirtualBox:~/Desktop$ sudo iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  10.0.2.55            0.0.0.0/0
DROP       all  --  10.0.2.7             0.0.0.0/0
ACCEPT     tcp  --  10.0.2.5             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.0.2.4             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.2.2.7             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.2.2.8             0.0.0.0/0            tcp dpt:80
ACCEPT     tcp  --  10.0.2.1             0.0.0.0/0            tcp dpt:80
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp dpt:80

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  0.0.0.0/0            10.0.2.55
DROP       all  --  10.0.2.55            0.0.0.0/0
DROP       all  --  0.0.0.0/0            10.0.2.7
DROP       all  --  10.0.2.7             0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  0.0.0.0/0            10.0.2.55
DROP       all  --  0.0.0.0/0            10.0.2.7
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.5             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.4             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.7             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.2.2.8             tcp spt:80
ACCEPT     tcp  --  0.0.0.0/0            10.0.2.1             tcp spt:80
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp spt:80
manos@manosvictim1-VirtualBox:~/Desktop$
```

Εικόνα 79. Iptables Ruleset του Victim1 [2/2]

3.2 SSH Password Authentication Bruteforce Attack

Συνθήκες Δικτύου

Στο δεύτερη κατηγορία επιθέσεων το Attacker VM επιχειρεί μια επίθεση Bruteforce στο Victim1, και συγκεκριμένα στην υπηρεσία SSH που διαθέτει μέσω του Port 22. Το λογισμικό ασφαλείας λειτουργεί για κάθε επιμέρους κατηγορία των παραπάνω, ωστόσο προτιμήθηκε η επίδειξη μιας Dictionary – Based επίθεσης. Είναι ο αποτελεσματικότερος τύπος Brute Force λόγω ευελιξίας και χαμηλού απαιτούμενου υπολογιστικού κόστους. Για την εύρεση λοιπόν του κωδικού που θα δώσει την πρόσβαση στην υπηρεσία SSH του Victim1, εγκαθίσταται στο Attacker VM το Hydra. Ένα δημοφιλές πρόγραμμα παραβίασης συνδέσεων σε δίκτυο που υποστηρίζει μεγάλο αριθμό πρωτοκόλλων για επιθέσεις ωμής βίας. Η εντολή που χρησιμοποιείται για την εκτέλεση της επίθεσης είναι η εξής.

sudo hydra -l manos -P /home/manos/Desktop/passwords.txt -s 22 -t 4 -f 10.0.2.5 ssh, όπου:

-l manos Καθορίζεται το όνομα χρήστη που θα χρησιμοποιεί το Hydra (Hostname του Victim1).

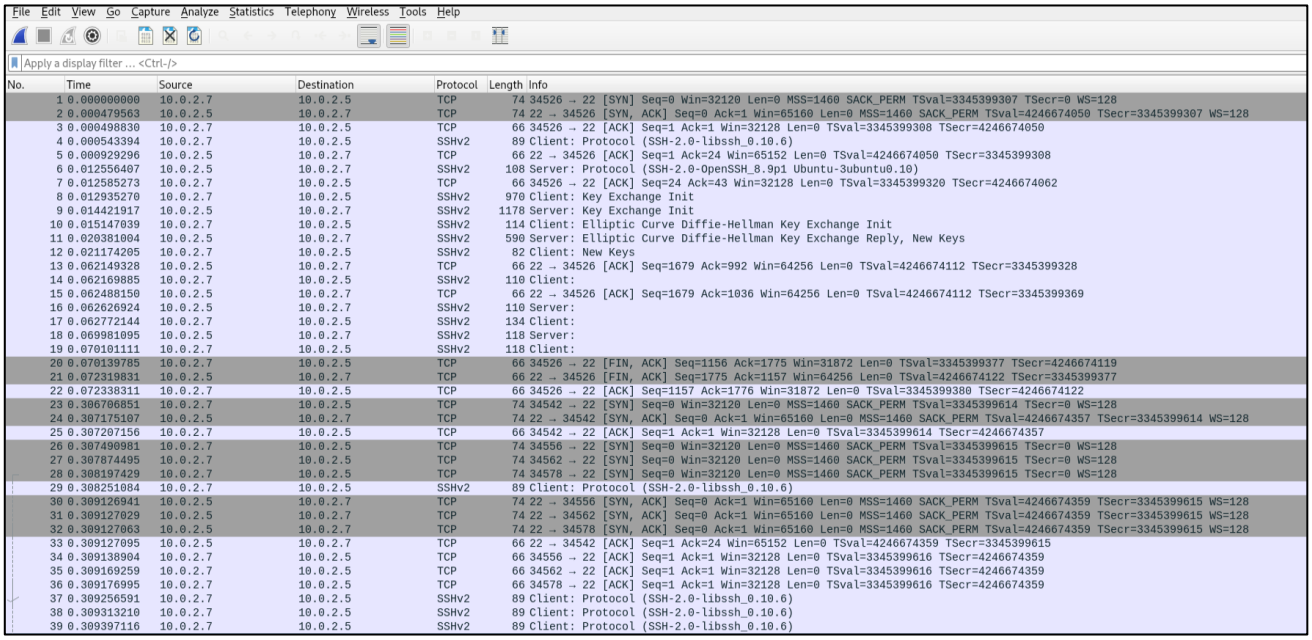
-P /home/manos/Desktop/passwords.txt Υποδεικνύει τη διαδρομή προς το αρχείο πιθανών κωδικών πρόσβασης (λεξικό ή λίστα λέξεων). Το πρόγραμμα θα συμβουλεύεται το passwords.txt για κάθε δοκιμή ταυτοποίησης, σε συνδυασμό με το όνομα χρήστη manos.

-s 22 10.0.2.5 ssh Προσδιορίζονται οι θύρα, IP και υπηρεσία του θύματος.

-t 4 Ορίζει τον αριθμό των παράλληλων νημάτων σε 4. Το Επιχειρούνται έτσι ισάριθμες προσπάθειες σύνδεσης ταυτοχρόνως, αυξάνοντας την ταχύτητα δοκιμών χωρίς να επιβαρύνεται το σύστημα ή ο στόχος.

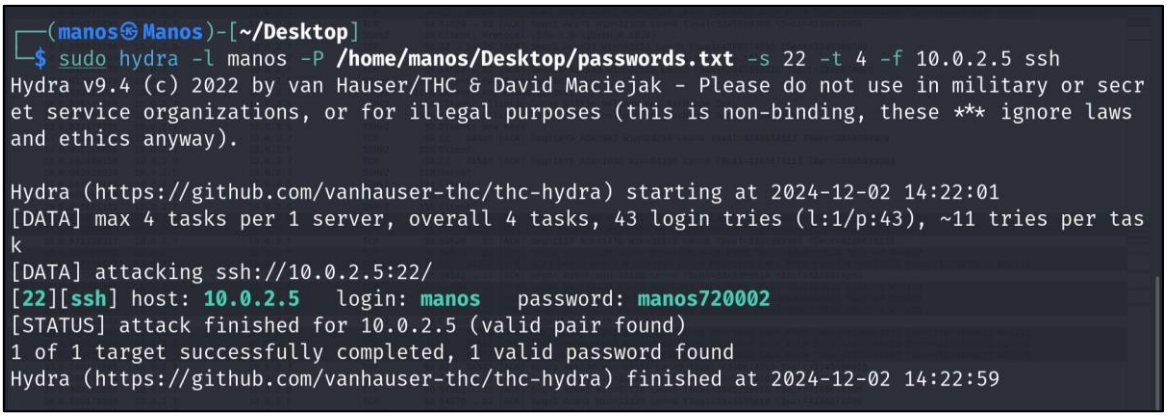
-f Δίνει εντολή στο Hydra να σταματήσει την επίθεση μόλις βρεθεί ένας έγκυρος συνδυασμός ονόματος χρήστη-συνθηματικού.

Εκτελώντας την παραπάνω εντολή στο Attacker VM, το εργαλείο αρχίζει να πραγματοποιεί δοκιμές κωδικών, καταγράφοντας την κακόβουλη κυκλοφορία της Εικόνας 80. Τα πακέτα με αριθμό 4, 37, 38 και 39 είναι αυτά που ανιχνεύονται από το Snort και οδηγούν στην ενεργοποίηση του IPS. Αποτελούν το πρώτο μέρος σύναψης μιας SSH χειραψίας, και περιέχουν την έκδοση SSH του Client, καθώς και διάφορες επιπρόσθετες λεπτομέρειες λογισμικού. Επίσης, το Output του Hydra ύστερα από μια επιτυχημένη επίθεση Bruteforce φαίνεται στην Εικόνα 81.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.7	10.0.2.5	TCP	74	34526 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=3345399307 TSecr=0 WS=128
2	0.000479563	10.0.2.5	10.0.2.7	TCP	74	22 → 34526 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4246674050 TSecr=3345399307 WS=128
3	0.000498830	10.0.2.7	10.0.2.5	TCP	66	34526 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=3345399308 TSecr=4246674050
4	0.000543394	10.0.2.7	10.0.2.5	SSHv2	89	Client: Protocol (SSH-2.0-libssh_0.10.6)
5	0.000929296	10.0.2.5	10.0.2.7	TCP	66	22 → 34526 [ACK] Seq=1 Ack=24 Win=65152 Len=0 TSval=4246674050 TSecr=3345399308
6	0.012556407	10.0.2.5	10.0.2.7	SSHv2	108	Server: Protocol (SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.10)
7	0.012585273	10.0.2.7	10.0.2.5	TCP	66	34526 → 22 [ACK] Seq=24 Ack=43 Win=32128 Len=0 TSval=3345399320 TSecr=4246674062
8	0.012935270	10.0.2.7	10.0.2.5	SSHv2	970	Client: Key Exchange Init
9	0.014421917	10.0.2.5	10.0.2.7	SSHv2	1178	Server: Key Exchange Init
10	0.015147039	10.0.2.7	10.0.2.5	SSHv2	114	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
11	0.020381004	10.0.2.5	10.0.2.7	SSHv2	590	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys
12	0.021174205	10.0.2.7	10.0.2.5	SSHv2	82	Client: New Keys
13	0.062149328	10.0.2.5	10.0.2.7	TCP	66	22 → 34526 [ACK] Seq=1679 Ack=992 Win=64256 Len=0 TSval=4246674112 TSecr=3345399328
14	0.062169885	10.0.2.7	10.0.2.5	SSHv2	110	Client:
15	0.062488150	10.0.2.5	10.0.2.7	TCP	66	22 → 34526 [ACK] Seq=1679 Ack=1036 Win=64256 Len=0 TSval=4246674112 TSecr=3345399369
16	0.062626924	10.0.2.5	10.0.2.7	SSHv2	110	Server:
17	0.062772144	10.0.2.7	10.0.2.5	SSHv2	134	Client:
18	0.069981095	10.0.2.5	10.0.2.7	SSHv2	118	Server:
19	0.070101111	10.0.2.7	10.0.2.5	SSHv2	118	Client:
20	0.070139785	10.0.2.7	10.0.2.5	TCP	66	34526 → 22 [FIN, ACK] Seq=1156 Ack=1775 Win=31872 Len=0 TSval=3345399377 TSecr=4246674119
21	0.072319831	10.0.2.5	10.0.2.7	TCP	66	22 → 34526 [FIN, ACK] Seq=1775 Ack=1157 Win=64256 Len=0 TSval=4246674122 TSecr=3345399377
22	0.072338311	10.0.2.7	10.0.2.5	TCP	66	34526 → 22 [ACK] Seq=1157 Ack=1776 Win=31872 Len=0 TSval=3345399380 TSecr=4246674122
23	0.306706851	10.0.2.7	10.0.2.5	TCP	74	34542 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=3345399614 TSecr=0 WS=128
24	0.307175107	10.0.2.5	10.0.2.7	TCP	74	22 → 34542 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4246674357 TSecr=3345399614 WS=128
25	0.307207156	10.0.2.7	10.0.2.5	TCP	66	34542 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=3345399614 TSecr=4246674357
26	0.307490981	10.0.2.7	10.0.2.5	TCP	74	34556 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=3345399615 TSecr=0 WS=128
27	0.307874495	10.0.2.7	10.0.2.5	TCP	74	34562 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=3345399615 TSecr=0 WS=128
28	0.308197429	10.0.2.7	10.0.2.5	TCP	74	34578 → 22 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=3345399615 TSecr=0 WS=128
29	0.308251084	10.0.2.7	10.0.2.5	SSHv2	89	Client: Protocol (SSH-2.0-libssh_0.10.6)
30	0.309126941	10.0.2.5	10.0.2.7	TCP	74	22 → 34556 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4246674359 TSecr=3345399615 WS=128
31	0.309127029	10.0.2.5	10.0.2.7	TCP	74	22 → 34562 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4246674359 TSecr=3345399615 WS=128
32	0.309127063	10.0.2.5	10.0.2.7	TCP	74	22 → 34578 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4246674359 TSecr=3345399615 WS=128
33	0.309127095	10.0.2.5	10.0.2.7	TCP	66	22 → 34542 [ACK] Seq=1 Ack=24 Win=65152 Len=0 TSval=4246674359 TSecr=3345399615
34	0.309138904	10.0.2.7	10.0.2.5	TCP	66	34556 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=3345399616 TSecr=4246674359
35	0.309169259	10.0.2.7	10.0.2.5	TCP	66	34562 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=3345399616 TSecr=4246674359
36	0.309176995	10.0.2.7	10.0.2.5	TCP	66	34578 → 22 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=3345399616 TSecr=4246674359
37	0.309256591	10.0.2.7	10.0.2.5	SSHv2	89	Client: Protocol (SSH-2.0-libssh_0.10.6)
38	0.309313210	10.0.2.7	10.0.2.5	SSHv2	89	Client: Protocol (SSH-2.0-libssh_0.10.6)
39	0.309397116	10.0.2.7	10.0.2.5	SSHv2	89	Client: Protocol (SSH-2.0-libssh_0.10.6)

Εικόνα 80. Καταγραφή Wireshark για SSH Password Auth. Brute Force Attack



