

ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



IONIAN
UNIVERSITY

-- Πτυχιακή Εργασία --

Εκπαίδευση Μοντέλων Τεχνητής Νοημοσύνης με Χρήση
Ανοικτών Δεδομένων Δικτυακής Κίνησης

Ελευθέριος Πιπιλακίδης
Επιβλέπων: – Εμμανουήλ Μάγκος –

September 28, 2024

Επιβλέπων Καθηγητής

Εμμανουήλ Μάγκος, Καθηγητής,
Ιόνιο Πανεπιστήμιο

Τριμελής Επιτροπή

Εμμανουήλ Μάγκος, Καθηγητής,
Ιόνιο Πανεπιστήμιο

Χριστόφορος Νταντογιάν, Επίκουρος Καθηγητής,
Ιόνιο Πανεπιστήμιο

Αγγελική Τσώχου, Αναπληρώτρια Καθηγήτρια,
Ιόνιο Πανεπιστήμιο

Περίληψη

Στην παρούσα διατριβή έχουμε την ανάπτυξη ενός καινοτόμου Intrusion Detection System (IDS), αξιοποιώντας τις σύγχρονες μεθοδολογίες του Machine Learning (ML) και της ανάλυσης δικτύων. Η προσέγγιση βασίζεται στη χρήση ανοικτών Packe Capture (PCAP) από τα οποία εξάγονται κρίσιμα στατιστικά χαρακτηριστικά των ροών δικτύου μέσω εργαλείων όπως το Network Flow Stream (NFStream)¹ και το Canadian Institute Cybersecurity Flow Meter (CICFlowMeter)². Τα παραγόμενα δεδομένα αξιοποιούνται για την εκπαίδευση και αξιολόγηση μοντέλων Artificial Intelligence (AI), με στόχο την αποτελεσματική ανίχνευση και πρόβλεψη κακόβουλων επιθέσεων που πλήττουν περιβάλλοντά υπολογιστικών δικτύων.

Αυτή η διαδικασία υλοποιήσεις ξεκινά με την προσεκτική επιλογή των καταλληλότερων συνόλων δεδομένων, τα οποία περιλαμβάνουν πραγματικά και προσημειωμένα αρχεία PCAP, προκειμένου να διασφαλιστεί η ποικιλομορφία των επιθέσεων που αναλύονται. Μετά την επιλογή των δεδομένων, ακολουθεί εκτενής προεπεξεργασία, η οποία αποτελείται από τον καθαρισμό δεδομένων, την αφαίρεση ακραίων τιμών και την κοινωνικοποίησή των χαρακτηριστικών, με στόχο τη διασφάλιση της ακρίβειας και την αξιοπιστία του συνόλου δεδομένων. Παράλληλα, πραγματοποιείται ανάλυση των επιθέσεων όπως Denial of Service (DOS), Distributed Denial of Service (DDoS) και άλλων εξελεγμένων μορφών επιθέσεων που έχουν καταγραφεί στα αρχεία PCAP.

Όσον αφορά το στάδιο της εξαγωγής στατιστικών πληροφοριών, εργαλεία όπως το NFStream, Suricata³ και Zeek⁴ επιτρέπουν την απόσπαση κρίσιμων χαρακτηριστικών, όπως ο αριθμός των πακέτων και ο αριθμός των tcp συνδέσεων, τα οποία αποτελούν θεμελιώδεις

¹<https://github.com/nfstream/nfstream>

²<https://github.com/ahlashkari/CICFlowMeter>

³<https://suricata.io>

⁴<https://zeek.org/>

μεταβλητές για την ανάλυσή ροών δικτύου. Αυτές οι πληροφορίες ενσωματώνονται σε ένα σύνολο δεδομένων, το οποίο χρησιμοποιείται για την εκπαίδευσή των αλγορίθμων μηχανικής μάθησης. Ένα από τα βασικά στοιχεία της παρούσας εργασίας ήταν η λεπτομερής τεκμηρίωση των επιθέσεων, καθώς και η δημιουργία και προσαρμογή νέων επιθέσεων μέσω προσομοιωμένων σεναρίων, με σκοπό την ενίσχυσή των συνόλων δεδομένων.

Στο τελικό στάδιο της εργασίας, τα μοντέλα μηχανικής μάθησης εκπαιδεύτηκαν χρησιμοποιώντας αλγόριθμους όπως Random Forest που έχει αποδεχθεί εξαιρετικά αποτελεσματικός στην ανίχνευση επιθέσεων μέσω των χαρακτηριστικών που διεξάχθηκαν από τα αρχεία PCAP. Στη συνέχεια η αξιολόγησή το μοντέλων έγινε με τη χρήση μετρικών απόδοσής, όπως Accuracy, Recall και η καμπύλη Receiver Operating Characteristic (ROC), όλα αυτά προκειμένου να εκτιμηθεί η ικανότητά ανίχνευσής.

Έτσι η παρούσα διατριβή καταλήγει σε σημαντικά συμπεράσματα σχετικά με την αποτελεσματικότητά των μοντέλων τεχνητής νοημοσύνης στην ανίχνευση επιθέσεων, παρέχοντας ταυτόχρονα πρακτικές προτάσεις για τη βελτίωση της απόδοσης των συστημάτων ανίχνευσης εισβολών. Επιπλέον προτείνονται κατευθύνσεις για μελλοντική έρευνά και εφαρμογές αυτών των τεχνικών σε πραγματικά δίκτυα, ενώ εξετάζεται η δυνατότητα βελτιστοποίησης των μοντέλων με τη χρήση μεθόδων όπως Principal Component Analysis (PCA), για τη μύωση της διάστασης των χαρακτηριστικών, καθώς και η ενσωμάτωσή πιο σύνθετων αλγορίθμων όπως Neural Networks για μεγαλύτερη ακρίβεια και ταχύτητά ανίχνευσης.

Πρόλογος και Ευχαριστίες

Η παρούσα Πτυχιακή εργασία αποτελεί το επιστέγασμα των σπουδών μου στο Τμήμα Πληροφορικής του Ιόνιου Πανεπιστήμιου. Κατά τη διάρκεια των σπουδών αυτών, είχα την ευκαιρία να εμβυθιστώ σε σημαντικά επιστημονικά πεδία και να αποκτήσω πολύτιμες γνώσεις και δεξιότητες, οι οποίες συνέβαλαν καθοριστικά στην προσωπική και επαγγελματική μου εξέλιξη.

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, Εμμανουήλ Μάγκο, για την εμπιστοσύνη που έδειξε στις ικανότητες μου και τη συνεχή υποστήριξη του. Τα πολύτιμα σχόλια και οι συμβουλές του με βοήθησαν να ξεπεράσω τις δυσκολίες και να ολοκληρώσω με επιτυχία την παρούσα διατριβή.

Επιπλέον, θα ήθελα να ευχαριστήσω τα μέλη της τριμελούς εξεταστικής επιτροπής, Καθηγητή κ. Εμμανουήλ Μάγκο, τον Επίκουρο Καθηγητή κ. Χριστόφορο Νταντογιάν καθώς και την Αναπληρώτρια Καθηγήτρια κα. Αγγελική Τσώχου για την καθοδήγηση και τις εποικοδομητικές υποδείξεις τους.

Φυσικά, Θα ήθελα να ευχαριστήσω τον Μεταδιδάκτορα δρ. Στυλιανό Καραγιάννη για την τεχνική υποστήριξη και την πολύτιμη βοήθεια του που έπαιξε καθοριστικό ρόλο για την παρούσα πτυχιακή εργασία.

Ιδιαίτερη αναφορά θα πρέπει να γίνει στην οικογένεια μου, η οποία με στήριξε συνεχώς σε κάθε στάδιο των σπουδών μου, καθώς και στους φίλους μου και συμφοιτητές μου, οι οποίοι αποτέλεσαν πηγή ενθάρρυνσης και έμπνευσης καθ' όλη τη διάρκεια των σπουδών μου.