```
┌──(manos㉿Manos)-[~/Desktop]
└─$ sudo hydra -l manos -P /home/manos/Desktop/passwords.txt -s 22 -t 4 -f 10.0.2.5 ssh
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secr
et service organizations, or for illegal purposes (this is non-binding, these *** ignore laws
and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-12-02 14:22:01
[DATA] max 4 tasks per 1 server, overall 4 tasks, 43 login tries (l:1/p:43), ~11 tries per tas
k
[DATA] attacking ssh://10.0.2.5:22/
[22][ssh] host: 10.0.2.5   login: manos   password: manos720002
[STATUS] attack finished for 10.0.2.5 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-12-02 14:22:59
```

Εικόνα 81. Hydra Output

Λειτουργία του IDS/IPS και Αποτελέσματα

Το πρώτο σενάριο λειτουργίας του IDS/IPS (Εικόνα 83) που θα αναλυθεί σε αυτή την επίθεση, προϋποθέτει ότι κατά τη λειτουργία του μηχανισμού ασφαλείας, ο εισβολέας βρίσκεται ακόμα στη φάση δοκιμών και δεν έχει παραβιάσει το θύμα. Το σύστημα ανιχνεύει την ύποπτη κυκλοφορία, εντοπίζει την κακόβουλη IP καθώς και αυτή του Victim1, και προχωράει στην εκτέλεση του Shield. Λόγω της ίδιας της φύσης της, στην επίθεση δεν συμπεριλαμβάνεται η χρήση Spoofed ή Decoy διευθύνσεων, με αποτέλεσμα να μην εκτελείται ανάλογος έλεγχος από το IDS/IPS. Στο Victim1, τροποποιείται το αρχείο ρυθμίσεων του SSH Deamon προκειμένου να αγνοούνται μελλοντικές προσπάθειες σύνδεσης μέσω κωδικού πρόσβασης, και ρυθμίζονται οι κανόνες του Iptables με σκοπό να απορρίπτουν πακέτα από την 10.0.2.7 (Εικόνα 82). Ως αποτέλεσμα των παραπάνω ενεργειών, το Hydra εγκλωβίζεται σε μια κατάσταση συνεχόμενων μη επιτυχημένων προσπαθειών, και παρόλο που ο κωδικός υπάρχει ακόμα στο αρχείο passwords.txt, θα επιστέψει το Output “1 of 1 target completed, 0 valid passwords found”.

Προχωρώντας στο, λιγότερο πιθανό, σενάριο όπου ο εισβολέας κατάφερε να “σπάσει” τον κωδικό του Victim1, και έχει πλέον αποκτήσει απομακρυσμένη πρόσβαση στη μηχανή, όπως φαίνεται και από την αναφορά της Εικόνας 84, η λειτουργία του συστήματος παραμένει επί το πλείστων πανομοιότυπη με το προηγούμενο σενάριο. Κύρια διαφορά αποτελεί η πρόσθετη ενέργεια απομάκρυνσης του εισβολέα από την προστατευόμενη συσκευή, η οποία από τη οπτική γωνία του Terminal στον Attacker φαίνεται ως εξής: “Connection to 10.0.2.5 Closed by Remote Host”.

```
manos@manosvictim1-VirtualBox:~/Desktop$ sudo cat /etc/ssh/sshd_config | grep "PasswordAuthentication"
PasswordAuthentication no
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication, then enable this but set PasswordAuthentication
manos@manosvictim1-VirtualBox:~/Desktop$ sudo iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
   40  2080 DROP       all  --  *      *       10.0.2.7             0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       0.0.0.0/0            10.0.2.7
    0     0 DROP       all  --  *      *       10.0.2.7             0.0.0.0/0

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
   40  4160 DROP       all  --  *      *       0.0.0.0/0            10.0.2.7
manos@manosvictim1-VirtualBox:~/Desktop$
```

Εικόνα 82. SSHD Config File & Iptables Ruleset του Victim

```
manos@manos-VirtualBox:~/Desktop$ ./mainscript.sh

                                                    Starting Up...

                        -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 03/12 02:07:33
 ^^^^ SSH Password Authentication Bruteforce Attempt ^^^^

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:
Source IP(s): 10.0.2.7
Destination -- Protected Network -- IP(s): 10.0.2.5
-- Protected-Netowrk Machine(s) Under Attack: victim1 --


                                                    For victim1...

                           Security Script (Shield) Installed Successfully to the Target System
---> Processing Malicious IP: 10.0.2.7 <---

Attacker is not currently connected to the Machine via SSH...

--- Disabling Password Authentication For SSH Connections ---
Password authentication has been successfully disabled for SSH.

--- Blocking Intruders IP(s) ---
Intruder Blocked. Make sure you enable SSH Password Authentication (if needed) after the Attack is completely Mitigated.
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 83. Generated Report για Bruteforce Attack (Σενάριο 1)

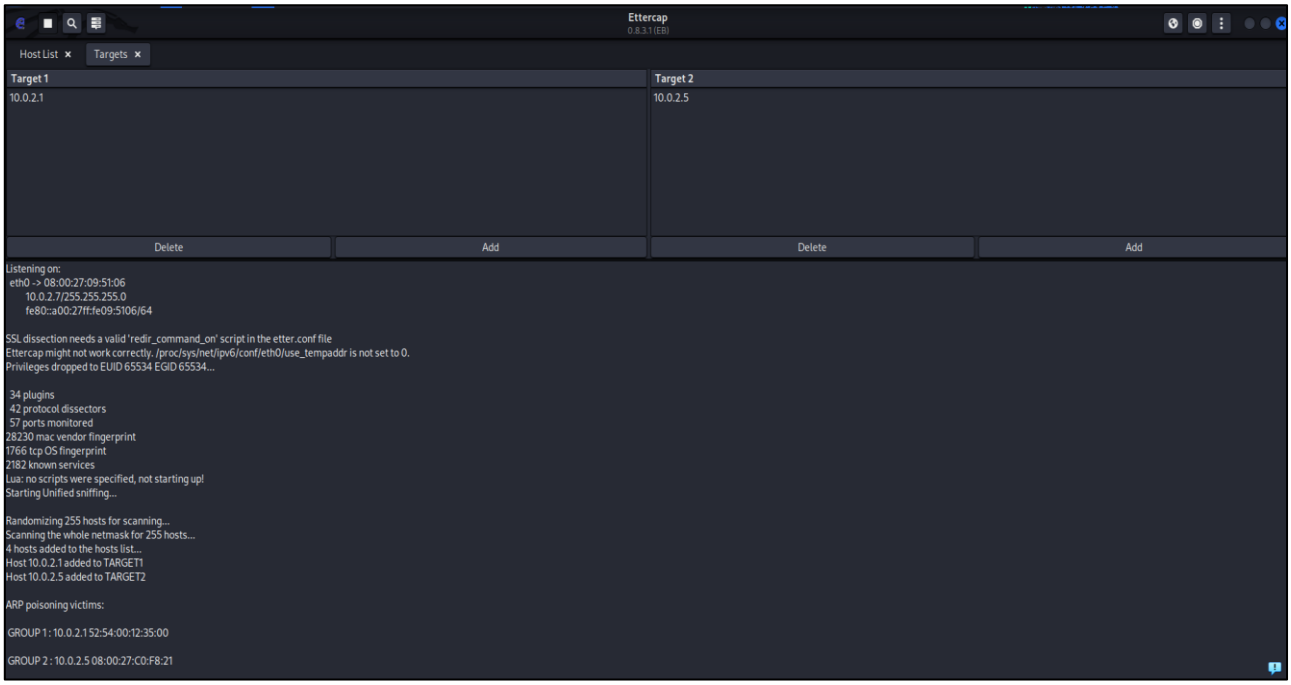
```
                                                    -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 03/12 03:55:59
 ^^^^ SSH Password Authentication Bruteforce Attempt ^^^^

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:
Source IP(s): 10.0.2.7
Destination -- Protected Network -- IP(s): 10.0.2.5
-- Protected-Netowrk Machine(s) Under Attack: victim1 --


                                                    For victim1...

                           Security Script (Shield) Installed Successfully to the Target System
---> Processing Malicious IP: 10.0.2.7 <---

(!) The Attacker is conneted to the Machine via SSH now (!)
Killing the malicious connection..
Done. Attacker Kicked off the System!

--- Disabling Password Authentication For SSH Connections ---
Password authentication has been successfully disabled for SSH.

--- Blocking Intruders IP(s) ---
Intruder Blocked. Make sure you enable SSH Password Authentication (if needed) after the Attack is completely Mitigated.
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 84. Generated Report για Bruteforce Attack (Σενάριο 2)

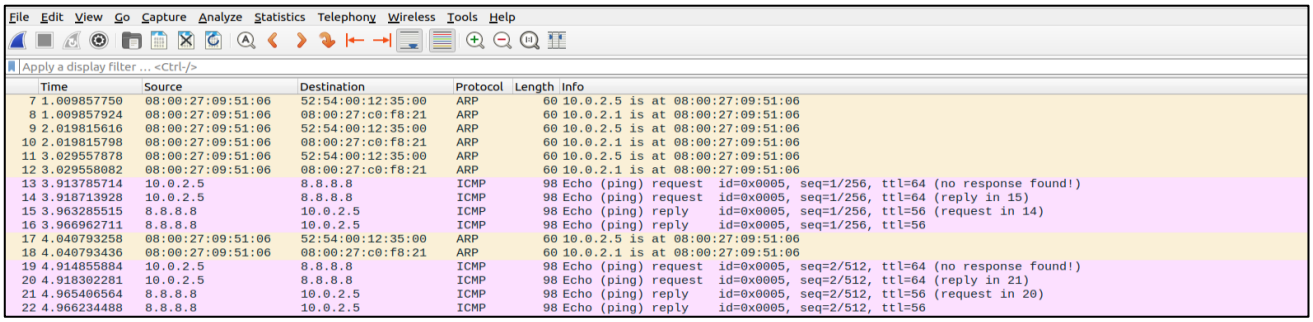
3.3 ARP Spoofing

Συνθήκες Δικτύου

Για τη συγκεκριμένη κατηγορία, επιλέχθηκε η προσομοίωση μιας Full Duplex ARP Spoofing Attack μεταξύ του Attacker και των θυμάτων Router και Victim1. Είναι ο δημοφιλέστερος τύπος επιθέσεων μεταξύ των δύο που συζητήθηκαν στο Θεωρητικό Υπόβαθρο, λόγω της δυνατότητας αναχαίτισης της κυκλοφορίας εκατέρωθεν. Παρέχει έτσι στον επιτιθέμενο πλήρη ορατότητα και έλεγχο των πακέτων, επιτρέποντας πιο προηγμένες επιθέσεις, όπως Malicious Code Injection. Επίσης, η στόχευση του Router ως το συμπληρωματικό θύμα της επίθεσης παρέχει σημαντικά πλεονεκτήματα για τον επιτιθέμενο, καθώς η επικοινωνία με τους δρομολογητές περιλαμβάνει έναν αρκετά ευρύτερο τύπο κυκλοφοριών. Για την αποστολή των παραπλανητικών (Spoofed) πακέτων ARP, εγκαταστάθηκε και ρυθμίστηκε στο Attacker VM το Ettercap. Ένα ευέλικτο εργαλείο που χρησιμοποιείται κυρίως για επιθέσεις Man In The Middle (MITM) σε δίκτυα LAN καθώς είναι σχεδιασμένο προκειμένου να παρακολουθεί και να αλλοιώνει ενεργά την κυκλοφορία. Μέσω του Graphical Interface (sudo ettercap -G) σαρώνεται το δίκτυο για ενεργούς Hosts, επιλέγονται τα θύματα 10.0.2.1 και 10.0.2.5, και στη συνέχεια καθορίζεται ο τύπος επίθεσης (Εικόνα 85). Στο εργαλείο Wireshark της Εικόνας 86 καταγράφονται τα κακόβουλα ARP πακέτα με αριθμό 7 έως 12 και 17/18, τα οποία στέλνονται από το Ettercap με σκοπό να “ξεγελάσει” τους δύο στόχους. Τα πακέτα αυτά παραβιάζουν τις τιμές ζευγαριών του Preprocessor στο Snort του IDS/IPS με αποτέλεσμα την παραγωγή των ανάλογων ειδοποιήσεων. Παράλληλα, στην καταγραφή φαίνεται η διαδικασία υποκλοπής της κυκλοφορίας, καθώς κάθε Request από τη διεύθυνση του Victim1 και Reply από την εξωτερική 8.8.8.8, περνάει πρώτα μέσα από το Attacker VM, πρώτού προωθηθεί στον προορισμό του (echo 1 | sudo tee /proc/sys/net/ip4/ip_forward).



Εικόνα 85. Ettercap ARP Spoofing GUI



	Time	Source	Destination	Protocol	Length	Info
7	1.009857750	08:00:27:09:51:06	52:54:00:12:35:00	ARP	60	10.0.2.5 is at 08:00:27:09:51:06
8	1.009857924	08:00:27:09:51:06	08:00:27:c0:f8:21	ARP	60	10.0.2.1 is at 08:00:27:09:51:06
9	2.019815616	08:00:27:09:51:06	52:54:00:12:35:00	ARP	60	10.0.2.5 is at 08:00:27:09:51:06
10	2.019815798	08:00:27:09:51:06	08:00:27:c0:f8:21	ARP	60	10.0.2.1 is at 08:00:27:09:51:06
11	3.029557878	08:00:27:09:51:06	52:54:00:12:35:00	ARP	60	10.0.2.5 is at 08:00:27:09:51:06
12	3.029558082	08:00:27:09:51:06	08:00:27:c0:f8:21	ARP	60	10.0.2.1 is at 08:00:27:09:51:06
13	3.913785714	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0005, seq=1/256, ttl=64 (no response found!)
14	3.918713928	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0005, seq=1/256, ttl=64 (reply in 15)
15	3.963285515	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0005, seq=1/256, ttl=56 (request in 14)
16	3.966962711	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0005, seq=1/256, ttl=56
17	4.040793258	08:00:27:09:51:06	52:54:00:12:35:00	ARP	60	10.0.2.5 is at 08:00:27:09:51:06
18	4.040793436	08:00:27:09:51:06	08:00:27:c0:f8:21	ARP	60	10.0.2.1 is at 08:00:27:09:51:06
19	4.914855884	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0005, seq=2/512, ttl=64 (no response found!)
20	4.918302281	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0005, seq=2/512, ttl=64 (reply in 21)
21	4.965406564	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0005, seq=2/512, ttl=56 (request in 20)
22	4.966234488	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0005, seq=2/512, ttl=56

Εικόνα 86. Καταγραφή Wireshark για ARP Spoofing και υποκλοπή κυκλοφορίας

Λειτουργία του IDS/IPS και Αποτελέσματα

Ως αντίδραση στις παραπάνω ενέργειες το IDS/IPS εντοπίζει τις MAC των εμπλεκόμενων Hosts και παράλληλα προβαίνει στον ειδικό χειρισμό προετοιμασίας, εφόσον η ARP Cache του Default Gateway έχει αλλοιωθεί. Κατά την εκτέλεση του Shield στο Victim1, το σύστημα ελέγχει τα ζευγάρια IP - MAC των ARP καταγραφών, και διορθώνει μόνο την κακόβουλη (MAC του Attacker). Εφόσον επιβεβαιώθηκε ότι το Router αποτελεί επίσης θύμα της επίθεσης, τροποποιείται ο πίνακας δρομολογήσεων με σκοπό τον καθορισμό του IDS/IPS ως Default Gateway της συσκευής, ενώ τόσο η MAC όσο και η IP του Attacker αποκλείονται από τη μηχανή μέσω του Iptables και Arptables αντιστοίχως. Η παραγόμενη αναφορά για το παραπάνω παρουσιάζεται στην Εικόνα 87, ενώ οι βασικές ρυθμίσεις που πραγματοποιούνται στο Victim1 κατά τη λειτουργία του συστήματος ασφαλείας παρατίθενται στην 88, ως μια γενική σύνοψη του σεναρίου.

```
manos@manos-VirtualBox:

--- Potential ARP Spoof Detected ---

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:
Source MAC(s): 08:00:27:09:51:06
Destination -- Protected Network -- MAC(s): 08:00:27:c0:f8:21 52:54:00:12:35:00
!Router's Cache is Probably Poisoned! Preparing System to act as the Default Gateway if needed.

--- System is Ready ---

Running a Scan to Map the Attacker's MAC(s) back to the Corresponding IP(s)
-> Finished Analyzing. Scan Results Indicate Malicious IP(s): 10.0.2.7 <-
-- Protected-Netowrk Machine(s) Under Attack: victim1 Router --


                                        For victim1...

                    Security Script (Shield) Installed Successfully to the Target System
Checking the System's ARP Cache Based on the Pre - Configured Values..

!! Mismatch for 10.0.2.1: 08:00:27:09:51:06 (Expected: 52:54:00:12:35:00) !!

--> Special handling for the Gateway's IP - 10.0.2.1 <---
Redirecting all traffic for this machine through the IDS/IPS to avoid the routers Poisoned Cache...
Done. Make sure you revert back to normal Network Configurations when the attack is completely mitigated (Network Congestion)

Refitting the System's ARP Cache Entries
Malicious Mac for 10.0.2.1 Dropped and Changed back to to 52:54:00:12:35:00

---- Expected Match found for 10.0.2.4 - 08:00:27:98:39:56. No action needed on this Pair ----

---- IP 10.0.2.5 not found in current ARP Cache. No action needed ----

Blocking known Intruders IP(s)..
IP(s) Blocked

Blocking known  Intruder's Mac Address...
MAC(s) Blocked.
```

Εικόνα 87. Generated Report για Full Duplex ARP Spoofing Attack

Αξίζει επίσης να σημειωθεί ότι ύστερα από το πέρας των παραπάνω ενεργειών, στις καταγραφές Wireshark του επιτιθέμενου δεν φαίνονται πλέον τα ICMP Request – Response μεταξύ των Victim1 και 8.8.8.8 το οποίο αποτελεί δείγμα επιτυχημένης αντιμετώπισης της επίθεσης.

	Before The Attack	**During The Attack**	**After The IDS/IPS Intervention**
ARP Cache Mappings	10.0.2.1 - 52:54:00:12:35:00	10.0.2.1 - 08:00:27:09:51:06	10.0.2.1 - 52:54:00:12:35:00 PERM
Iptables Rules	-	-	DENY ALL TRAFFIC TO/FROM: 10.0.2.7
Arptables Rules	-	-	DENY ALL TRAFFIC TO/FROM: 08:00:27:09:51:06
Routing Table	DEFAULT VIA: 10.0.2.1	DEFAULT VIA: 10.0.2.1	DEFAULT VIA: 10.0.2.4

Εικόνα 88. Σύνοψη Βασικών Ρυθμίσεων στο Victim1

3.4 ICMP Redirection

Συνθήκες Δικτύου

Για την προσομοίωση της παρούσας επίθεσης, επιλέχθηκε σενάριο κατά το οποίο το Attacker VM επιχειρεί μια επίθεση ICMP Redirection στο Victim1 χρησιμοποιώντας μια Spoofed IP. Αναλυτικότερα, επιλέγεται η διεύθυνση του Router ως υποτιθέμενος αποστολέας κακόβουλης κυκλοφορίας, καθώς μια διαδικασία Redirection For Host μπορεί να επιχειρείται αποκλειστικά από δρομολογητές. Για τη μέγιστη δυνατή αποδοτικότητα της επίθεσης, είναι απαραίτητο ο εισβολέας να έχει ήδη πρόσβαση στην υφιστάμενη κυκλοφορία του θύματος, με σκοπό να στέλνει τα πακέτα Redirect τις κατάλληλες στιγμές (όταν δηλαδή το Victim1 επικοινωνεί με διεύθυνση εκτός του VLAN). Με αυτόν τον τρόπο, αποφεύγονται ενδεχόμενες προσπάθειες κατακλυσμού ICMP Redirect πακέτων στο δίκτυο, αυξάνοντας τη μυστικότητα της επίθεσης. Συνεπώς, στις ρυθμίσεις του VirtualBox ενεργοποιείται η λειτουργία χωρίς κριτήρια (Promiscuous Mode) για τον Attacker, και παράλληλα εγκαθίσταται στο VM το Netwox. Το τελευταίο αποτελείται από ένα σύνολο ισχυρών ενσωματωμένων εργαλείων ανοιχτού κώδικα, τα οποία χρησιμοποιούνται με σκοπό την εκτέλεση εργασιών δικτύου, συμπεριλαμβανομένης και της δημιουργίας πακέτων. Συγκεκριμένα, για την επίθεση χρησιμοποιείται η μονάδα Netwox 86 καθώς αυτή είναι που δημιουργεί τα κακόβουλα Redirection πακέτα. Η εντολή που χρησιμοποιείται για την ενεργοποίηση του προγράμματος είναι η εξής.

sudo netwox 86 --device "eth0" --filter "src host 10.0.2.5" --gw 10.0.2.7 --code 1 --src-ip 10.0.2.1 όπου,

--device “eth0” Καθορίζεται η διεπαφή δικτύου από την οποία θα σταλθούν τα πακέτα ICMP Redirect.

--filter “src host 10.0.2.5” Εφαρμόζεται ένα φίλτρο πακέτων για την επιλογή της κυκλοφορίας με βάση την πηγή της, δηλαδή την IP του Victim1.

--gw 10.0.2.7 Ορίζεται η ψεύτικη πύλη (ελεγχόμενη από τον επιτιθέμενο) στην οποία ο στόχος θα πρέπει να ανακατευθύνει την κυκλοφορία του. Το θύμα λοιπόν (10.0.2.5) θα εξαπατηθεί ώστε να δρομολογήσει τα πακέτα του μέσω του 10.0.2.7.

--code 1 Διευκρινίζεται ο κωδικός ανακατεύθυνσης των πακέτων ICMP. Η τιμή “1” αναφέρεται σε συγκεκριμένες IP προορισμού (Redirect For Host), σε αντίθεση με την “0” – Redirect For Network.

--src-ip 10.0.2.1 Καθορίζεται η παραποιημένη IP προέλευσης για τα παραγόμενα πακέτα ανακατεύθυνσης. Η παράμετρος είναι απαραίτητη, καθώς εξασφαλίζει ότι το πακέτο θα αναγνωρίζεται από τον πυρήνα του θύματος ως νόμιμο. Αν η παραποιημένη IP πηγής δεν ταιριάζει με την τρέχουσα πύλη του Victim1 (10.0.2.1), τότε το Linux Kernel θα αγνοήσει την ανακατεύθυνση ως μη έγκυρη και δεν θα ενημερώσει την κρυφή μνήμη δρομολόγησης (Routing Cache).

Επιχειρώντας ICMP Ping από το Victim1 VM προς τον προορισμό 8.8.8.8, και παράλληλα εκτελώντας την παραπάνω εντολή προκύπτει η καταγραφή Wireshark της Εικόνας 89. Σημειώνεται ότι όπως και παραπάνω, στο Attacker VM έχει ενεργοποιηθεί η προώθηση κυκλοφορίας (ip_forward=1). Τα πακέτα με μαύρο χρώμα είναι αυτά που ανακατευθύνουν το θύμα προς την 10.0.2.7 κάθε φορά που στέλνονται πακέτα στο 8.8.8.8. Ομοίως, η διαδικασία μπορεί να επαναληφθεί για πολλαπλές εξωτερικές IP παραλήπτη με μοναδικό σκοπό την αλλοίωση της Routing Cache. Για το παρόν σενάριο το τελευταίο αποτυπώνεται στην Εικόνα 90 (ip route show cache), σε συνδυασμό με το Output της αρχικής εντολής Ping. Όπως φαίνεται και από τα τελευταία πέντε πακέτα Reply, η κυκλοφορία επέστρεψε σε φυσιολογικά επίπεδα τη στιγμή που απενεργοποιήθηκε το Netwox.

	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=1/256, ttl=64 (reply in 2)
2	0.056839887	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256, ttl=56 (request in 1)
3	1.001147944	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=2/512, ttl=64 (reply in 4)
4	1.060027787	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=2/512, ttl=56 (request in 3)
5	2.001846200	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=3/768, ttl=64 (reply in 6)
6	2.058951204	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=3/768, ttl=56 (request in 5)
7	3.003166073	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=4/1024, ttl=64 (reply in 10)
8	3.037511008	08:00:27:09:51:06	ff:ff:ff:ff:ff:ff	ARP	60	Who has 10.0.2.5? Tell 10.0.2.7
9	3.037716082	08:00:27:c0:f8:21	08:00:27:09:51:06	ARP	60	10.0.2.5 is at 08:00:27:c0:f8:21
10	3.065049842	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=4/1024, ttl=56 (request in 7)
11	3.126244282	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
12	4.004805984	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=64 (no response found!)
13	4.005048489	08:00:27:09:51:06	ff:ff:ff:ff:ff:ff	ARP	60	Who has 10.0.2.1? Tell 10.0.2.7
14	4.005048575	52:54:00:12:35:00	08:00:27:09:51:06	ARP	60	10.0.2.1 is at 52:54:00:12:35:00
15	4.005192619	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=63 (reply in 18)
16	4.029250472	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
17	4.029250706	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
18	4.062042765	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=5/1280, ttl=56 (request in 15)
19	5.005666559	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=6/1536, ttl=64 (no response found!)
20	5.005850818	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=6/1536, ttl=63 (reply in 23)
21	5.036823979	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
22	5.036824182	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
23	5.066264043	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=6/1536, ttl=56 (request in 20)
24	5.450024247	08:00:27:c0:f8:21	52:54:00:12:35:00	ARP	60	Who has 10.0.2.1? Tell 10.0.2.5
25	5.450920323	52:54:00:12:35:00	08:00:27:c0:f8:21	ARP	60	10.0.2.1 is at 52:54:00:12:35:00
26	6.007437867	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=7/1792, ttl=64 (no response found!)
27	6.007438060	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=7/1792, ttl=63 (reply in 30)
28	6.044342655	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
29	6.044342986	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
30	6.078119496	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=7/1792, ttl=56 (request in 27)
31	7.008964004	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=8/2048, ttl=64 (no response found!)
32	7.009168761	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=8/2048, ttl=63 (reply in 35)
33	7.051613430	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
34	7.051613628	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
35	7.066519158	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=8/2048, ttl=56 (request in 32)
36	7.421693047	fe80::a00:27ff:fec0…	ff02::2	ICMPv6	70	Router Solicitation from 08:00:27:c0:f8:21
37	8.010299751	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=9/2304, ttl=64 (no response found!)
38	8.010531712	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=9/2304, ttl=63 (reply in 41)
39	8.058789799	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
40	8.058790033	10.0.2.1	10.0.2.5	ICMP	70	Redirect (Redirect for host)
41	8.067386872	8.8.8.8	10.0.2.5	ICMP	98	Echo (ping) reply id=0x0001, seq=9/2304, ttl=56 (request in 38)
42	9.012248672	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=10/2560, ttl=64 (no response found!)
43	9.012430597	10.0.2.5	8.8.8.8	ICMP	98	Echo (ping) request id=0x0001, seq=10/2560, ttl=63 (reply in 48)

Εικόνα 89. Καταγραφή Wireshark για ICMP Redirect For Host

```
manos@manosvictim1-VirtualBox:~/Desktop$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=56 time=57.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=56 time=59.1 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=56 time=57.4 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=56 time=62.1 ms
From 10.0.2.1 icmp_seq=4 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=5 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=5 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=5 ttl=56 time=57.5 ms
From 10.0.2.1 icmp_seq=6 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=6 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=6 ttl=56 time=60.9 ms
From 10.0.2.1 icmp_seq=7 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=7 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=7 ttl=56 time=70.9 ms
From 10.0.2.1 icmp_seq=8 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=8 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=8 ttl=56 time=58.1 ms
From 10.0.2.1 icmp_seq=9 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=9 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=9 ttl=56 time=57.4 ms
From 10.0.2.1 icmp_seq=10 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=10 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=10 ttl=56 time=353 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=56 time=59.5 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=56 time=157 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=56 time=58.2 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=56 time=57.2 ms
^C
--- 8.8.8.8 ping statistics ---
14 packets transmitted, 14 received, +13 errors, 0% packet loss, time 13019ms
rtt min/avg/max/mdev = 57.237/87.530/352.691/77.721 ms
manos@manosvictim1-VirtualBox:~/Desktop$ ip route show cache
8.8.8.8 via 10.0.2.7 dev enp0s3
    cache <redirected> expires 182sec
```

Εικόνα 90. Σύνοψη Επίθεσης ICMP Redirect

Λειτουργία του IDS/IPS και Αποτελέσματα

Τα προαναφερόμενα πακέτα ανακατεύθυνσης ενεργοποιούν το μηχανισμό άμυνας, από τον οποίο προκύπτει και η αναφορά της Εικόνας 91. Το σύστημα εντόπισε ότι η IP προέλευσης είναι Spoofed και προέβη στην εκτέλεση του Shield στο Victim1. Εκεί, ύστερα από έλεγχο της Routing Cache, ανέδειξε την κακόβουλη IP (10.0.2.7), στην οποία καταλήγει η ανακατευθυνόμενη κυκλοφορία προς το 8.8.8.8. Προχώρησε με την προσθήκη της διεύθυνσης του Attacker στους κανόνες απόρριψης των Iptables, αλλά και της γενικής απαγόρευσης των πακέτων ICMP Redirect από οποιαδήποτε διεύθυνση (και μέσω του Kernel). Τέλος, έσβησε όλες τις υπάρχουσες καταγραφές της Routing Cache. Αποτέλεσμα αυτών των ενεργειών είναι η εξομάλυνση της κυκλοφορίας του ICMP Ping, παρά το γεγονός ότι το Netwox παραμένει ενεργό, αλλά και η διόρθωση της Cache. Τα παραπάνω σε συνδυασμό με το Output του Iptables στο Victim1, παρατίθενται στην Εικόνα 92.