Με ειλικρινή ευγνωμοσύνη,
Ελευθέριος Πιπιλαχίδης

Περιεχόμενα

A	Εισαγωγή	1
A.1	ΠΑΡΟΜΟΙΕΣ ΕΡΓΑΣΙΕΣ	4
A.2	ΣΥΝΕΙΣΦΟΡΑ	6
A.3	ΕΝΔΕΙΚΤΙΚΗ ΔΟΜΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΑ ΠΑΡΑΚΑΤΩ ΒΑΣΙΚΑ ΚΕΦΑΛΑΙΑ ΚΑΙ ΕΝΟΤΗΤΕΣ:	7
B	Μεθοδολογία	9
C	Κακόβουλες Επιθέσεις και οι βασικοί αλγόριθμοι τους	44
D	Προεπεξεργασία Δεδομένων	50
E	Ανίχνευση Εισβολών μέσω Ανάλυσης Δικτυακής Κίνησης	66
F	Συμπεράσματα και Μελλοντικές Προτάσεις	75
	Συντμήσεις	83
	Γλωσσάρι Ξενικών Όρων	86

Κατάλογος Σχημάτων

B.1	Enter Caption	12
B.2	Δημιουργία του κατάλληλου dataset(.csv) από το eve.json	14
B.3	Δημιουργία του κατάλληλου dataset(.csv) από το zeel logs	15
B.4	Συγχώνευση των suricat.csv, zeek.csv, NFStream.csv σε ένα merged.csv. . .	16
B.5	Ο Neural Network εκπαιδεύετε με τα merge.csv και παράγει το μοντέλο NeuralNetwork.h5	18
B.6	Ο Random Forest εκπαιδεύετε με τα merge.csv και παράγει το μοντέλο RandomForst.pkl	19
B.7	Ο Support Vector Machine εκπαιδεύετε με τα merge.csv	20
B.8	Αξιολόγηση και βελτιστοποίηση μοντέλων	21
B.9	Live Capture IDS:FIRST	22
B.10	Live Capture IDS:SECOND	23
B.11	Zeek Logs output	25
B.12	Εκτέλεση Zeek για ανάλυση PCAP αρχείων και αποθήκευση logs.	28
B.13	Suricata Logs output	29
B.14	Ανάλυση καταγραφών Suricata από το αρχείο eve.json.	31
B.15	Εκτέλεση Suricata για ανάλυση PCAP αρχείων και αποθήκευση logs.	34
B.16	Wireshark ανάλυση πακέτων, προβολή δικτυακής δραστηριότητας.	38
B.17	Εκπαίδευση Random Forest, πρόβλεψη και αξιολόγηση ακρίβειας.	40
B.18	Εκπαίδευση SVM, πρόβλεψη και αξιολόγηση ακρίβειας.	41
B.19	Εκπαίδευση MLP νευρωνικού, αξιολόγηση με ακρίβεια.	42
C.20	Εκτέλεση DoS επίθεσης με Hping3 σε συγκεκριμένη διεύθυνση IP και πόρτα. .	45
C.21	Εκτέλεση SQL Injection επίθεσης με SQLMap σε ευπαθή ιστοσελίδα.	46
C.22	Εκτέλεση XSS επίθεσης με XSSer σε ευπαθή ιστοσελίδα.	47

C.23 Εκτέλεση Brute Force επίθεσης με Hydra σε SSH.	48
C.24 Εκτέλεση port scanning με Nmap σε συγκεκριμένη διεύθυνση IP.	49
E.25 Ανάλυση αρχείου PCAP με χρήση NFStream και εξαγωγή χαρακτηριστικών ροής δικτύου.	68
E.26 Πίνακας δικτυακών ροών με στοιχεία όπως IP πηγής, IP προορισμού, πρωτόκολλο, θύρες, συνολικά πακέτα, συνολικά bytes και διάρκεια ροής σε milliseconds. . .	69

Κατάλογος Πινάκων

<i>E.1</i>	<i>Απόδοση των μοντέλων σε μετρικές αξιολόγησης</i>	<i>72</i>
<i>E.2</i>	<i>Ανάλυση Ψευδώς Θετικών και Ψευδώς Αρνητικών Αποτελεσμάτων</i>	<i>73</i>

Κεφάλαιο Α

Εισαγωγή

Η αλματώδης ανάπτυξη της τεχνολογίας των υπολογιστών και των τηλεπικοινωνιών τα τελευταία χρόνια έχει αλλάξει ριζικά τον τρόπο με τον οποίο οι κοινωνίες και οι επιχειρήσεις λειτουργούν. Η ευρεία χρήση του διαδικτύου και των δικτυακών υπηρεσιών έχει προσφέρει νέες δυνατότητες για επικοινωνία, επεξεργασία δεδομένων και παροχή υπηρεσιών, αλλά ταυτόχρονα έχει δημιουργήσει σημαντικούς κινδύνους όσον αφορά την ασφάλεια των Πληροφοριών. Οι κυβερνοεπιθέσεις αυξάνονται με γεωμετρική πρόοδο, με επιθέσεις που στοχεύουν από μεμονωμένους χρήστες και μικρές επιτηρήσεις, έως μεγάλα δίκτυα πολυεθνικών εταιρειών και κυβερνητικών υπηρεσιών.[4] [7] [8]

Στο σύγχρονο περιβάλλον, η ασφάλεια των δικτύων αποτελεί θεμελιώδεις λίθος για τη διατήρηση της εμπιστευτικότητας, της ακεραιότητας και της διαθεσιμότητας των πληροφοριών. Οι κυβερνοεγκληματίες ωστόσο έχουν εξελιχθεί χρησιμοποιώντας όλο και πιο σύνθετες μεθόδους και τεχνικές για να παραβιάσουν τα δίκτυα και να αποκτήσουν πρόσβασή σε ευαίσθητα δεδομένα. Στο πλαίσιο αυτό τα Intrusion Detection Systems έχουν αναδειχθεί ως μια από τις βασικές λύσεις για την ανίχνευση και την αποτροπή τέτοιων επιθέσεων. Τα IDS είναι συστήματα που παρακολουθούν τη δραστηριότητα σε ένα δίκτυο ή σύστημα για ύποπτες ενέργειες και αποσκοπούν στην έγκαιρη αναγνώριση και ανίχνευση πιθανών εισβολών.[30] [14]

Τα παραδοσιακά IDS βασίζονται σε υπογραφές(signatures) επιθέσεων ή κανόνες που έχουν οριστεί εκ των προτέρων, προκειμένου να εντοπίζουν συγκεκριμένες μορφές κακόβουλης δραστηριότητας. Αν και αυτά τα συστήματά είναι αποτελεσματικά για την ανίχνευση γνω-

στών επιθέσεων άλλα δεν έχουν τόσες δυνατότητές ενάντια σε νέες, άγνωστες ή μεταλλαγμένες επίθεσις καθώς δε διαθέτουν την ικανότητά να προσαρμόζονται σε νέα δεδομένα και απειλές. Για να αντιμετωπιστούν οι αδυναμίες αυτές, οι ερευνητές και οι επαγγελματίες της ασφάλειας έχουν στραφεί στη μηχανική μάθησή, η οποία επιτρέπει στα IDS να 'μαθαίνουν' από τα δεδομένα και να αναγνωρίζουν πρότυπα επιθέσεων χωρίς την ανάγκη για προκαθορισμένες υπογραφές.[26] [10]