```
manos@manos-VirtualBox: ~/Desktop

              -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 04/12 02:53:25

ICMP Redirect Packet(s) Detected *** Could be Malicious ***


Anilyzing Snort Logs to Extract Crucial Information about the Attack


Logs Check Came back with the Following:

Source IP(s): 10.0.2.1

Destination -- Protected Network -- IP(s): 10.0.2.5


Checking source IP(s) against pre-Configured Network Variables
and running a Network Scan to sort out the Spoofed IP(s)

All Source IP(s) match known network IP(s). Definitely spoofed!

-- Protected-Netowrk Machine(s) Under Attack: victim1 --



                                        For victim1...


                  Security Script (Shield) Installed Successfully to the Target System

--> Checking System's Routing Cache <--


Check Reported back with..

Destination IP(s) - 8.8.8.8 -
Are being redirected to Gateway(s): 10.0.2.7

-> 10.0.2.7 is not a trusted Gateway address! Blocking it now <-

Intruder(s) Blocked!


### Flushing Routing Cache and Configuring Kernel to Drop ICMP Redirection Packets From all Insecure Sources ###
Make sure to change this Configuration once the threat is Mitigated

Done. Traffic will be redirected to the Default Gateway.
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 91. Generated Report για ICMP Redirection For Host

```
manos@manosvictim1-VirtualBox: ~/Desktop
From 10.0.2.1 icmp_seq=13 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=13 ttl=56 time=84.7 ms
From 10.0.2.1 icmp_seq=14 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=14 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=14 ttl=56 time=63.8 ms
From 10.0.2.1 icmp_seq=15 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=15 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=15 ttl=56 time=58.4 ms
From 10.0.2.1 icmp_seq=16 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=16 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=16 ttl=56 time=57.1 ms
From 10.0.2.1 icmp_seq=17 Redirect Host(New nexthop: 10.0.2.7)
From 10.0.2.1 icmp_seq=17 Redirect Host(New nexthop: 10.0.2.7)
64 bytes from 8.8.8.8: icmp_seq=17 ttl=56 time=56.8 ms
64 bytes from 8.8.8.8: icmp_seq=18 ttl=56 time=170 ms
64 bytes from 8.8.8.8: icmp_seq=19 ttl=56 time=57.2 ms
64 bytes from 8.8.8.8: icmp_seq=20 ttl=56 time=57.6 ms
64 bytes from 8.8.8.8: icmp_seq=21 ttl=56 time=56.3 ms
64 bytes from 8.8.8.8: icmp_seq=22 ttl=56 time=57.6 ms
64 bytes from 8.8.8.8: icmp_seq=23 ttl=56 time=56.9 ms
64 bytes from 8.8.8.8: icmp_seq=24 ttl=56 time=66.2 ms
^C
--- 8.8.8.8 ping statistics ---
24 packets transmitted, 24 received, +27 errors, 0% packet loss, time 23035ms
rtt min/avg/max/mdev = 56.308/91.323/330.613/73.107 ms
manos@manosvictim1-VirtualBox:~/Desktop$ ip route show cache
8.8.8.8 via 10.0.2.1 dev enp0s3
    cache
manos@manosvictim1-VirtualBox:~/Desktop$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    7   392 DROP       icmp --  any    any     anywhere             anywhere             icmp redirect
    0     0 DROP       all  --  any    any     10.0.2.7             anywhere

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  any    any     anywhere             10.0.2.7
    0     0 DROP       all  --  any    any     10.0.2.7             anywhere

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  any    any     anywhere             10.0.2.7
manos@manosvictim1-VirtualBox:~/Desktop$
```

Εικόνα 92. Αποτελέσματα Ενεργειών του IDS/IPS

3.5 Denial of Service

Συνθήκες Δικτύου

Η προσομοίωση μιας ρεαλιστικής επίθεσης άρνησης υπηρεσιών στο εικονικό δίκτυο είναι αδύνατη, λόγω έλλειψης πόρων υλικού. Το Laptop στο οποίο φιλοξενείται το περιβάλλον πειράματος δεν διαθέτει την απαραίτητη υπολογιστική ισχύ για την παράλληλη εκτέλεση και αντιμετώπιση επιθέσεων μεγάλης κλίμακας. Ωστόσο, υπενθυμίζεται ότι το λογισμικό ασφάλειας έχει σχεδιαστεί για να είναι σε θέση να διαχειρίζεται πολλαπλές IP προέλευσης (Distributed Denial of Service – DDoS), ανεξάρτητα από το περιβάλλον λειτουργίας του. Επιλέχθηκε συνεπώς σενάριο κατά το οποίο το Attacker VM κατακλύζει με πακέτα κυκλοφορίας το Victim1, χρησιμοποιώντας ένα εκ των τριών πρωτοκόλλων που συζητήθηκαν στο Θεωρητικό Υπόβαθρο της αντίστοιχης επίθεσης. Για την παραγωγή των πλημμυρών στο δίκτυο, χρησιμοποιήθηκε από τη μηχανή του επιτιθέμενου το Hping3, ένα ευέλικτο πρόγραμμα δημιουργίας και αποστολής εξατομικευμένων πακέτων πολλαπλών πρωτοκόλλων (Εικόνα 93).

Λειτουργία του IDS/IPS και Αποτελέσματα

Ο μηχανισμός άμυνας λειτουργεί ικανοποιητικά δεδομένων των συνθηκών που τέθηκαν παραπάνω. Σύμφωνα με τα δεδομένα της Εικόνας 94, τα οποία προέκυψαν από τους μέσους όρους πειραματικών μετρήσεων, η χρήση της διαθέσιμης CPU και εύρους ζώνης διατηρήθηκε σε χαμηλότερα επίπεδα συγκριτικά με το διάστημα κορύφωσης των τριών επιθέσεων DoS. Ενδεικτικά, στην Εικόνα 95 παρατίθεται η αναφορά που προέκυψε ύστερα από μια πλημμύρα τύπου ICMP - PING. Σημειώνεται ως μέρος της διαδικασίας αντιμετώπισης, η επιλογή που δίνεται στο διαχειριστή ασφάλειας, αναφορικά με την αναστολή της λειτουργίας του Victim1. Σε αυτή την περίπτωση, το πρόγραμμα θα τερματιζόταν πρόωρα, ύστερα από την απενεργοποίηση του θύματος. Το σύστημα ωστόσο, εκτός από τη δυνατότητα να εντοπίζει τις IP προέλευσης πλημμυρών, διαθέτει παράλληλο μηχανισμό ελέγχου των παραποιημένων διευθύνσεων. Εάν προκύψουν ταιριάσματα με IP γνωστών συσκευών του δικτύου, δεν είναι δυνατή η εφαρμογή κανόνων απόκλισης, και αντί αυτού επιχειρείται μόνο Rate Limiting στο αντίστοιχο πρωτόκολλο κυκλοφορίας. Το τελευταίο αποτυπώνεται από το Report που προέκυψε ύστερα από μια UDP – DNS Flood, με Spoofed IP αυτή του Router (Εικόνα 96). Για την αποφυγή του επερχόμενου Scan από το IDS/IPS VM, ρυθμίζονται κατάλληλα τα ARP Tables του Attacker (sudo arptables -A INPUT --src-mac 08:00:27:98:39:56 -j DROP

Flood Type	Terminal Command	Action
TCP - SYN	sudo hping3 -S -p 80 --flood 10.0.2.5	Πλημμυρίζει με πακέτα τύπου SYN την υπηρεσία Web (Apache Software) του Victim1 στη θύρα 80. Το θύμα για κάθε αίτηση σύνδεσης απαντάει με αντίστοιχα SYN/ACK.
UDP - DNS	sudo hping3 --udp -p 53 --file dns_querry.bin -d 32 --flood 10.0.2.5	Πλημμυρίζει την υπηρεσία DNS του Victim1 στη θύρα 53 (Bind9 Software) με κακοσχηματισμένα πακέτα χρησιμοποιώντας το dns_querry Binary File. Το θύμα για κάθε παραμορφωμένη αίτηση Querry απαντάει με αντίστοιχα διαγνωστικά πακέτα.
ICMP - PING	sudo hping3 --icmp --flood 10.0.2.5	Πλημμυρίζει το Victim1 VM με Echo Ping Requests, αναγκάζοντάς το να απαντήσει με ανάλογα πακέτα Response.
Spoofed	sudo hping3 {flood type} -a {spoofed ip address} 10.0.2.5	Ανάλογα με τον τύπο πλημμύρας που θα επιλεχθεί, στα πακέτα προέλευσης του επιτιθέμενου καθώς και στις αντίστοιχες απαντήσεις του θύματος θα αναγράφεται η πλαστογραφημένη IP.

Εικόνα 93. Σύνοψη πλημμυρών Hping3

	Average CPU Usage (Total %)	Bandwidth Usage Distribution (RX – TX) of Total Sample 537 MB
Peak of The Attack	40,33%	306 MB – 231 MB
After Blocking the Attacker(s)	25,33%	537 MB – 0 MB
After Implementing Rate Limiting	29,66%	530MB – 7 MB

Εικόνα 94. Στατιστική Απεικόνιση Λειτουργίας του IDS/IPS σε DoS

```
manos@manos-VirtualBox: ~/Desktop
                    -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 09/12 04:14:07

*** Denial Of Service (DOS) Attempt ***

--- Type: ICMP Echo - Request Flood Attack ---

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:

Source IP(s): 10.0.2.7

Destination -- Protected Network -- IP(s): 10.0.2.5


** Recommended ** Do you want to shut down the System(s) under Attack? (yes/no): no

 ----> Anti-DOS Mode Enabled. Will try to handle the Flood <----

Will block all traffic from/to identified malicious IPs and implement Rate Limiting for the corresponding protocols,
to smooth out the bandwidth usage and alleviate the CPU(s). These configurations will help prevent further strain,
allowing the Victim System(s) to maintain stability and prevent further resource exhaustion until the attack subsides.


 >>>  Will try to find the original Attacker's IP(s) <<<

Checking source IP(s) against pre-Configured Network Variables
and running a Network Scan to sort out the Spoofed IP(s)

Finished Analyzing. Will utilize Malicious IP(s): 10.0.2.7

-- Protected-Netowrk Machine(s) Under Attack: victim1 --



                                                 For victim1...

                    Security Script (Shield) Installed Successfully to the Target System


*** Attacker was Blocked and Rate Limiting for ICMP Traffic was applied Successfully ***
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 95. Generated Report για TCP - SYN Flood

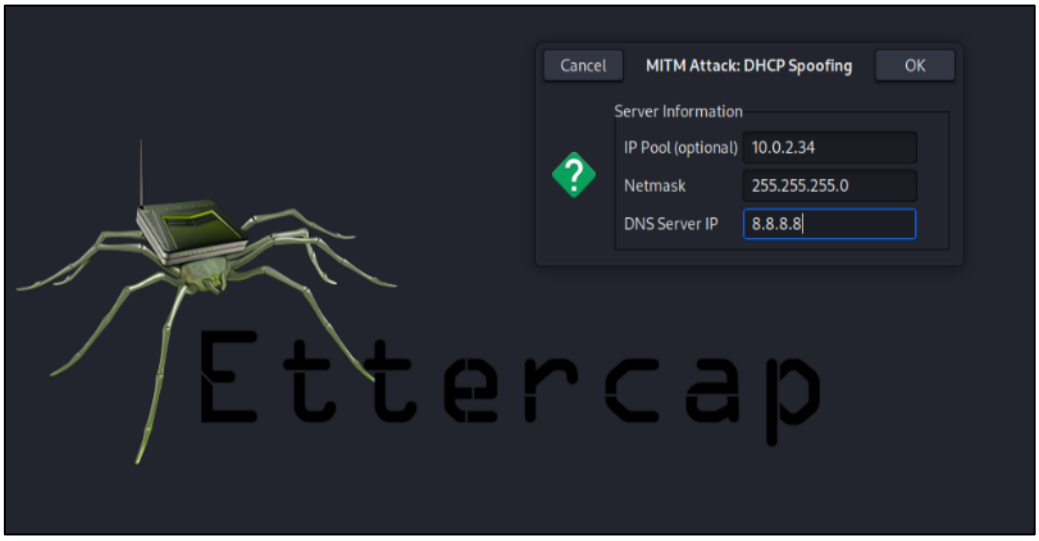
```
manos@manos-VirtualBox: ~/Desktop
manos@manos-VirtualBox: ~/Desktop
Snort Detected Malicious Traffic at 09/12 05:03:57

*** Denial Of Service (DOS) Attempt ***

--- Type: UDP Request Flood Attack ---

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:

Source IP(s): 10.0.2.1

Destination -- Protected Network -- IP(s): 10.0.2.5


** Recommended ** Do you want to shut down the System(s) under Attack? (yes/no): no

 ----> Anti-DOS Mode Enabled. Will try to handle the Flood <----

Will block all traffic from/to identified malicious IPs and implement Rate Limiting for the corresponding protocols,
to smooth out the bandwidth usage and alleviate the CPU(s). These configurations will help prevent further strain,
allowing the Victim System(s) to maintain stability and prevent further resource exhaustion until the attack subsides.


 >>>  Will try to find the original Attacker's IP(s) <<<

Checking source IP(s) against pre-Configured Network Variables
and running a Network Scan to sort out the Spoofed IP(s)

Unable to pinpoint any Source IPs!! Only Rate Limiting
for the UDP Protocol will be utilized moving forward

-- Protected-Netowrk Machine(s) Under Attack: victim1 --



                                                 For victim1...

                    Security Script (Shield) Installed Successfully to the Target System


*** Rate Limiting was applied Successfully for UDP Traffic ***
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 96. Generated Report για UDP Flood με Spoofed IP

3.6 DHCP Spoofing

Συνθήκες Δικτύου

Το λογισμικό IDS/IPS έχει σχεδιαστεί προκειμένου να αναγνωρίζει μια επίθεση DHCP Spoofing σε οποιοδήποτε εκ των δύο σταδίων που αναλύθηκαν στο πλαίσιο των συνθηκών ανταγωνισμού, μεταξύ των κακόβουλων και εξουσιοδοτημένων διακομιστών DHCP. Επιλέχθηκε σενάριο κατά το οποίο το Attacker VM (10.0.2.7) έχει κερδίσει τις Race Conditions με τον DHCP Server του δικτύου (10.0.2.3). Το Victim1 VM επιλέγει εν τάχει το κακόβουλο Offer κατά τη διάρκεια μιας διαδικασίας ανάθεσης διεύθυνσης IP, και η επίθεση ολοκληρώνεται επιτυχώς. Για την παροχή των παραπάνω υπηρεσιών, χρησιμοποιήθηκε εκ νέου το εργαλείο Ettercap, και ρυθμίστηκε κατάλληλα με σκοπό να ακούει για συνδέσεις DHCP. Όπως φαίνεται και στην Εικόνα 97, ο Attacker θα αναθέσει στο θύμα του την IP 10.0.2.34/24 με “κακόβουλο” DNS Server τον 8.8.8.8. Ως διεύθυνση Default Gateway, το Ettercap επιλέγει την ίδια IP με τη μηχανή στην οποία εκτελείται, δηλαδή αυτή του Attacker. Καταγράφοντας τη διαδικασία DHCP, προκύπτει το Wireshark Output της Εικόνας 98, ενώ στην 99 παρατίθενται οι νέες ρυθμίσεις δικτύου του Victim1, όπως προέκυψαν από τις ενέργειες του Ettercap.



Εικόνα 97. Ettercap DHCP Configurations

30	50.884078928	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xc3e7356a
31	50.884540584	10.0.2.3	255.255.255.255	DHCP	590	DHCP Offer - Transaction ID 0xc3e7356a
32	50.891575509	10.0.2.7	255.255.255.255	DHCP	582	DHCP Offer - Transaction ID 0xc3e7356a
33	50.892276813	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0xc3e7356a
34	50.903682751	10.0.2.7	255.255.255.255	DHCP	582	DHCP ACK - Transaction ID 0xc3e7356a

Εικόνα 98. Καταγραφή Wireshark Διαδικασίας DHCP

```
manos@manosvictim1-VirtualBox:/etc/netplan$ ip addr show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c0:f8:21 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.34/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 1721sec preferred_lft 1721sec
    inet6 fe80::214b:bc8a:4499:8022/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
manos@manosvictim1-VirtualBox:/etc/netplan$ ip route
default via 10.0.2.7 dev enp0s3
10.0.0.0/8 dev enp0s9 proto kernel scope link src 10.2.2.8 metric 101
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.34
169.254.0.0/16 dev enp0s9 scope link metric 1000
manos@manosvictim1-VirtualBox:/etc/netplan$ sudo resolvectl status | grep "DNS Servers"
       DNS Servers: 8.8.8.8
manos@manosvictim1-VirtualBox:/etc/netplan$
```

Εικόνα 99. Ρυθμίσεις Victim1 ύστερα από DHCP Spoofing Attack

Λειτουργία του IDS/IPS και Αποτελέσματα

Ο μηχανισμός ασφάλειας του IDS/IPS VM ενεργοποιείται κατά την ανίχνευση των πακέτων OFFER και ACK, με αποτέλεσμα την αναφορά της Εικόνας 100. Το σύστημα εντόπισε την IP του αποστολέα (10.0.2.7), ενώ όσον αφορά τον παραλήπτη, ανίχνευσε ότι πρόκειται για το Victim1 VM με συνηθισμένη τιμή IP την 10.0.2.5. Δημιούργησε το αρχείο yaml στατικών αναθέσεων IP, και με την έγκριση (yes) του διαχειριστή το εφάρμοσε στο Shield του θύματος. Αποτέλεσμα των παρεμβάσεων είναι οι νέες ρυθμίσεις δικτύου της Εικόνας 101, με στατική πλέον διεύθυνση την 10.0.2.50 (Emergency IP) και την επαναφορά των 10.0.2.1 και 192.168.1.1/1.1.1.1 ως προεπιλεγμένες IP Default Gateway και DNS Servers. Όσο για τις αντίστοιχες, κακόβουλες διευθύνσεις που ανιχνεύθηκαν κατά τη διαδικασία της αντιμετώπισης (10.0.2.7 – 8.8.8.8) ταξινομήθηκαν και προστέθηκαν στους κανόνες απόρριψης όλων των τύπων κυκλοφορίας των Iptables.

```
                                        Starting Up...

                    -- Defensive System Initialized -- Waiting for Snort to Generate Alerts --

Snort Detected Malicious Traffic at 09/12 17:45:35

*** Detected DHCP Offer Packet From Untrusted Source ***

--- Potential Rogue DHCP Server ---

Anilyzing Snort Logs to Extract Crucial Information about the Attack

Logs Check Came back with the Following:

Source IP(s): 10.0.2.7

Destination -- Protected Network -- IP(s): 10.0.2.5

Netplan Configuration file (01-network-manager-all.yaml.victim1) created for victim1
to pass emergencystatic IP address 10.0.2.50/24. Sending it now!

>> Done. Will use this File later if needed. <<

-- Protected-Netowrk Machine(s) Under Attack: victim1 --


                                        For victim1...

                        Security Script (Shield) Installed Successfully to the Target System

--- Checking the Gateway and DNS Servers IPs ---

!! Untrusted Gateway Found: 10.0.2.7

!! Untrusted DNS Server Found: 8.8.8.8

Do you want to apply the Static IP Configuration file that was created earlier for this System? (yes/no): yes

Flushing all Malicious Configurations on the Interface First..

Done! Static IP is redy to be Applied

Blocking all Malicious Identified IPs..

Blocked!

Applying new Neplan Network Configurations and Exiting

Make sure to enable DHCP Services (If Needed) when the attack is completely Mitigated!
Connection to 10.2.2.8 closed.
manos@manos-VirtualBox:~/Desktop$
```

Εικόνα 100. Generated Report για DHCP Spoofing

```
manos@manosvictim1-VirtualBox:/etc/netplan$ ip addr show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c0:f8:21 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.50/24 brd 10.0.2.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a85f:46b3:80af:5f26/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
manos@manosvictim1-VirtualBox:/etc/netplan$ ip route
default via 10.0.2.1 dev enp0s3 proto static
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.50
10.2.2.0/24 dev enp0s9 proto kernel scope link src 10.2.2.8
manos@manosvictim1-VirtualBox:/etc/netplan$ sudo resolvectl status | grep "DNS Servers"
         DNS Servers: 192.168.1.1 1.1.1.1
manos@manosvictim1-VirtualBox:/etc/netplan$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  10.0.2.7             anywhere
DROP       all  --  dns.google           anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  anywhere             10.0.2.7
DROP       all  --  10.0.2.7             anywhere
DROP       all  --  anywhere             dns.google
DROP       all  --  dns.google           anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  anywhere             10.0.2.7
DROP       all  --  anywhere             dns.google
manos@manosvictim1-VirtualBox:/etc/netplan$
```

Εικόνα 101. Ρυθμίσεις Victim1 ύστερα από την παρέμβαση IDS/IPS

4

ΒΙΛΒΙΟΓΡΑΦΙΚΗ ΕΠΙΣΚΟΠΙΣΗ

Ζητήματα που αφορούν τη σχεδίαση, την ανάπτυξη και την ενσωμάτωση των IDS/IPS σε δίκτυα διαφόρων λειτουργιών και απαιτήσεων έχουν μελετηθεί εκτενώς από την παγκόσμια βιβλιογραφία. Σε αυτό το Κεφάλαιο, θα παρουσιαστούν συνοπτικά ορισμένες από αυτές τις προσεγγίσεις, ενώ κάθε πηγή θα αξιολογηθεί συγκριτικά με τις λειτουργίες και τους μηχανισμούς άμυνας που μελετήθηκαν στην παρούσα εργασία. Ανάλογος τρόπος παρουσίασης θα πραγματοποιηθεί και σε βιβλιογραφικά άρθρα στα οποία προτείνονται συγκεκριμένες μεθοδολογίες αντιμετώπισης των επιθέσεων που μελετήθηκαν έως τώρα.

4.1 Μελέτες Σχεδιασμού και Υλοποίησης Συστημάτων IDS/IPS

Στο [33] προτείνεται ένα υβριδικό Intrusion Detection System, το οποίο βασίζεται σε δύο βασικές στρατηγικές ανίχνευσης που αναλύθηκαν παραπάνω, τις Anomaly και Signature Based. Συγκεκριμένα, οι συγγραφείς μελετούν τη βελτίωση του Snort, μέσω της ενσωμάτωσης τεχνικών μηχανικής μάθησης με σκοπό την ελαχιστοποίηση των ψευδώς θετικών ανιχνεύσεων. Το σύστημα που αναπτύχθηκε, βασίστηκε σε αλγορίθμους όπως Support Vector Machines (SVM) κα σε ανάλυση δεδομένων από γνωστά Datasets, όπως τα DARPA IDS. Έδειξε σημαντική μείωση των ψευδώς θετικών και αρνητικών συναγερμών (FPR - FNR) όταν εφαρμόστηκαν συνδυασμοί αλγορίθμων, όπως το Ensemble SVM και οι βελτιστοποιημένες εκδόσεις του με Firefly και Ant Colony Optimization. Ωστόσο, τα συστήματα που βασίζονται σε τεχνικές μηχανικής μάθησης εκ φύσεως αποτελούν Black – Box τεχνολογίες. Τα δεδομένα εξόδου λοιπόν δεν είναι εύκολα ερμηνεύσιμα, καθιστώντας μη αποτελεσματική μια ενδεχόμενη ενσωμάτωσή τους με συστήματα πρόληψης εισβολών (IPS). Στη συγκεκριμένη μελέτη όπου η γρήγορη ανάλυση των δεδομένων ανίχνευσης από τις επερχόμενες μονάδες αντιμετώπισης είναι μεγίστης σημασίας, προτιμάται μια αυστηρώς Signature Based προσέγγιση σχεδιασμού.

Στο [34] μελετάται η ενσωμάτωση του Snort και του NTOP με σκοπό τη δημιουργία συστήματος ανίχνευσης εισβολών σε δίκτυα (Network Based Detection). Ο βασικός σχεδιασμός περιλαμβάνει τρία κύρια μέρη: την απόκτηση δεδομένων, την επεξεργασία τους, καθώς και το κέντρο ελέγχου. Το Snort αναλαμβάνει εξ ολοκλήρου την ανίχνευση επιθέσεων με βάση συγκεκριμένο Ruleset (Signature Based Intrusion Detection System), ενώ το NTOP προσφέρει δυνατότητες παρακολούθησης και ανάλυσης της ροής δεδομένων. Με χρήση τεχνικών όπως το NAPI, για βελτίωση σε λήψη εισερχόμενης κυκλοφορίας, το σύστημα παρέμεινε ανθεκτικό όσον αφορά την απώλεια πακέτων, με ποσοστό λιγότερο από 5% σε συνθήκες αυξημένης κινητικότητας. Σε αντίθεση με το [33], διαθέτει ερμηνεύσιμα δεδομένα εξόδου με αποτελεσματικό τρόπο, ωστόσο αυτά δεν αξιοποιούνται από κάποιον πρόσθετο μηχανισμό αντιμετώπισης των επιθέσεων, όπως η τεχνολογία Dynamic Shield Deployment. Το τελευταίο αφορά και τη υλοποίηση του IDS στο [35], όπου η ανίχνευση των επιθέσεων βασίζεται στους κανόνες της κοινότητας του Snort (Community Rules). Παρά την συχνή ανανέωση του Ruleset με καινοτόμους και αποτελεσματικούς κανόνες, είναι απαραίτητη η ενσωμάτωση εξατομικευμένων υπογραφών με σκοπό την κάλυψη των ειδικών απαιτήσεων στα δίκτυα.

Μηχανισμός παράλληλης ανίχνευσης και ενεργής αντιμετώπισης επιθέσεων (IDS/IPS) σε επίπεδο δικτύων υλοποιείται στο [36], με συνδυασμό των εργαλείων Snort και Netfilter. Το μοντέλο περιλαμβάνει τρείς βασικές μονάδες. Αυτή της ανίχνευσης εισβολών (Snort), τον έλεγχο πολιτικής (Policy Control), και ένα προγραμματιζόμενο Firewall. Το σύστημα εντοπίζει τις απειλές μέσω του Snort και στη συνέχεια ενημερώνει δυναμικά τους κανόνες του Firewall με τη χρήση του Netfilter, για την αποτροπή της επιθετικής δραστηριότητας. Ο μηχανισμός λειτουργίας του συστήματος λειτουργεί παρόμοια με αυτόν της παρούσας εργασίας, ωστόσο όπως αποδείχθηκε και στα παραπάνω Κεφάλαια, ο αποκλεισμός του επιτιθέμενου δεν είναι σε όλες τις περιπτώσεις επαρκής λύση για την πλήρη καταστολή ορισμένων επιθέσεων. Επιπλέον, ο συγγραφέας δεν προτείνει κάποιον τρόπο εντοπισμού των τεχνικών υπεκφυγής (Spoofed ή Decoy IPs), που χρησιμοποιούνται κατά κόρον από τους εισβολείς. Ο εκ παραδρομής αποκλεισμός μιας γνωστής συσκευής δικτύου όπως το Router, λόγω ανεπάρκειας αντίστοιχου ελέγχου προέλευσης, μπορεί να επιφέρει ολέθριες συνέπειες στην ασφάλεια των προστατευόμενων δικτύων. Παράλληλα, με την προσθήκη αυξανόμενα μεγάλου όγκου ψεύτικων διευθύνσεων IP στους κανόνες απόρριψης ενός κεντρικοποιημένου Firewall, θα προκύψουν άμεσα ζητήματα συντήρησης, και κυρίως έλλειψης υπολογιστικών πόρων, όσο το σύστημα κλιμακώνεται.