Τα παραδοσιακά συστήματα ανίχνευσης εισβολών (IDS) βασίζονται κυρίως σε τεχνικές που βασίζονται σε υπογραφές, οι οποίες απαιτούν τη δημιουργία και συνεχή ενημέρωση υπογραφών για κάθε γνωστή απειλή. Αυτό τα καθιστά αναποτελεσματικά έναντι επιθέσεων μηδενικής ημέρας, όπου δεν υπάρχουν προκαθορισμένα πρότυπα για την ανίχνευση. Επιπλέον, η συνεχής εξάρτηση από υπογραφές σημαίνει ότι τα συστήματα αυτά ενδέχεται να μην ανιχνεύουν νέες ή προηγμένες τεχνικές επιθέσεων που δεν έχουν ακόμη εντοπιστεί. Σε αυτό το πλαίσιο, η μηχανική μάθηση (ML) μπορεί να προσφέρει σημαντικά πλεονεκτήματα, επιτρέποντας στα IDS να μαθαίνουν από ανοικτά σύνολα δεδομένων και να εντοπίζουν ανωμαλίες χωρίς να εξαρτώνται από υπογραφές. Η παρούσα έρευνα αποσκοπεί στη διερεύνηση του κατά πόσον τα ML-IDS μπορούν να ενσωματώσουν πιο αποτελεσματικές τεχνικές, να προσαρμοστούν σε επιθέσεις μηδενικής ημέρας και να αξιοποιήσουν τα διαθέσιμα ανοικτά σύνολα δεδομένων για την ανάπτυξη μοντέλων που μπορούν να εφαρμοστούν σε περιβάλλοντα πραγματικού κόσμου. Αναλύοντας τα αποτελέσματα και τη συμβατότητα των συνόλων δεδομένων, θα εξαχθούν συμπεράσματα που θα μας επιτρέψουν να εντοπίσουμε τα κρίσιμα διδάγματα της έρευνας και να κατανοήσουμε τη σημασία αυτής της προσέγγισης.[24] [23] [5]

Η πολυπλοκότητα και η συχνότητα των επιθέσεων στον κυβερνοχώρο αυξάνονται ραγδαία, καθιστώντας την ανίχνευση και την αντιμετώπιση αυτών των απειλών μια από τις μεγαλύτερες προκλήσεις στην ασφάλεια των δικτύων. Σύμφωνα με πρόσφατες μελέτες, οι επιθέσεις Zero Day αποτελούν σοβαρή απειλή, καθώς είναι δύσκολο να εντοπιστούν με τη χρήση παραδοσιακών μεθόδων που βασίζονται σε υπογραφές. Οι Bhuyan κ.ά. (2018) επισημαίνουν ότι οι επιτιθέμενοι υιοθετούν όλο και πιο εξελιγμένες τεχνικές, ενώ οι Jiang κ.ά. (2019) τονίζουν την ανάγκη για συστήματα που μπορούν να προσαρμόζονται γρήγορα σε νέες απειλές και ευπάθειες. Τα παραδοσιακά συστήματα ανίχνευσης εισβολών δεν είναι σε θέση να ανιχνεύσουν άγνωστες επιθέσεις, γεγονός που αναδεικνύει τη σημασία της ενσωμάτωσης τεχνολογιών μηχανικής μάθησης για την ενίσχυση της προστασίας. [4] [11]

Μία από τις κύριες προκλήσεις κατά την ανάπτυξη ενός συστήματος ανίχνευσης εισβολών με βάση τη μηχανική μάθηση είναι η διαχείριση του μεγάλου όγκου δεδομένων που προκύπτουν από τη συνεχή παρακολούθηση των ροών δικτύου. Η συλλογή και η επεξεργασία αυτών των δεδομένων απαιτεί υψηλή υπολογιστική ισχύ και χώρο αποθήκευσης, ενώ η ακρίβεια των μοντέλων μπορεί να επηρεαστεί από ψευδώς θετικά αποτελέσματα, γεγονός που μπορεί να οδηγήσει σε ψευδείς ειδοποιήσεις και αυξημένο διαχειριστικό φόρτο. Επιπλέον, η συνεχής εξέλιξη των τεχνικών επίθεσης απαιτεί τακτική ενημέρωση και επανεκπαίδευση των μοντέλων, προκειμένου να παραμένουν αποτελεσματικά έναντι νέων και άγνωστων απειλών.[13] [28]

Σε αυτή την πτυχιακή εργασία εστιάζουμε στην ανάπτυξη και την αξιολόγησή ενός συστήματος ανίχνευσης εισβολών που βασίζεται σε τεχνικές μηχανικής μάθησης και ανάλυσής ροών δικτύου. Ο στόχος είναι να δημιουργηθεί ένα αυτοματοποιημένο σύστημά που θα μπορεί να αναγνωρίζει ανωμαλίες σε πραγματικό χρόνο και να προσαρμόζεται στις διαρκώς εξελισσόμενες απειλές. Για την εκπαίδευση και αξιολόγηση του συστήματος χρησιμοποιούνται ανοιχτά σύνολα δεδομένων PCAP που περιλαμβάνουν ροές δικτύου και επιθέσεις ποικίλων μορφών. Με τη χρήση εργαλείων όπως NFStream ή το CICFlowMeter, εξάγονται στατιστικά δεδομένα ροών δικτύου, τα οποία αποτελούν τη βάση για την εκπαίδευση μοντέλων μηχανικής μάθησης.[3] [27]

Ενσωμάτωση νέων τεχνολογιών και εργαλείων: Με την ταχεία εξέλιξη των τεχνολογιών στον τομέα της ασφάλειας δικτύων, Τα σύγχρονα IDS ενσωματώνουν νέες προσεγγίσεις, όπως ανάλυση μεγάλων δεδομένων (Big Data Analytics) και αλγόριθμους τεχνίτης νοημοσύνης. Έτσι εξετάζεται η δυνατότητά ενσωμάτωσης εργαλείων όπως το Zeek και το Suricata, τα οποία προσφέρουν λεπτομερή ανάλυση των ροών δικτύου και την παρακολούθηση των δικτυακών δραστηριοτήτων. Η συνεργασία αυτών των εργαλείων με τα μοντέλα ML προσφέρει αυξημένη ακρίβεια στην ανίχνευση και κατηγοριοποίηση των κυβερνοεπιθέσεων, διασφαλίζοντας ότι οι οργανισμοί μπορούν να προστατευτούν από ευρύτερο φάσμα απειλών.[29] [6]

Με την εισαγωγή των ML στα IDS επιτυγχάνεται η δημιουργία αυτοματοποιημένων συστημάτων που έχουν την ικανότητα να προσαρμόζονται σε νέες συνθήκες και απειλές. Αυτό επιτρέπει στα συστήματα να εντοπίζουν άγνωστες επιθέσεις και να προσφέρουν πιο αποδοτική

προστασία, κάτι που είναι ζωτικής σημασίας στο σύγχρονο περιβάλλον ασφάλειάς των πληροφοριών. Έτσι ο στόχος είναι να δημιουργηθούν νέες προοπτικές για τη βελτίωση των IDS με τα μοντέλα ML.[1] [12] [1]

A.1 Παρόμοιες Εργασίες

1. Intrusion Detection System with Machine Learning and Multiple Datasets

(2023): Η εργασία αυτή επικεντρώνεται στην ανάπτυξη ενός συστήματος ανίχνευσης εισβολών (IDS) με χρήση μηχανικής μάθησης και πολλαπλών συνόλων δεδομένων. Χρησιμοποιεί τεχνικές βελτιστοποίησης υπερπαραμέτρων για τη βελτίωση της ακρίβειας και της αποτελεσματικότητας των μοντέλων, όπως το XGBoost και το Random Forest. Τα αποτελέσματα δείχνουν εξαιρετική απόδοση, με ακρίβεια 99,9%. [19]

2. Survey on Intrusion Detection Systems Based on Machine Learning Techniques for the Protection of Critical Infrastructure (2023):

Το έγγραφο αυτό εξετάζει τη χρήση τεχνικών μηχανικής μάθησης για την ανίχνευση επιθέσεων σε κρίσιμες υποδομές. Συζητά τη χρήση δεδομένων από το IoT και τη βιομηχανική ασφάλεια, προσφέροντας μια εκτενή επισκόπηση των τεχνικών και των προκλήσεων για τα IDS σε αυτά τα περιβάλλοντα.[22]

3. Machine Learning-Based Intrusion Detection System for Autonomous Vehicles (2022):

η έρευνα αυτή επικεντρώνεται στην ανίχνευση εισβολών σε αυτόνομα οχήματα με τη χρήση μοντέλων μηχανικής μάθησης, όπως το SVM και το Deep Learning. Αξιολογεί τον αντίκτυπο της ανίχνευσης εισβολών στην ασφάλεια των επικοινωνιών δικτύου εντός του οχήματος.[20]

4. Anomaly-Based Intrusion Detection System for IoT Using Machine Learning (2022):

Σε αυτή την εργασία, η ανίχνευση ανωμαλιών στο IoT γίνεται με τη χρήση τεχνικών μηχανικής μάθησης. Εξετάζει τα μοντέλα Random Forest και Decision Trees, τονίζοντας την ανάγκη για αλγόριθμους χαμηλής κατανάλωσης που μπορούν να εφαρμοστούν σε μικροσυσκευές IoT.[17]

5. A Comprehensive Review on Machine Learning for Intrusion Detection Systems (2020):

Η ανασκόπηση αυτή παρουσιάζει τη χρήση της μηχανικής μάθησης για την ανίχνευση εισβολών, εστιάζοντας σε τρέχουσες προκλήσεις όπως οι επιθέσεις μηδενικής ημέρας και η προστασία κρίσιμων υποδομών. Συζητά την ανάγκη για καλύτερη κατηγοριοποίηση των επιθέσεων και την ενσωμάτωση μεγαλύτερων συνόλων δεδομένων.[15]

6. A Hybrid Intrusion Detection System Combining Machine Learning and Signature-Based Detection (2021):

Η έρευνα αυτή συνδυάζει τεχνικές μηχανικής μάθησης με ανίχνευση βάσει υπογραφών, προτείνοντας ένα υβριδικό μοντέλο για την ανίχνευση τόσο γνωστών όσο και άγνωστων επιθέσεων. Το υβριδικό μοντέλο επιτυγχάνει με ακρίβεια 96% χρησιμοποιώντας έναν συνδυασμό CNN και παραδοσιακών μεθόδων ανίχνευσης.[16]

7. Deep Learning-Based Intrusion Detection System for Industrial Control Systems (2021):

η έρευνα επικεντρώνεται στα βιομηχανικά συστήματα ελέγχου και στον τρόπο με τον οποίο μπορούν να βελτιωθούν τα συστήματα ανίχνευσης εισβολών μέσω τεχνικών βαθιάς μάθησης. Αξιοποίηση προβλεπτικών μοντέλων όπως το LSTM για την ανίχνευση ανωμαλιών σε συστήματα SCADA.[18]

8. Multi-Tiered Intrusion Detection System Using Machine Learning (2022):

Η εργασία αυτή εξετάζει την υλοποίηση ενός πολυεπίπεδου IDS με χρήση αλγορίθμων μηχανικής μάθησης για την προστασία μεγάλων δικτύων. Η μελέτη αναλύει τη χρήση δεδομένων από πολλαπλές πηγές, όπως δίκτυα κινητής τηλεφωνίας και βιομηχανικές επικοινωνίες.[21]

Πού διαφέρει η παρούσα εργασία ;

Το έργο μας ξεχωρίζει με την ενσωμάτωση δεδομένων από διάφορες πηγές (Zeek, Suricata, NFStream) και την ανάλυση τόσο ροών δεδομένων όσο και μεμονωμένων πακέτων. Χρησιμοποιούμε επίσης προσαρμοσμένα δεδομένα με δικές μας επιθέσεις για την εκπαίδευση των μον-

τέλων, προσαρμοσμένα σε πραγματικά περιβάλλοντα δικτύου. Αυτές οι τεχνικές παρέχουν ένα πολυεπίπεδο σύστημα ανίχνευσης που επιτυγχάνει αυξημένη ακρίβεια και καλύτερη ανίχνευση επιθέσεων σε σύγκριση με τα παραδοσιακά μοντέλα που λαμβάνουν υπόψη μόνο συγκεκριμένα χαρακτηριστικά του δικτύου.[2] [25] [9]

A.2 Συνεισφορά

Η εργασία ακολουθεί την εξής δομή:

- **Συλλογή και ανάλυσή δεδομένων:** Επιλογή κατάλληλων ανοιχτών συνόλων δεδομένων δικτύου, ανάλυση των χαρακτηριστικών τους και εξαγωγή στατιστικών δεδομένων ροών.
- **Ανάλυση των επιθέσεων και εξαγωγή των χαρακτηριστικών:** Διερεύνηση των επιθέσεων που περιέχονται στα δεδομένα και χρήση προσαρμοσμένων επιθέσεων για τον εμπλουτισμό των dataset.
- **Εκπαίδευση και αξιολόγηση μοντέλων ML:** Χρήση των εξαγόμενων χαρακτηριστικών για την εκπαίδευσή μοντέλων όπως το Radom Forest και άλλων αλγορίθμων. Τα μοντέλα αξιολογούνται με βάση την απόδοσή τους στην ανίχνευση εισβολών.
- **Προσαρμοσμένες επιθέσεις και βελτιστοποίηση:** Δημιουργία νέων επιθέσεων για την ενίσχυση του συνόλου δεδομένων και την περαιτέρω αξιολόγηση της αποτελεσματικότητας των μοντέλων.

A.3 Ενδεικτική Δομή της πτυχιακής εργασίας περιλαμβάνει τα παρακάτω βασικά κεφάλαια και ενότητες:

1. Περίληψη: Εδώ παρουσιάζεται συνοπτικά η εργασία, περιλαμβάνοντας το αντικείμενο, τους στόχους, τη μεθοδολογία και τα βασικά συμπεράσματα.
2. Πρόλογος και Ευχαριστίες: Στην ενότητα αυτή αναφέρονται οι προσωπικές ευχαριστίες του συγγραφέα προς τον επιβλέποντα καθηγητή, την επιτροπή και τα άτομα που συνέβαλαν στην εργασία.
3. Εισαγωγή: Παρουσιάζεται το γενικό πλαίσιο και το πρόβλημα που εξετάζεται στην εργασία, οι στόχοι και η σημασία του θέματος.
4. Μεθοδολογία: Εδώ περιγράφονται τα βήματα που ακολούθησε ο συγγραφέας για τη συλλογή, ανάλυση, και επεξεργασία των δεδομένων, καθώς και η επιλογή των μοντέλων μηχανικής μάθησης.
5. Κακόβουλες Επιθέσεις και οι Χαρακτηρισμοί τους: Εδώ αναλύονται οι διαφορετικοί τύποι κακόβουλων επιθέσεων που περιέχονται στα δεδομένα, όπως DoS, DDoS, SQL Injection, XSS κ.ά.
6. Προεπεξεργασία Δεδομένων: Περιγράφεται η διαδικασία καθαρισμού των δεδομένων και η προετοιμασία τους για την εκπαίδευση των μοντέλων μηχανικής μάθησης.
7. Ανίχνευση Εισβολών μέσω Μηχανικής Μάθησης: Αναλύεται η εκπαίδευση και αξιολόγηση των μοντέλων, όπως Random Forest, SVM, και Neural Networks, και τα αποτελέσματα της απόδοσής τους.
8. Συμπεράσματα: Αναλύονται τα κύρια συμπεράσματα της εργασίας, η συνεισφορά της στην ασφάλεια δικτύων και οι μελλοντικές κατευθύνσεις για έρευνα.