Οι συγγραφείς του [37] εστιάζουν στη ρύθμιση του Snort με σκοπό να λειτουργεί σε Inline Mode, δημιουργώντας έτσι ένα Network Based Intrusion Prevention System. Εξετάζουν διάφορες τεχνικές, όπως DPDK, PF_RING και χρήση εξειδικευμένων καρτών δικτύου (Napatech smartNICs) με σκοπό τη βελτίωση της απόδοσης στην επεξεργασία δικτυακής κυκλοφορίας γραμμής (line-rate). Για τις δοκιμές που πραγματοποιήθηκαν συμπεριλήφθηκε και η χρήση του IXIA BreakingPoint System για τη δημιουργία κυκλοφορίας και την προσομοίωση κακόβουλων FTP εντολών. Το Snort IPS σημείωσε ικανοποιητικά ποσοστά ανίχνευσης και απόρριψης των κακόβουλων πακέτων, ωστόσο εμφάνισε αρκετούς περιορισμούς στην απόδοσή του, όταν ο αριθμός και πολυπλοκότητα των κανόνων αυξήθηκε. Επιπροσθέτως, εφόσον τα συστήματα Inline τοποθετούνται συνήθως σε κρίσιμα σημεία της δικτυακής περιμέτρου, όπως πίσω από το Firewall ή στο Gateway, όπου και παρακολουθούν την εισερχόμενη και την εξερχόμενη κυκλοφορία, υστερούν στην ανίχνευση και την αντιμετώπιση ενδογενών απειλών, όπως αυτές μελετήθηκαν εκτενώς στα παραπάνω Κεφάλαια.

Στο άρθρο [38] περιγράφεται η υλοποίηση ενός συστήματος ανίχνευσης και πρόληψης εισβολών (IDS/IPS) προοριζόμενο κυρίως για εγκατάσταση σε μικρές επιχειρήσεις. Βασίζεται στο Snort, το οποίο ενσωματώνεται σε κατάλληλη συσκευή Raspberry Pi. Το σύστημα ασφάλειας λειτουργεί ως Network IDS/IPS καταγράφοντας κακόβουλες δραστηριότητες και αποκλείοντας τις ύποπτες διευθύνσεις. Στους υφιστάμενους ελέγχους, περιλαμβάνονται προσομοιώσεις επιθέσεων όπως σαρώσεις θυρών, με το σύστημα να ανιχνεύει και να καταγράφει όλες τις ύποπτες διαδικασίες. Μπορεί να αποτελέσει μια λύση χαμηλού κόστους για μικρές επιχειρήσεις, καθώς η χρήση του Raspberry Pi μειώνει σημαντικά τα έξοδα υλοποίησης και λειτουργίας. Επιπλέον, προσφέρει τη δυνατότητα προσαρμογής στις ανάγκες του εκάστοτε δικτύου μέσω του ευέλικτου λογισμικού Snort το οποίο ρυθμίζεται με εξατομικευμένους κανόνες. Ωστόσο, στη μελέτη δεν προτείνεται συγκεκριμένος αυτοματοποιημένος μηχανισμός αποκλεισμού των κακόβουλων διευθύνσεων, ή κάποιος έλεγχος για Evasion Techniques, και αντ' αυτού αφήνεται στην διαμόρφωση του εκάστοτε διαχειριστή.

Τέλος, στο [39] αναλύεται μια μέθοδος ενσωμάτωσης του Snort με το IPSec, για τη δημιουργία ενός Host Based Intrusion Prevention System. Το σύστημα ασφάλειας χρησιμοποιεί τις δυνατότητες του IPSec για να επεξεργάζεται δυναμικά κακόβουλα πακέτα, δημιουργώντας και τροποποιώντας ενεργά τα φίλτρα κυκλοφορίας του, μέσω της συνεργασίας με το Snort. Η μέθοδος αυτή είναι ιδιαίτερα αποτελεσματική στον ταχύ αποκλεισμό των πακέτων, καθώς δεν απαιτούνται πρόσθετες διεργασίες στον πυρήνα του εκάστοτε συστήματος. Μοναδικός περιορισμός της μελέτης αποτελεί το γεγονός ότι το HB - IDS/IPS εφαρμόζεται αποκλειστικά σε Windows OS.

4.2 Μελέτες Μεθόδων Ανίχνευσης και Αντιμετώπισης Συγκεκριμένων Επιθέσεων

Στο [40] παρουσιάζεται μια βελτιωμένη μέθοδος ανίχνευσης σαρώσεων Nmap, μέσω του συνόλου κανόνων Comprehensive Nmap Detection Rules (CNDR). Η μεθοδολογία βασίζεται στον εντοπισμό των ιδιαίτερων χαρακτηριστικών που παρατηρούνται στα συνηθισμένα πακέτα του σαρωτή, όπως τα ύποπτα Packet Headers και συνδυασμοί Flags των εκάστοτε πρωτοκόλλων. Οι κανόνες CNDR σχεδιάστηκαν για να εντοπίζουν τόσο τα τυπικά Scans όσο και τις τεχνικές παράκαμψης IDS, όπως η τροποποίηση και ο τεμαχισμός πακέτων, καθώς οι και καθυστερήσεις στην αποστολή τους. Σημειώνει υψηλό ποσοστό ορθών ανιχνεύσεων σε πληθώρα συνθηκών κυκλοφορίας όμως υστερεί λόγω της στατικής φύσης των κανόνων. Εάν ο επιτιθέμενος ρυθμίσει κατάλληλα τον πηγαίο κώδικα του Nmap, δημιουργώντας πακέτα σάρωσης που αποκλίνουν από τις προσδοκίες των κανόνων, τότε είναι σε θέση να παρακάμψει εύκολα το εν λόγω σύστημα. Η λύση στο πρόβλημα δίνεται από το μοντέλο διαδοχικών νευρωνικών δικτύων που μελετάται στο [41] με σκοπό την ανίχνευση σαρώσεων σε θύρες TCP. Συγκεκριμένα, τα δεδομένα κυκλοφορίας αναλύονται διαδοχικά με τη χρήση δύο μονάδων. Η πρώτη διαχωρίζει τα πακέτα σε γενικές κατηγορίες δικτύου (TCP, UDP, IPv4, κ.λπ.), ενώ η δεύτερη εξειδικεύεται στην αναγνώριση και ανάλυση των TCP μηνυμάτων. Ελέγχει τις μη ολοκληρωμένες χειραψίες και τους ανώμαλους συνδυασμούς σημαιών, χωρίς να βασίζεται σε προκαθορισμένα μοτίβα υπογραφών, αποφεύγοντας έτσι την ανάγκη συνεχούς ανανέωσης των κανόνων με νέες, άγνωστες απειλές. Όπως εξηγήθηκε και παραπάνω, το πρόβλημα των νευρωνικών δικτύων ως επαρκή μέσα για τη διατήρηση της ασφάλειας στα δίκτυα, παρατηρείται κυρίως στη δυσκολία ερμηνείας των αποτελεσμάτων που παράγουν σε πραγματικό χρόνο (Black Box Systems).

Στο [42] προτείνεται μέθοδος ανίχνευσης των επιθέσεων ARP Spoofing, μέσω εκτέλεσης διαδοχικών Bash Script στο περιβάλλον του προστατευόμενου χρήστη. Το πρώτο Script αποθηκεύει σε ένα αρχείο καταγραφών τις τρέχουσες τιμές διευθύνσεων IP και MAC των γνωστών δικτυακών συσκευών. Έπειτα, ένα δεύτερο Bash Script εκτελείται συνεχώς στο παρασκήνιο αναμένοντας την ανανέωση του αρχείου, και στη συνέχεια συγκρίνοντας τις τρέχουσες IP με τις αντίστοιχες αποθηκευμένες. Σε περίπτωση που εντοπιστούν παρεκκλίσεις από τις προκαθορισμένες τιμές, εμφανίζεται άμεσα μια ειδοποίηση στον χρήστη. Η παραπάνω μεθοδολογία, παρά το γεγονός ότι είναι αρκετά αποδοτική στην ανίχνευση των επιθέσεων, απαιτεί εκτεταμένους πόρους λόγω της μόνιμης εκτέλεσης του Script. Παρεμφερής δυσκολία κλιμάκωσης εντοπίζεται και στο σύστημα του [43], ωστόσο σε αυτό προτείνεται πρόσθετος μηχανισμός αντιμετώπισης της επίθεσης. Συγκεκριμένα, η ανίχνευση πραγματοποιείται μέσω των Snort Preprocessors όπως και στην παρούσα εργασία. Παράλληλα, στη μονάδα της αντιμετώπισης τροποποιούνται κατάλληλα οι καταγραφές

των IP – MAC της ARP Cache, επαναφέροντας την στην αρχική της κατάσταση. Όπως προαναφέρθηκε, το σύστημα πρέπει να εκτελείται σε κάθε προστατευόμενο Host ξεχωριστά, καθιστώντας το ακατάλληλο για δίκτια μεγαλύτερου εύρους. Τέλος, μια τεχνολογία αποτροπής αυτού του τύπου επιθέσεων η οποία δεν εξαρτάται από στατικές καταγραφές των δεδομένων δικτύου είναι ο μηχανισμός αυθεντικοποίησης S - ARP [44]. Εφαρμόζεται προφανώς σε ARP Frames, πιστοποιώντας τον εκάστοτε αποστολέα. Ο έλεγχος πραγματοποιείται μέσω κατάλληλης ψηφιακής υπογραφής των μηνυμάτων, χρησιμοποιώντας ασύμμετρη κρυπτογραφία, ενώ η αυθεντικότητα της παραπάνω επαληθεύεται από τον παραλήπτη με το δημόσιο κλειδί του αποστολέα. Ωστόσο η λειτουργία του απαιτεί ξεχωριστή αρχή διαχείρισης και συντήρησης (AKD), ενώ πάσχει από προβλήματα συμβατότητας.

Για επιθέσεις τύπου Bute Force ο μηχανισμός ανίχνευσης του [45] χρησιμοποιεί τεχνικές στατιστικής ανάλυσης των DNS ερωτημάτων τύπου PTR (Pointer Resource Records) από τους προστατευόμενους SSH διακομιστές. Η σύνδεση των δύο πρωτοκόλλων έγκειται στη χρήση αντίστροφων DNS ερωτημάτων (PTR Queries) κατά τη διάρκεια συνδέσεων, όπου ο Server αναζητά το Hostname που αντιστοιχεί στην IP διεύθυνση του πελάτη. Σε επιθέσεις τύπου Brute Force, όπου αποστέλλονται μαζικά αιτήματα σύνδεσης, οι SSH Servers στέλνουν πολλαπλά ερωτήματα, προκαλώντας χαρακτηριστικές αιχμές στην κίνηση DNS. Την απουσία συγκεκριμένου μηχανισμού καταστολής καλύπτει η μελέτη [46], αξιοποιώντας τη δυνατότητα των δέντρων αποφάσεων (Decision Trees) να ταξινομούν την κίνηση δικτύου σε κανονική ή κακόβουλη. Τα χαρακτηριστικά κυκλοφορίας που αναλύονται περιλαμβάνουν μοτίβα συνδέσεων καθώς και ρυθμό αποτυχημένων προσπαθειών. Με την ανίχνευση μιας επίθεσης, το σύστημα μπορεί να αποκλείσει την κακόβουλη κίνηση, να ειδοποιήσει διαχειριστές ή να εφαρμόσει περιορισμούς στο ρυθμό συνδέσεων (Rate Limiting). Η ίδια φιλοσοφία αναγνώρισης επιθέσεων αλλά και αποκλεισμού των κακόβουλων διευθύνσεων αναλύεται και στην παρούσα εργασία, βασιζόμενη όμως σε στατικό μοντέλο υπογραφών.

Οι συγγραφείς του [47] υλοποιούν σύστημα ανίχνευσης επιθέσεων Denial of Service μέσω τεχνικών Data Mining. Το μοντέλο βασίζεται στην εξαγωγή και ανάλυση προτύπων από μεγάλα σύνολα δεδομένων, με σκοπό τον εντοπισμό ύποπτης δικτυακής δραστηριότητας. Η αναγνώριση των διαφόρων μοτίβων επιθέσεων πραγματοποιήθηκε με τη χρήση του αλγορίθμου Apriori ενώ από τον μηχανισμό IP Traceback προκύπτουν οι IP προέλευσης. Ωστόσο, οι λειτουργίες του συστήματος σταματούν σε πρώιμο σημείο της επίθεσης , χωρίς να αξιοποιούνται ουσιαστικά οι ανακτημένες διευθύνσεις αποστολέα. Αντί αυτού στο [48] προτείνεται μια καινοτόμα προσέγγιση στον μετριασμό των επιθέσεων TCP - SYN Flood χρησιμοποιώντας μια συνδυαστική μέθοδο που περιλαμβάνει ανίχνευση, ταξινόμηση και άμυνα. Η βασική φιλοσοφία του IDS – IPS ταυτίζεται εν πολλοίς με αυτή της συγκεκριμένης εργασίας, στα πλαίσια της κατηγοριοποίησης

των εντοπισμένων IP σε Spoofed ή μη διευθύνσεις, και εν τέλει στον αποκλεισμό τους. Στην παραπάνω μελέτη, οι ύποπτες IP εισάγονται σε έναν Naive Bayes ταξινομητή, ο οποίος χρησιμοποιεί χαρακτηριστικά όπως ο αριθμός συνδέσεων, η γεωγραφική τοποθεσία και ορισμένα γνωστά κακόβουλα Bots με σκοπό να καθορίσει αν μια διεύθυνση ανήκει σε επιτιθέμενο ή σε νόμιμο χρήστη. Βάσει των αποτελεσμάτων της ταξινόμησης, το σύστημα εφαρμόζει αμυντικές στρατηγικές, όπως φιλτράρισμα IP διευθύνσεων στους δρομολογητές μέσω αυτόματων Scripts που δημιουργούν λίστες πρόσβασης (Access Lists) ή κανόνες στο Iptables. Στο έργο τους, οι συγγραφείς τονίζουν τη σημασία της σωστής ρύθμισης των ορίων ώστε να αποφεύγεται ο αποκλεισμός νόμιμης κυκλοφορίας. Σημειώνεται ότι η μονάδα της ανίχνευσης ελέγχει τις ενεργές συνδέσεις TCP. Εάν ο αριθμός τους υπερβεί ένα προκαθορισμένο όριο, οι σχετικές διευθύνσεις IP αποθηκεύονται σε αρχείο καταγραφής για περαιτέρω ανάλυση.