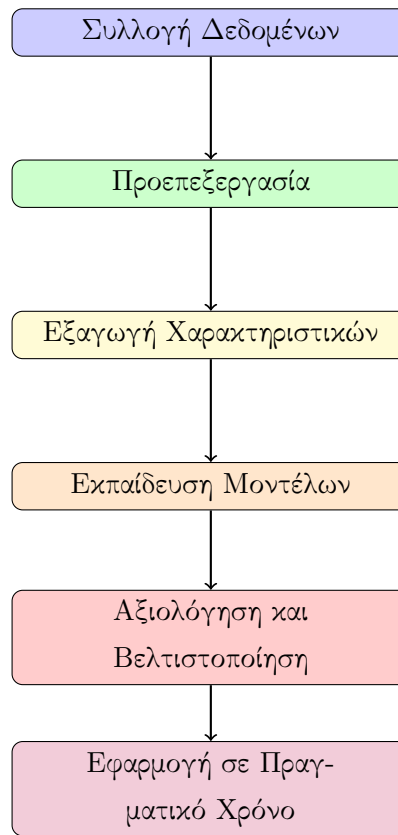
9. Συντμήσεις: Παρουσιάζονται οι συντμήσεις των όρων που χρησιμοποιούνται στην εργασία.
10. Γλωσσάρι Ξενικών Όρων: Εδώ γίνεται αναφορά σε όρους που αναφέρονται στα αγγλικά, συνοδευόμενοι από την ελληνική τους μετάφραση.

Κεφάλαιο Β

Μεθοδολογία

Η μεθοδολογία που ακολουθήθηκε στην παρούσα εργασία επικεντρώνεται στην ανάπτυξη ενός IDS μέσω της χρήσης τεχνικών μηχανικής μάθησης και ανάλυσης ροών δικτύου. Η διαδικασία περιλαμβάνει την επιλογή κατάλληλων datasets, την ανάλυση και προεπεξεργασία των δεδομένων, την εξαγωγή στατιστικών χαρακτηριστικών από πακέτα δικτύου, καθώς και την εκπαίδευση και αξιολόγηση μοντέλων μηχανικής μάθησης για την ανίχνευση κακόβουλων δραστηριοτήτων.

Για την ανάπτυξη του συστήματός, επιλέχθηκαν ανοιχτά datasets από διάφορες πηγές, συμπεριλαμβανομένων των CICIDS 2017, Defcon και Netresec Datasets. Τα δεδομένα αυτά περιλαμβάνουν πραγματικές ροές δικτύου και επίθεσις ποικίλων τύπων, όπως DoS και DDoS επιθέσεις, brute force επιθέσεις σε πρωτόκολλα όπως SSH και HTTP, καθώς και επιθέσεις μέσω ευπαθών δικτυακών πρωτοκόλλων όπως το DNS. Η επιλογή των datasets αυτών εξασφαλίζει την ποικιλομορφία στις επιθέσεις, διευκολύνοντας έτσι την ανάπτυξη ενός αποτελεσματικού IDS.



Στην ενότητα αυτή, θα αναλύσουμε τις επιθέσεις που περιέχονται στα επιλεγμένα datasets. Η ανάλυση αυτή περιλαμβάνει τις τεχνικές που χρησιμοποιούνται, καθώς και τα χαρακτηριστικά που μπορούμε να εξάγουμε από τα δεδομένα δικτυακής Κυκλοφορίας για να ανιχνεύσουμε αυτές τις επιθέσεις. Αυτό θα βοηθήσει στην καλύτερη κατανόηση του τρόπου με το οποίο μπορούν να εκπαιδευτούν και να αξιολογηθούν τα μοντέλα ανίχνευσης εισβολών.

1. **Προεπεξεργασία δεδομένων:** Αφού κατεβάσαμε τα PCAP από τις πηγές μας (Defcon, Netresec, CICIDS 2017 Dataset), τα αναλύσαμε με τα εργαλεία Suricata, Zeek και NFStream, ακολουθώντας μια διαδικασία προεπεξεργασίας για την εξαγωγή χρήσιμων πληροφοριών. Αυτό περιελάμβανε τον καθαρισμό των δεδομένων, τη διαχείριση των ελλιπών τιμών και τη μετατροπή των δεδομένων σε κατάλληλη μορφή για την εκπαίδευση των μοντέλων μας. Με αυτόν τον τρόπο εξασφαλίσαμε ότι τα δεδομένα ήταν επαρκώς οργανωμένα και αξιόπιστα ώστε να παρέχουν ακριβή και ουσιαστικά αποτελέσματα για την εκπαίδευση του συστήματος ανίχνευσης εισβολών.

Listing B.1: Εκτέλεση του *Suricata* για ανάλυση PCAP και παραγωγή *.json*

```
1 suricata -r /path/to/OnePCAP.pcap -l /path/to/file/for/save
```

Listing B.2: Εκτέλεση του *Zeek* για ανάλυση PCAP και εξαγωγή *logs*

```
1 zeek -r input.pcap
```

Listing B.3: Συγχώνευση αρχείων PCAP με το *mergcap*

```
1 mergcap -w /home/kali/Documents/PoPo2/merged_output.pcap /home/↵
    kali/Documents/PoPo2/pcapchu/*.pcap
```

```

1 import os
2 import pandas as pd
3 from nfstream import NFStreamer
4 import glob
5
6 # Helper to convert protocol numbers to string
7 PROTOCOL_MAP = {6: 'TCP', 17: 'UDP', 1: 'ICMP'} # Add more protocols as needed
8
9 def process_nfstream_in_chunks(pcap_file, output_csv, chunk_size=10000):
10     """
11     Επεξεργασία PCAP αρχείου σε chunks και αποθήκευση σε CSV με σωστή μορφή για συγχώνευση με Suricata και Zeek.
12     """
13     # Έλεγχος αν το PCAP αρχείο υπάρχει
14     if not os.path.exists(pcap_file):
15         raise FileNotFoundError(f"Το PCAP αρχείο δεν βρέθηκε: {pcap_file}")
16
17     # Δημιουργία NFStreamer για το PCAP αρχείο
18     streamer = NFStreamer(source=pcap_file)
19
20     # Αρχικοποίηση του αρχείου CSV, γράφοντας τις επικεφαλίδες
21     if not os.path.exists(output_csv):
22         with open(output_csv, 'w') as f:
23             f.write('timestamp,src_ip,dest_ip,src_port,dest_port,protocol\n')
24
25     # Επεξεργασία των flows σε chunks
26     flows_chunk = []
27     count = 0
28
29     for flow in streamer:
30         # Προσθήκη των δεδομένων με σωστή μορφή για το timestamp και το πρωτόκολλο
31         flows_chunk.append({
32             'timestamp': pd.to_datetime(flow.bidirectional_first_seen_ms, unit='ms').strftime('%Y-%m-%dT%H:%M:%S'),
33             'src_ip': flow.src_ip,
34             'dest_ip': flow.dst_ip,
35             'src_port': flow.src_port,
36             'dest_port': flow.dst_port,
37             'protocol': PROTOCOL_MAP.get(flow.protocol, str(flow.protocol)) # Μετατροπή αριθμητικών πρωτοκόλλων σε string
38         })
39         count += 1
40
41     # Γράφουμε τα δεδομένα στο CSV όταν φτάνουμε το μέγεθος του chunk
42     if count % chunk_size == 0:
43         df_chunk = pd.DataFrame(flows_chunk)
44         df_chunk.to_csv(output_csv, mode='a', header=False, index=False)
45         flows_chunk = [] # Αδειάζουμε το chunk για το επόμενο batch
46
47     # Γράφουμε τα υπόλοιπα δεδομένα
48     if flows_chunk:
49         df_chunk = pd.DataFrame(flows_chunk)
50         df_chunk.to_csv(output_csv, mode='a', header=False, index=False)
51
52     print(f"Η επεξεργασία του {pcap_file} ολοκληρώθηκε και τα δεδομένα αποθηκεύτηκαν στο {output_csv}.")
53
54 if __name__ == "__main__":
55     # Αναζήτηση για το πρώτο διαθέσιμο αρχείο .pcap στον τρέχοντα φάκελο
56     pcap_files = glob.glob("*.pcap")
57
58     if not pcap_files:
59         print("Δεν βρέθηκε κανένα αρχείο PCAP στον τρέχοντα φάκελο.")
60         exit(1)
61
62     # Επιλογή του πρώτου αρχείου PCAP
63     pcap_file = pcap_files[0]
64     print(f"Επιλεγμένο PCAP αρχείο: {pcap_file}")
65
66     # Διαδρομή για το αρχείο εξόδου CSV
67     output_csv = 'PRONfstream_data.csv' # Μπορείς να αλλάξεις τη διαδρομή αν χρειάζεται
68
69     # Επεξεργασία του PCAP αρχείου και αποθήκευση των δεδομένων σε CSV
70     process_nfstream_in_chunks(pcap_file, output_csv)
71