Οι συγγραφείς του [49] παραθέτουν μια ολοκληρωμένη προσέγγιση για ανίχνευση και αντιμετώπιση επιθέσεων DHCP Spoofing σε περιβάλλον LAN. Η Detection Engine αναπτύσσεται με τη χρήση Python και της βιβλιοθήκης Scapy, με σκοπό την συχνή αποστολή αυτοματοποιημένων DHCP Discover. Μόλις λάβει πίσω τα αντίστοιχα Offer πακέτα, συγκρίνει τις IP και MAC των διακομιστών με τις αποθηκευμένες διευθύνσεις των εξουσιοδοτημένων Hosts. Εάν οι διευθύνσεις κάποιου Server δεν ανήκουν στην εν λόγω λίστα, τότε το σύστημα μεταβαίνει στη μονάδα ενεργής καταστολής της απειλής. Αποστέλλει μεγάλο αριθμό πλαστών μηνυμάτων DHCP Discover μόνο προς τον Rogue Server, με σκοπό να εξαντλήσει την IP Pool του και ουσιαστικά να τον θέσει εκτός λειτουργίας. Ο μηχανισμός ανίχνευσης δεν είναι παθητικός και απαιτεί συνεχή αποστολή των Discover πακέτων, επιβαρύνοντας το εύρος ζώνης δικτύου. Επιπλέον, αν η ανίχνευση δεν γίνει έγκαιρα, ο Rogue DHCP Server μπορεί να έχει ήδη αναθέσει τις κακόβουλες ρυθμίσεις δικτύου σε θύματα, καθιστώντας αδύνατη την αποτροπή της επίθεσης. Αυτό υπογραμμίζει την ανάγκη για σωστό συγχρονισμό και γρήγορη απόκριση, όπως το σύστημα που προτείνεται στο [50]. Βασίζεται στη χρήση των Managed μεταγωγέων και της λειτουργίας DHCP Snooping την οποία συνήθως διαθέτουν By Default. Κατά την ενεργοποίησή της, δημιουργείται μια βάση δεδομένων που περιλαμβάνει τις IP και MAC διευθύνσεις των Hosts δικτύου, τη διάρκεια της δέσμευσης (Lease Time), καθώς και τη θύρα που χρησιμοποιούν (Trusted ή Untrusted). Αυτή η βάση χρησιμοποιείται για την επαλήθευση των DHCP πακέτων και τη μελλοντική απόρριψη κακόβουλων προσφορών από μη αξιόπιστες θύρες. Η προσέγγιση είναι πλήρης και αποτελεσματική, ωστόσο προϋποθέτει πόρους υλικού και σωστή διαμόρφωση των συσκευών. Σε σενάρια κλιμάκωσης δικτύου, η συνεχής παρακολούθηση και επεξεργασία δεδομένων από τον μεταγωγέα μπορεί να αυξήσει σημαντικά τον φόρτο του, επηρεάζοντας τη σταθερότητά του.

Τέλος, στο έγγραφο [51] περιγράφεται συγκεκριμένη μέθοδος ανίχνευσης αλλά και αντιμετώπισης επιθέσεων ICMP Redirection. Οι χρήστες (Supplicants) είναι σε θέση να εντοπίζουν τη νομιμότητα των πακέτων ανακατεύθυνσης, ελέγχοντας την αντιστοιχία μεταξύ της IP διεύθυνσης πηγής και αντίστοιχης MAC στο μήνυμα. Η ανάλυση δικτυακής κυκλοφορίας ακολουθεί τα πρότυπα Cross – Layer Consistency, εξετάζοντας περεταίρω τα ύποπτα πακέτα Redirection, εάν αυτό που πυροδότησε την κακόβουλη διαδικασία, είναι τύπου UDP. Χρησιμοποιείται τμήμα του Header του (πχ. Source – Destination Port, Checksum) και συγκρίνεται με τις προσδοκώμενες τιμές. Εάν οι πληροφορίες είναι ασυνεπείς, το μήνυμα θεωρείται κακόβουλο. Προχωρώντας στη μονάδα αντιμετώπισης, προτείνεται μια πληθώρα λύσεων, όπως Access Control Lists (ACLs) σε Routers και Switches, και προσωπικά Firewall χρηστών. Ωστόσο, το πεδίο εφαρμογής τους περιορίζεται αποκλειστικά σε δίκτυα Wi-Fi, γεγονός που περιορίζει την ευελιξία και τη χρησιμότητά του σε άλλους τύπους δικτύων, όπως ενσύρματα LAN ή VLAN. Καθολική λύση στην αντιμετώπιση των επιθέσεων σε συστήματα Linux δίνεται στο [51], όπου προτείνεται απενεργοποίηση αποδοχής πακέτων ICMP Redirect από οποιαδήποτε διεύθυνση αποστολέα, ρυθμίζοντας κατάλληλα το Linux Kernel. Έτσι, εξασφαλίζεται η προστασία από επιθέσεις που εκμεταλλεύονται αυτές τις λειτουργίες (net.ipv4.conf.all.accept_redirects=0). Μεταξύ άλλων, η τεχνική αυτή χρησιμοποιείται στους αντίστοιχους μηχανισμούς αντιμετώπισης της παρούσας μελέτης.

5

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΟΠΤΙΚΕΣ

5.1 Περιορισμοί και Συμπεράσματα της Εργασίας

Η παρούσα εργασία είχε ως κύριο στόχο τη σχεδίαση και ανάπτυξη ενός ολοκληρωμένου συστήματος Ανίχνευσης και Αντιμετώπισης Επιθέσεων (IDS/IPS) εντός των εικονικών τοπικών δικτύων (VLAN), χρησιμοποιώντας υποδομές περιορισμένων υπολογιστικών πόρων. Παρά την επιτυχημένη ολοκλήρωση του στόχου αυτού, υπάρχουν ορισμένοι περιορισμοί που είναι σημαντικό να επισημανθούν, ώστε να κατανοηθεί το αντικειμενικό εύρος εφαρμογής και οι δυνατότητές του. Αρχικά, το σύστημα αξιολογήθηκε σε περιβάλλον προσομοίωσης χρησιμοποιώντας ένα Laptop περιορισμένης υπολογιστικής ισχύος. Παρότι οι επιδόσεις του ήταν ικανοποιητικές για το συγκεκριμένο πειραματικό πλαίσιο, οι χρόνοι απόκρισης ενδέχεται να επηρεάζονται σε δίκτυα με μεγαλύτερο όγκο δεδομένων ή με περισσότερες ταυτόχρονες επιθέσεις. Επίσης, βασίζεται σε προκαθορισμένους κανόνες (Signature Based Detection), οι οποίοι, αν και αποτελεσματικοί για τις επιθέσεις που εξετάστηκαν, ενδέχεται να είναι ανεπαρκείς για την ανίχνευση νέων ή εξελιγμένων μορφών εισβολών. Η ανάγκη συνεχούς ενημέρωσης της βιβλιοθήκης κανόνων αποτελεί περιορισμό που μπορεί να υποβαθμίσει την απόδοση και την ευχρηστία του συστήματος μακροπρόθεσμα. Εν κατακλείδι, το μικρό μέγεθος του πειραματικού VLAN δεν επιτρέπει τη γενίκευση των συμπερασμάτων σε δίκτυα αυξημένου εύρους ή πολυπλοκότητας. Οι δυναμικές της κυκλοφορίας σε πιο σύνθετα περιβάλλοντα ενδέχεται να επηρεάσουν την αποτελεσματικότητα του συστήματος, καθώς αυτό δεν δοκιμάστηκε σε εξεζητημένες αρχιτεκτονικές VLAN.

Παρά τους παραπάνω περιορισμούς, το εργαλείο απέδειξε την αποδοτικότητά του στον εντοπισμό και την αντιμετώπιση επιθέσεων σε πραγματικό χρόνο. Η εργασία αυτή αποτελεί μια βάση για περαιτέρω έρευνα και βελτιώσεις, με σκοπό την παράκαμψη των παραπάνω και την επέκταση της λειτουργικότητας του συστήματος σε ποικίλα περιβάλλοντα και τύπους δικτύων.

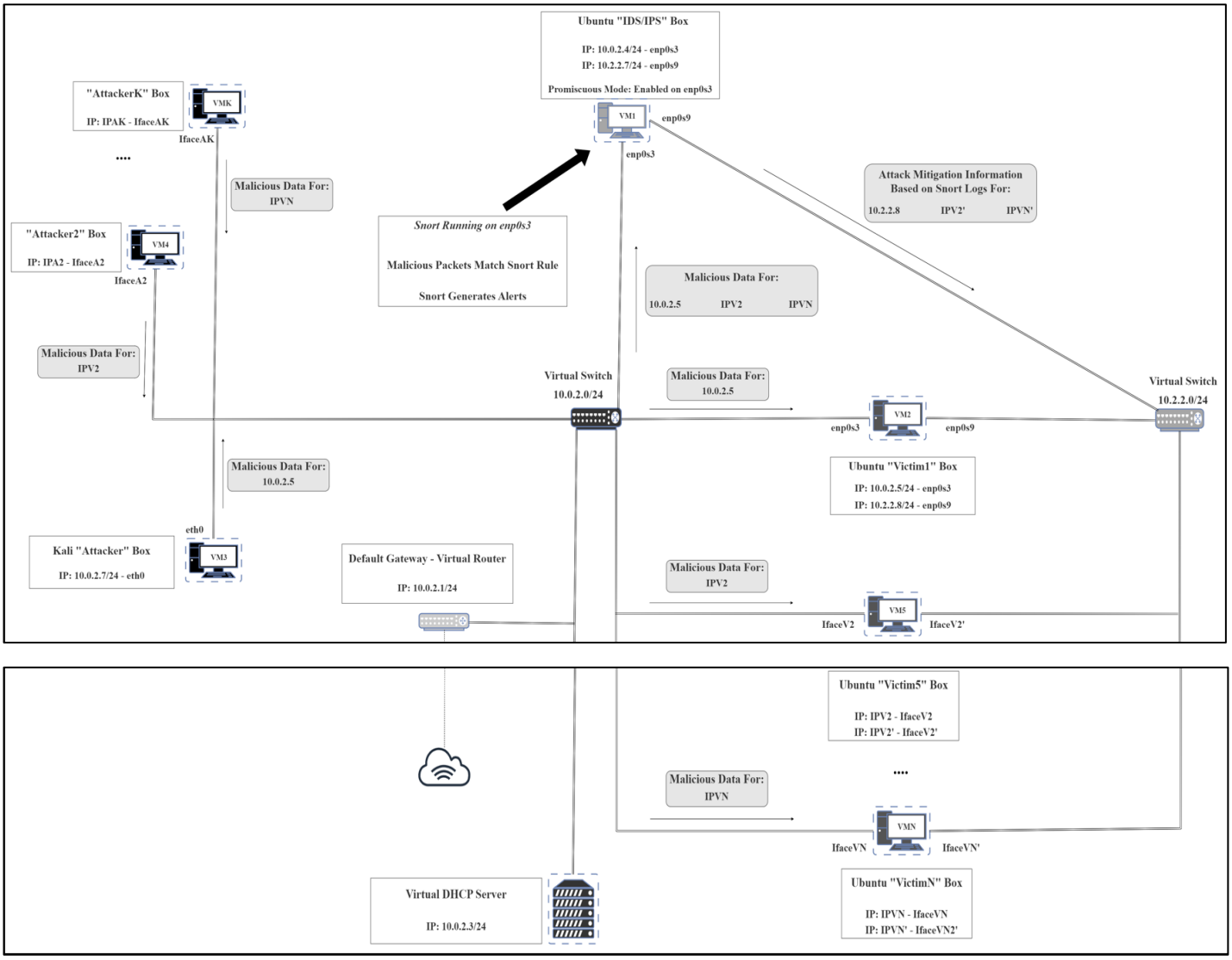
5.2 Μελλοντικές Επεκτάσεις

Οι προτεινόμενες προοπτικές για μελλοντική ανάπτυξη και βελτίωση του συστήματος είναι οι εξής:

1. **Δοκιμή σε μεγαλύτερης κλίμακας προσομοιώσεις**. Όπως έχει εξηγηθεί και παραπάνω το σύστημα, έχει τη δυνατότητα επέκτασης σε μεγαλύτερα δίκτυα λόγω της ευέλικτης και κλιμακωτής (Scalable) αρχιτεκτονικής του. Μια μελλοντική μελέτη θα μπορούσε να επικεντρωθεί στη δοκιμή του σε LAN με περισσότερους επιτιθέμενους (Attackers) αλλά και στόχους (Victims), όπως απεικονίζεται στην τοπολογία της Εικόνας 102. Η καταγραφή λοιπόν της απόδοσής του σε περιβάλλοντα αυξανόμενης πολυπλοκότητα θα μπορούσε να αποτελέσει σημείο αναφοράς για την επέκταση της μελέτης.

2. **Ενσωμάτωση σε Inline Mode**. Τοποθετώντας τη φυσική συσκευή IDS/IPS στην περίμετρο του δικτύου, και ρυθμίζοντας το Snort να λειτουργεί σε Inline Mode, το πεδίο εφαρμογής αναβαθμίζεται σημαντικά, καθώς πλέον το σύστημα λειτουργεί και ως τείχος προστασίας (Firewall). Αυτό θα του επιτρέψει να παρεμβαίνει άμεσα στη ροή της κυκλοφορίας, απορρίπτοντας κακόβουλα πακέτα πριν φτάσουν στον προορισμό τους, αλλά και να ενεργεί άμεσα στο προστατευόμενο σύνολο, εάν ο επιτιθέμενος διαπεράσει το Firewall.