```

Σχήμα Β.1: *Enter Caption*

2. Για να ενοποιήσουμε όλα τα αρχεία CSV σε ένα ενιαίο σύνολο δεδομένων, ακολουθήσαμε τα παρακάτω βήματα. Αρχικά, κάθε αρχείο CSV που προερχόταν από τις πηγές δεδομένων μας (όπως το NFStream, η Zeek και η Suricata) περιείχε διαφορετικές πληροφορίες σχετικά με τις ροές δικτύου και τα χαρακτηριστικά των επιθέσεων. Χρησιμοποιώντας τη βιβλιοθήκη pandas της Python, εισαγάγαμε καθένα από τα αρχεία CSV και προετοιμάσαμε τα δεδομένα τους για συγχώνευση. Οι στήλες που ήταν κοινές μεταξύ των αρχείων (π.χ. διευθύνσεις IP, θύρες, πρωτόκολλα) χρησιμοποιήθηκαν ως κλειδιά για τη συγχώνευση.

Χρησιμοποιήσαμε τη μέθοδο ‘merge()’ της pandas για να συγχωνεύσουμε τα αρχεία CSV σε ένα ενιαίο σύνολο δεδομένων, διασφαλίζοντας ότι δεν υπήρχαν διπλές ή ασύμβατες εγγραφές. Επιπλέον, φροντίσαμε να διαχειριστούμε τα δεδομένα που λείπουν (missing values), ώστε να μην επηρεάζουν την εκπαίδευση του μοντέλου. Το τελικό συγχωνευμένο σύνολο δεδομένων περιείχε όλες τις απαραίτητες πληροφορίες για την ανάλυση και την εκπαίδευση των μοντέλων μηχανικής μάθησης.

```

1 import os
2 import json
3 import pandas as pd
4
5 def process_suricata_eve(file_path, output_path, chunk_size=5000):
6     """ Process Suricata eve.json file and save cleaned data with labels to CSV """
7     processed_data = []
8
9     try:
10         with open(file_path, 'r') as f:
11             # Read JSON lines in chunks
12             for i, line in enumerate(f):
13                 event = json.loads(line)
14
15                 # Check if necessary fields exist
16                 if 'src_ip' in event and 'dest_ip' in event:
17                     # Assign a label based on event type
18                     # Assuming 'alert' indicates malicious traffic
19                     label = 1 if event.get('event_type') == 'alert' else 0
20
21                 # Append processed data
22                 processed_data.append({
23                     'timestamp': event.get('timestamp', None),
24                     'src_ip': event.get('src_ip', '0.0.0.0'), # Default IP if missing
25                     'dest_ip': event.get('dest_ip', '0.0.0.0'), # Default IP if missing
26                     'src_port': event.get('src_port', 0), # Default port if missing
27                     'dest_port': event.get('dest_port', 0), # Default port if missing
28                     'protocol': event.get('proto', 'unknown'), # Default protocol if missing
29                     'event_type': event.get('event_type', 'unknown'), # Default event type if missing
30                     'label': label
31                 })
32
33             # Save to file in chunks for memory management
34             if (i + 1) % chunk_size == 0:
35                 df = pd.DataFrame(processed_data)
36                 df.to_csv(output_path, mode='a', header=not os.path.exists(output_path), index=False)
37                 processed_data = [] # Clear memory
38
39             # Process remaining data
40             if processed_data:
41                 df = pd.DataFrame(processed_data)
42                 df.to_csv(output_path, mode='a', header=not os.path.exists(output_path), index=False)
43
44             print(f"Suricata data processed and saved as {output_path}")
45
46     except Exception as e:
47         print(f"Error processing Suricata eve.json: {e}")
48
49 # Example usage
50 log_dir = '/home/kali/Documents/Models/CSVcreator'
51 suricata_log_path = os.path.join(log_dir, 'eve.json')
52 output_path = os.path.join(log_dir, 'PROprocessed_suricata.csv')
53
54 # Call the function to process the Suricata eve.json file
55 process_suricata_eve(suricata_log_path, output_path)
56

```

Σχήμα Β.2: Δημιουργία του κατάλληλου *dattaset(.csv)* από το *eve.json*

```

import os
import pandas as pd

def get_zeeq_fields(file_path):
    """Extract the Zeek field names from the '#fields' line in the log file"""
    with open(file_path, 'r') as f:
        for line in f:
            if line.startswith('#fields'):
                fields = line.strip().split()[1:] # Skip the '#fields' part
                return fields
    raise ValueError("No '#fields' line found in the Zeek log")

def process_zeeq_log(file_path, output_path, chunk_size=5000):
    """Process Zeek .log files and save cleaned data to CSV"""
    try:
        # Get field names from the Zeek log
        zeeq_fields = get_zeeq_fields(file_path)

        # Define the columns we need
        required_columns = ['ts', 'id.orig_h', 'id.orig_p', 'id.resp_h', 'id.resp_p', 'proto']

        # Ensure the required columns exist in the log
        if not set(required_columns).issubset(zeeq_fields):
            raise ValueError(f"Missing required columns in Zeek log. Available columns: {zeeq_fields}")

        # Map the field names to the expected format
        column_mapping = {
            'ts': 'timestamp',
            'id.orig_h': 'src_ip',
            'id.orig_p': 'src_port',
            'id.resp_h': 'dest_ip',
            'id.resp_p': 'dest_port',
            'proto': 'protocol'
        }

        processed_data = []

        # Read the file in chunks for memory efficiency
        for chunk in pd.read_csv(file_path, sep='\t', comment='#', names=zeeq_fields, chunksize=chunk_size, engine='python'):
            # Filter and rename columns
            cleaned_chunk = chunk[required_columns].rename(columns=column_mapping)

            # Convert Unix timestamp to a format compatible with Suricata
            cleaned_chunk['timestamp'] = pd.to_datetime(cleaned_chunk['timestamp'], unit='s')

            # Append the cleaned chunk to the processed data list
            processed_data.append(cleaned_chunk)

        # If no valid data was found, raise an error
        if not processed_data:
            print("No valid data found in the Zeek log. Please check the log format and ensure the necessary columns exist.")
            return

        # Concatenate all chunks
        final_df = pd.concat(processed_data, ignore_index=True)

        # Save the processed data to CSV
        final_df.to_csv(output_path, index=False)
        print(f"Zeek data processed and saved as {output_path}")

    except FileNotFoundError:
        print(f"File not found: {file_path}")
    except Exception as e:
        print(f"Error processing Zeek logs: {e}")

# Example usage
log_dir = '/home/kali/Documents/Models/CSVcreator'
zeeq_log_path = os.path.join(log_dir, 'conn.log')
output_path = os.path.join(log_dir, 'PROprocessed_zeeq.csv')
process_zeeq_log(zeeq_log_path, output_path)