3. **Αύξηση του ρεπερτορίου των επιθέσεων**. Με βάση το μηχανισμό που έχει ήδη υλοποιηθεί όσον αφορά την κατηγοριοποίηση των επιθέσεων από τους αντίστοιχων Handlers το σύστημα μπορεί να επεκτείνει περεταίρω το πεδίο εφαρμογής του. Η προσθήκη νέων κανόνων Snort, καθώς και η ανάλογη κατηγοριοποίηση των επιθέσεων που περιγράφονται στο πλαίσιο της Mainscript.sh. θα έχει ως αποτέλεσμα τη δυναμική κλιμάκωση της λειτουργικότητάς του σε μια ευρύτερη γκάμα σεναρίων εισβολής. Ακόμα, για υποκατηγορίες των ήδη γνωστών τύπων επιθέσεων, προτείνεται η υλοποίηση εμφωλευμένων Handler που θα εμβαθύνουν περεταίρω στα δεδομένα των Snort Logs, χωρίς υπερβολικό Overhead κώδικα.



Εικόνα 102. Τοπολογία Scaled Δικτύων (Χωρισμένη για Λόγους Ευκρίνειας)

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

[1] S. Rohit, "83+ Shocking Cybersecurity Statistics For 2024 (Latest Data)," January 2024. [Online]. Available: https://www.demandsage.com/cybersecurity-statistics/. [Accessed October 2024].

[2] Z. Florian, "The Sectors Most Targeted by Cybercrime," March 2024. [Online]. Available: https://www.statista.com/chart/31985/number-of-cyber-attacks-recorded-per-sector/. [Accessed October 2024].

[3] S. Florin, "The Hidden Impact of Cybercrime on Mental Health," [Online]. Available: https://acronymsolutions.com/resources/the-hidden-impact-of-cyber-attacks-on-mental-health/. [Accessed October 2024].

[4] Sanction Scanner, "Identity Theft Statistics," September 2024. [Online]. Available: https://www.sanctionscanner.com/blog/identity-theft-statistics-587. [Accessed October 2024].

[5] IdentityTheft.org, "2024 Identity Theft Facts and Statistics," 2024. [Online]. Available: https://identitytheft.org/statistics/. [Accessed October 2024].

[6] M. C. -. EPRS, "Understanding cybercrime," March 2024. [Online]. Available: https://www.europarl.europa.eu/RegData/etudes/BRIE/2024/760356/EPRS_BRI(2024)760356_EN.pdf. [Accessed October 2024].

[7] M. Steve, "Cybercrime To Cost The World 8 Trillion Annually In 2023," October 2022. [Online]. Available: https://cybersecurityventures.com/cybercrime-to-cost-the-world-8-trillion-annually-in-2023/. [Accessed October 2024].

[8] K. Matthew, "What is ransomware?," June 2024. [Online]. Available: https://www.ibm.com/topics/ransomware. [Accessed October 2024].

[9] D. W. Camille, "15 cybercrime statistics you ought to know," June 2024. [Online]. Available: https://www.independent.co.uk/advisor/vpn/cybercrime-statistics. [Accessed October 2024].

[10] B. Maine, "What is a VLAN? Ultimate Guide to How VLANs Work," July 2023. [Online]. Available: https://www.esecurityplanet.com/networks/what-is-a-vlan/#logical-partitioning. [Accessed October 2024].

[11] Meraki, "Comparing Layer 3 and Layer 2 Switches," February 2024. [Online]. Available: https://documentation.meraki.com/MS/Layer_3_Switching/Comparing_Layer_3_and_Layer_2_Switches. [Accessed October 2024].

[12] Tashina, "The Role of a Firewall in Network Security," November 2020. [Online]. Available: https://aardwolfsecurity.com/the-role-of-a-firewall-in-network-security/. [Accessed October 2024].

[13] Z. Peter, "Network Segmentation Using Firewalls: Secure Your Data," [Online]. Available: https://firewallfundamentals.com/network-segmentation-using-firewalls/. [Accessed October 2024].

[14] Branka, "Cybercrime Statistics – 2024," February 2024. [Online]. Available: https://truelist.co/blog/cybercrime-statistics/. [Accessed October 2024].

[15] Bace, R. and Mell, P. (2001), Intrusion Detection Systems, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD [Accessed November 2024].

[16] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, 'Intrusion detection system: A comprehensive review', Journal of Network and Computer Applications, vol. 36, no. 1, pp. 16–24, Jan. 2013, doi: 10.1016/j.jnca.2012.09.004.

[17] PaloaltoNetworks, "What is an Intrusion Prevention System?," [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-prevention-system-ips. [Accessed November 2024].

[18] Stiawan, Deris & Yaseen, Ala & Idris, Yazid & Bakar, Kamarulnizam & Abdullah, Hanan. (2012). Intrusion prevention system: A survey. Journal of Theoretical and Applied Information Technology. 7. 44-54.

[19] "Raspberry Pi 4," [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. [Accessed November 2024].

[20] Cisco Systems, Inc.,"Cisco Small Business 300 Series Managed Switches," [Online]. Available: https://www.cisco.com/c/en/us/support/switches/small-business-300-series-managed switches/series.html#~tab-extra. [Accessed November 2024].

[21] S. A. R. Shah, B. Issac, and S. M. Jacob, 'Intelligent Intrusion Detection System Through Combined and Optimized Machine Learning', Int. J. Comp. Intel. Appl., vol. 17, no. 02, p. 1850007, Jun. 2018, doi: 10.1142/S1469026818500074.

[22] M. Roesch, 'Snort – Lightweight Intrusion Detection for Networks', Proceedings of LISA '99: 13th Systems Administration Conference Seattle, Washington, USA, November 7–12, 1999

[23] © 2005 - 2022 Nmap Sofware LLC, "Nmap Reference Guide," [Online]. Available: https://nmap.org/book/man.html. [Accessed November 2024].

[24] "Challenge Response Authentication Mechanism (CRAM)," March 2022. [Online]. Available: https://www.geeksforgeeks.org/challenge-response-authentication-mechanism-cram/. [Accessed November 2024].

[25] "What is DNS hijacking?,". [Online]. Available: https://www.kaspersky.com/resource-center/definitions/what-is-dns-hijacking. [Accessed November 2024].

[26] Ross Heintzkill, "What is a DHCP Starvation Attack?," [Online]. Available: https://www.cbtnuggets.com/blog/technology/networking/what-is-a-dhcp-starvation-attack. [Accessed November 2024].

[27] Oracle, "Oracle VirtualBox: User Guide for Release 7.1," [Online]. Available: https://www.virtualbox.org/manual/. [Accessed November 2024].

[28] OpenSSH, " OpenSSH Manual Pages," [Online]. Available: https://www.openssh.com/manual.html. [Accessed November 2024].

[29] "Bash Scripting – Introduction to Bash and Bash Scripting," February 2024. [Online]. Available: https://www.geeksforgeeks.org/bash-scripting-introduction-to-bash-and-bash-scripting/#applications-of-bash-scripts. [Accessed November 2024].

[30] Cisco and/or its affiliates, " The Snort Project, SNORT® users manual, version 2.9.16," 2020. [Online]. Available: http://manual-snort-org.s3-website-us-east-1.amazonaws.com/. [Accessed November 2024].

[31] "Snort Intrusion Prevention System," [Online]. Available: https://www.snort.org.[Accessed November 2024].

[32] NetworkLessons, " DHCP - Understanding the Magic Cookie," [Online]. Available: https://notes.networklessons.com/dhcp-understanding-the-magic-cookie. [Accessed November 2024].

[33] S. A. R. Shah, B. Issac, and S. M. Jacob, 'Intelligent Intrusion Detection System Through Combined and Optimized Machine Learning', Int. J. Comp. Intel. Appl., vol. 17, no. 02, p. 1850007, Jun. 2018, doi: 10.1142/S1469026818500074.

[34] Y. Peng and H. Wang, 'Design and implementation of network instruction detection system based on snort and NTOP', in 2012 International Conference on Systems and Informatics (ICSAI2012), Yantai, China: IEEE, May 2012, pp. 116–120. doi: 10.1109/ICSAI.2012.6223247.

[35] T. Widodo and A. S. Aji, 'Implementation of Intrusion Detection System (IDS) and Snort Community Rules to Detect Types of Network Attacks', IJCA, vol. 183, no. 42, pp. 30–35, Dec. 2021, doi: 10.5120/ijca2021921821.

[36] J. Xi, 'A Design and Implement of IPS Based on Snort', in 2011 Seventh International Conference on Computational Intelligence and Security, Sanya, Hainan, China: IEEE, Dec. 2011, pp. 771–773. doi: 10.1109/CIS.2011.175.

[37] M. Sheeraz, M. Hanif Durad, S. Tahir, H. Tahir, S. Saeed, and A. M. Almuhaideb, 'Advancing Snort IPS to Achieve Line Rate Traffic Processing for Effective Network Security Monitoring', IEEE Access, vol. 12, pp. 61848–61859, 2024, doi: 10.1109/ACCESS.2024.3395123.

[38] J. E. C. De La Cruz, C. A. R. Goyzueta, and C. D. Cahuana, 'Intrusion Detection and Prevention System for Production Supervision in Small Businesses Based on Raspberry Pi and Snort', in 2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON), Lima, Peru: IEEE, Sep. 2020, pp. 1–4. doi: 10.1109/INTERCON50315.2020.9220240.

[39] Jiqiang Zhai and Yining Xie, 'Researh on Network Intrusion Prevention System Based on Snort', in Proceedings of 2011 6th International Forum on Strategic Technology, Harbin, Heilongjiang, China: IEEE, Aug. 2011, pp. 1133–1136. doi: 10.1109/IFOST.2011.6021220.

[40] S. Liao et al., 'A Comprehensive Detection Approach of Nmap: Principles, Rules and Experiments', in 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Chongqing, China: IEEE, Oct. 2020, pp. 64–71. doi: 10.1109/CyberC49757.2020.00020.

[41] B. Hartpence and A. Kwasinski, ‘Combating TCP Port Scan Attacks Using Sequential Neural Networks’, in 2020 International Conference on Computing, Networking and Communications (ICNC), Big Island, HI, USA: IEEE, Feb. 2020, pp. 256–260. doi: 10.1109/ICNC47757.2020.9049730.

[42] A. A. M. M. Amin and M. S. Mahamud, ‘An Alternative Approach of Mitigating ARP Based Man-in-the-Middle Attack Using Client Site Bash Script’, in 2019 6th International Conference on Electrical and Electronics Engineering (ICEEE), Istanbul, Turkey: IEEE, Apr. 2019, pp. 112–115. doi: 10.1109/ICEEE2019.2019.00029.

[43] X. Hou, Z. Jiang, and X. Tian, ‘The detection and prevention for ARP Spoofing based on Snort’, in 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), Taiyuan, China: IEEE, Oct. 2010, pp. V5-137-V5-139. doi: 10.1109/ICCASM.2010.5619113.

[44] D. Bruschi, A. Ornaghi, and E. Rosti, ‘S-ARP: a secure address resolution protocol’, in 19th Annual Computer Security Applications Conference, 2003. Proceedings., Las Vegas, Nevada, USA: IEEE, 2003, pp. 66–74. doi: 10.1109/CSAC.2003.1254311.

[45] M. Kumagai, Y. Musashi, D. A. L. Romana, K. Takemori, S. Kubota, and K. Sugitani, ‘SSH Dictionary Attack and DNS Reverse Resolution Traffic in Campus Network’, in 2010 Third International Conference on Intelligent Networks and Intelligent Systems, Shenyang, China: IEEE, Nov. 2010, pp. 645–648. doi: 10.1109/ICINIS.2010.9.

[46] Ibrahim, Nebras. (2023). IMPROVING THE EFFICIENCY OF BRUTE-FORCE ATTACK DETECTION USING DECISION TREES: AN ANALYSIS STUDY. 2. 2181-4384. Available: https://www.researchgate.net/publication/375834787. [Accessed December 2024].

[47] K. Garg and R. Chawla, ‘DETECTION OF DDOS ATTACKS USING DATA MINING’, International Journal of Computing and Business Research, vol. 2, no. 1, 2011. Available: https://researchmanuscripts.com/PapersVol2N1Jan2011/3.pdf. [Accessed December 2024].

[48] M. Bogdanoski, A. Toshevski, D. Bogatinov, and M. Bogdanoski, ‘A novel approach for mitigating the effects of the TCP SYN flood DDoS attacks’. Available: https://www.researchgate.net/publication/375431794. [Accessed December 2024].

[49] Jony, A., & Islam, M. N. (2023). An effective technique to automatically detect and neutralize rogue DHCP server. 2023 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD), 244–248. https://doi.org/10.1109/ICICT4SD59951.2023.10303033

[50] H. A. S. Adjei, M. T. Shunhua, G. K. Agordzo, Y. Li, G. Peprah and E. S. A. Gyarteng, "SSL Stripping Technique (DHCP Snooping and ARP Spoofing Inspection)," 2021 23rd International Conference on Advanced Communication Technology (ICACT), PyeongChang, Korea (South), 2021, pp. 187-193, doi: 10.23919/ICACT51234.2021.9370460.

[51] X. Feng, Q. Li, K. Sun, Y. Yang, and K. Xu, ‘Man-in-the-Middle Attacks without Rogue AP: When WPAs Meet ICMP Redirects’, in 2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA: IEEE, May 2023, pp. 3162–3177. doi: 10.1109/SP46215.2023.10179441.

[52] Andrew Ayer, "ICMP Redirect Attacks in the Wild," July 2013. [Online]. Available: https://www.agwa.name/blog/post/icmp_redirect_attacks_in_the_wild . [Accessed December 2024].

[53] M. Roesch, ‘Snort – Lightweight Intrusion Detection for Networks’, 1999. Available: https://www.usenix.org/legacy/event/lisa99/full_papers/roesch/roesch.pdf. [Accessed December 2024].

ΠΑΡΑΡΤΗΜΑ ΚΩΔΙΚΑ

Ο πηγαίος κώδικας του συστήματος IDS/IPS που αναπτύχθηκε στο πλαίσιο της παρούσας εργασίας, καθώς και οι βασικές παραμετροποιήσεις του δικτύου στην αντίστοιχη Εικονική Μηχανή (VM), είναι διαθέσιμα στο αποθετήριο του έργου στο GitHub. Μπορείτε να αποκτήσετε πρόσβαση στις λεπτομέρειες υλοποίησης και στη συνολική τεκμηρίωση του Project μέσω του ακόλουθου συνδέσμου.

https://github.com/ManosKaramitros/Thesis_IDS_IPS/tree/main