```

Σχήμα Β.3: Δημιουργία του κατάλληλου *dattaset(.csv)* από το *zeel logs*

```

import os
import pandas as pd

def merge_and_save_in_chunks(nfstream_path, suricata_path, zeek_path, output_path, chunksize=10000):
    """
    Process and merge NfStream, Suricata, and Zeek data in chunks, saving results to disk step by step.
    """

    temp_file_1 = output_path + "_temp1.csv"
    temp_file_2 = output_path + "_temp2.csv"

    # Merge NfStream and Suricata in chunks and save to temp_file_1
    print("Merging NfStream and Suricata in chunks...")
    first_chunk = True
    for nfstream_chunk in pd.read_csv(nfstream_path, chunksize=chunksize):
        for suricata_chunk in pd.read_csv(suricata_path, chunksize=chunksize):
            chunk_merged = pd.merge(nfstream_chunk, suricata_chunk, on=['timestamp', 'src_ip', 'dest_ip', 'src_port', 'dest_port', 'protocol'],
how='outer')
            chunk_merged.to_csv(temp_file_1, mode='a', index=False, header=first_chunk)
            first_chunk = False

    # Merge the temp_file_1 with Zeek in chunks and save to temp_file_2
    print("Merging temp_file_1 with Zeek in chunks...")
    first_chunk = True
    for zeek_chunk in pd.read_csv(zeek_path, chunksize=chunksize):
        for temp_chunk in pd.read_csv(temp_file_1, chunksize=chunksize):
            final_merged = pd.merge(temp_chunk, zeek_chunk, on=['timestamp', 'src_ip', 'dest_ip', 'src_port', 'dest_port', 'protocol'],
how='outer')
            final_merged.to_csv(temp_file_2, mode='a', index=False, header=first_chunk)
            first_chunk = False

    # Fill missing event_type and label values with 'unknown' and 0 respectively
    print("Filling missing event_type and label values...")
    final_merged_data = pd.read_csv(temp_file_2)
    final_merged_data['event_type'].fillna('unknown', inplace=True)
    final_merged_data['label'].fillna(0, inplace=True)

    # Save the final merged data to the output file
    print(f"Saving final merged data to {output_path}...")
    final_merged_data.to_csv(output_path, index=False)

    # Delete temporary files
    os.remove(temp_file_1)
    os.remove(temp_file_2)

    print(f"Η συγχώνευση ολοκληρώθηκε και τα δεδομένα αποθηκεύτηκαν στο {output_path}.")

if __name__ == "__main__":
    # Define file paths
    log_dir = '/home/kali/Documents/Models/CSVcreator'
    nfstream_path = os.path.join(log_dir, 'PROnfstream_data.csv')
    suricata_path = os.path.join(log_dir, 'PROprocessed_suricata.csv')
    zeek_path = os.path.join(log_dir, 'PROprocessed_zeek.csv')
    output_path = os.path.join(log_dir, 'PROfinal_merged_output.csv')

    # Run the process and merge in chunks
    merge_and_save_in_chunks(nfstream_path, suricata_path, zeek_path, output_path)

```

Σχήμα Β.4: Συγχώνευση των *suricat.csv*, *zeek.csv*, *NFStream.csv* σε ένα *merged.csv*.

3. **Εκπαίδευση των Μοντέλων Μηχανικής Μάθησης:** Μετά την εξαγωγή των απαραίτητων χαρακτηριστικών από τα δεδομένα, προχωρήσαμε στην εκπαίδευση διαφόρων αλγορίθμων μηχανικής μάθησης, όπως οι Random Forest, SVM (Support Vector Machine) και Neural Network (NN). Κάθε μοντέλο εκπαιδεύτηκε για την ανίχνευση εισβολών, αξιοποιώντας στατιστικά στοιχεία δικτύου που εξάγονται από ερ-γαλεία ανάλυσης ροής (όπως το NfStream) και επιθέσεις που εντοπίστηκαν μέσω των Zeek και Suricata.

Τα μοντέλα εκπαιδεύτηκαν χρησιμοποιώντας χωριστά σύνολα δεδομένων για εκπαίδευση (train) και δοκιμή (test) για την αξιολόγηση της απόδοσής τους. Στη συνέχεια, τα μοντέλα αποθηκεύτηκαν ως αρχεία .pkl για τα μοντέλα που βασίζονται στο scikit-learn και ως αρχεία .h5 για τα μοντέλα νευρωνικών δικτύων που χρησιμοποιούν τη βιβλιοθήκη Keras. Αυτά τα αποθηκευμένα μοντέλα χρησιμοποιούνται ως μέρος του συστήματος ανίχνευσης εισβολών (IDS) και μπορούν να φορτωθούν και να χρησιμοποιηθούν για την ανίχνευση εισβολών σε πραγματικό χρόνο.

```

import os
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import load_model

# Φόρτωση του dataset
file_path = '/home/kali/Documents/Models/NN/NormalData2.csv'
data = pd.read_csv(file_path)

# Χρήση forward fill για την συμπλήρωση κενών τιμών
data.fillna(method='ffill', inplace=True)

# Εξασφάλιση ότι οι src_port και dest_port είναι αριθμητικές
data['src_port'] = pd.to_numeric(data['src_port'], errors='coerce')
data['dest_port'] = pd.to_numeric(data['dest_port'], errors='coerce')

# Διαγραφή γραμμών όπου οι πόρτες δεν μπόρεσαν να μετατραπούν σε αριθμητικές
data.dropna(subset=['src_port', 'dest_port'], inplace=True)

# Κωδικοποίηση των IP διευθύνσεων ως κατηγορίες με το LabelEncoder
label_encoder_ip = LabelEncoder()
data['src_ip'] = label_encoder_ip.fit_transform(data['src_ip'])
data['dest_ip'] = label_encoder_ip.fit_transform(data['dest_ip'])

# Κωδικοποίηση του πρωτοκόλλου ως κατηγορία
data['protocol'] = data['protocol'].astype('category').cat.codes

# Προετοιμασία του πίνακα χαρακτηριστικών X και του στόχου y
X = data[['src_ip', 'src_port', 'dest_ip', 'dest_port', 'protocol']]
y = data['label']

# Κανονικοποίηση των αριθμών των ports
scaler = StandardScaler()
X[['src_port', 'dest_port']] = scaler.fit_transform(X.loc[:, ['src_port', 'dest_port']])

# Διαχωρισμός του dataset σε σύνολα εκπαίδευσης και ελέγχου
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Έλεγχος αν υπάρχει ήδη αποθηκευμένο μοντέλο
pkl_dir = '/home/kali/Documents/Models/NN/H5/'
pkl_file_path = os.path.join(pkl_dir, '3model.h5')

if os.path.exists(pkl_file_path):
    print("Φόρτωση υπάρχοντος μοντέλου ...")
    model = load_model(pkl_file_path)

    # Επανασύνθεση του μοντέλου για εκπαίδευση με νέο optimizer
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
else:
    print("Δημιουργία νέου μοντέλου ...")
    # Δημιουργία του μοντέλου
    model = Sequential()
    model.add(Dense(64, input_dim=X_train.shape[1], activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))

    # Συμπίληση του μοντέλου
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Εκπαίδευση του μοντέλου
model.fit(X_train, y_train, epochs=10, batch_size=64, verbose=1)

# Αξιολόγηση του μοντέλου
accuracy = model.evaluate(X_test, y_test, verbose=1)
print(f"Ακρίβεια του Μοντέλου: {accuracy[1]}")

# Αποθήκευση του μοντέλου
model.save(pkl_file_path)
print(f"Το μοντέλο αποθηκεύτηκε ως {pkl_file_path}")

```

```

import os
import pandas as pd
import joblib
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score

# Define paths
file_path = '/home/kali/Documents/Models/RFC/PROprocessed_suricata.csv'
pkl_directory = '/home/kali/Documents/Models/RFC/PKL'
pkl_file_path = os.path.join(pkl_directory, 'rf_model.pkl')

# Load the dataset
data = pd.read_csv(file_path)

# Fill missing values in 'label' and other relevant columns
data.ffill(inplace=True) # Use forward fill to handle missing values

# Drop non-numeric columns that are not useful for training
data = data.drop(columns=['timestamp', 'src_ip', 'dest_ip', 'protocol', 'event_type'])

# Convert src_port and dest_port to numeric, forcing errors to NaN and dropping those rows
data['src_port'] = pd.to_numeric(data['src_port'], errors='coerce')
data['dest_port'] = pd.to_numeric(data['dest_port'], errors='coerce')

# Drop rows where ports are NaN after conversion
data.dropna(subset=['src_port', 'dest_port'], inplace=True)

# Separate features (X) and target (y)
X = data.drop(columns=['label']) # Features
y = data['label'] # Target

# Split the dataset into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Check if there is an existing .pkl model
if os.path.exists(pkl_file_path):
    # Load the existing model
    print("Loading existing Random Forest model from .pkl file ...")
    rf_model = joblib.load(pkl_file_path)

    # Retrain the existing model
    print("Retraining the Random Forest model ...")
    rf_model.fit(X_train, y_train)
else:
    # Train a new Random Forest Classifier
    print("Training a new Random Forest model ...")
    rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
    rf_model.fit(X_train, y_train)

# Make predictions
y_pred = rf_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy}")
print("Classification Report:")
print(classification_report(y_test, y_pred))

# Save the model to .pkl file
print("Saving the Random Forest model to .pkl file ...")
joblib.dump(rf_model, pkl_file_path)
print(f"Model saved to {pkl_file_path}")

```

Σχήμα Β.6: Ο Random Forest εκπαιδεύετε με τα merge.csv και παράγει το μοντέλο RadomForst.pkl

```
import dask.dataframe as dd
import pandas as pd
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

# Load your dataset in a memory-friendly way using Dask
data = dd.read_csv("/home/kali/Documents/PoPo2/PROfinal_merged_output.csv", usecols=['src_port', 'dest_port', 'protocol', 'label'])

# Convert Dask DataFrame to Pandas (you can limit rows for testing using .head(N))
data = data.compute()

# Encode the 'protocol' column as numerical values
label_encoder = LabelEncoder()
data['protocol'] = label_encoder.fit_transform(data['protocol'])

# Filter out rows with missing labels if necessary
data = data.dropna(subset=['label'])

# Define the feature columns and label column
X = data[['src_port', 'dest_port', 'protocol']]
y = data['label']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Train the SVM model with optimized parameters
svm_model = SVC(kernel='rbf', C=1, gamma='auto', cache_size=200) # You can adjust cache_size based on memory
svm_model.fit(X_train_scaled, y_train)

# Test the model
y_pred = svm_model.predict(X_test_scaled)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"SVM Model Accuracy: {accuracy:.4f}")
```

Σχήμα Β.7: Ο Support Vector Machine εκπαιδεύετε με τα merge.csv

4. **Αξιολόγηση και βελτιστοποίηση μοντέλων:** τα μοντέλα αξιολογήθηκαν με βάση μετρήσεις επιδόσεων όπως η ακρίβεια, η ανάκληση και το F1-score. Οι υπερ-παράμετροι βελτιστοποιήθηκαν επίσης για καλύτερες επιδόσεις.

```

Epoch 1/10
27885/27885 — 39s 1ms/step - accuracy: 0.9954 - loss: 0.0157
Epoch 2/10
27885/27885 — 39s 1ms/step - accuracy: 0.9992 - loss: 0.0029
Epoch 3/10
27885/27885 — 37s 1ms/step - accuracy: 0.9993 - loss: 0.0025
Epoch 4/10
27885/27885 — 41s 1ms/step - accuracy: 0.9993 - loss: 0.0025
Epoch 5/10
27885/27885 — 40s 1ms/step - accuracy: 0.9993 - loss: 0.0024
Epoch 6/10
27885/27885 — 39s 1ms/step - accuracy: 0.9993 - loss: 0.0023
Epoch 7/10
27885/27885 — 38s 1ms/step - accuracy: 0.9993 - loss: 0.0022
Epoch 8/10
27885/27885 — 38s 1ms/step - accuracy: 0.9993 - loss: 0.0021
Epoch 9/10
27885/27885 — 37s 1ms/step - accuracy: 0.9993 - loss: 0.0022
Epoch 10/10
27885/27885 — 37s 1ms/step - accuracy: 0.9993 - loss: 0.0021
23902/23902 — 23s 965us/step - accuracy: 0.9994 - loss: 0.0021
Ακρίβεια του Μοντέλου: 0.9993959665298462

```

```

Loading existing Random Forest model from .pkl file ...
Retraining the Random Forest model ...
Model Accuracy: 0.9997470062405127
Classification Report:

```

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	213771
1.0	1.00	1.00	1.00	296123
accuracy			1.00	509894
macro avg	1.00	1.00	1.00	509894
weighted avg	1.00	1.00	1.00	509894

```

Saving the Random Forest model to .pkl file ...
Model saved to /home/kali/Documents/Models/RFC/PKL/rf_model.pkl
(kali@kali)~[~/Documents/Models/RFC]
$
(kali@kali)~[~/Documents/Models/RFC]
$

```

Σχήμα Β.8: Αξιολόγηση και βελτιστοποίηση μοντέλων

5. **Δημιουργία συστήματος IDS σε πραγματικό χρόνο:** Τέλος, χρησιμοποιήσαμε τα εκπαιδευμένα μοντέλα για να δημιουργήσουμε ένα σύστημα ανίχνευσης εισβολών σε πραγματικό χρόνο.

```
import os
import nfstream
import pandas as pd
import time
import pickle
from tensorflow.keras.models import load_model
from datetime import datetime
import glob

# Διάστημα καταγραφής σε δευτερόλεπτα και μέγιστος αριθμός πακέτων
capture_interval = 60 # 1 λεπτό
capture_packets = 100 # 100 πακέτα πριν δημιουργηθεί νέο αρχείο PCAP

# Φόρτωση όλων των μοντέλων .pkl από το φάκελο /PKL
pkl_models = []
for model_path in glob.glob('/home/kali/Documents/IDS/PKL/*.pkl'):
    with open(model_path, 'rb') as f:
        pkl_models.append(pickle.load(f))

# Φόρτωση όλων των .h5 μοντέλων από το φάκελο /H5
h5_models = []
for model_path in glob.glob('/home/kali/Documents/IDS/H5/*.h5'):
    try:
        h5_models.append(load_model(model_path))
    except Exception as e:
        print(f"Σφάλμα φόρτωσης μοντέλου από το {model_path}: {e}")

# Συνάρτηση για δημιουργία της CSV γραμμής από το NFStream flow
def create_flow_dict(flow):
    return {
        'timestamp': pd.to_datetime(flow.bidirectional_first_seen_ms, unit='ms').strftime('%Y-%m-%dT%H:%M:%S'),
        'src_ip': flow.src_ip,
        'dest_ip': flow.dst_ip,
        'src_port': flow.src_port,
        'dest_port': flow.dst_port,
        'protocol': flow.protocol,
        'flow_size': flow.bidirectional_bytes, # Προσθέτουμε το μέγεθος της ροής
        'duration': flow.bidirectional_duration_ms / 1000, # Διάρκεια ροής σε δευτερόλεπτα
        'label': 0.0 # Default label
    }
```

Σχήμα Β.9: *Live Capture IDS:FIRST*

```

# Συνάρτηση για ανάλυση των δεδομένων με πλειοψηφία μοντέλων και λεπτομέρειες
def ensemble_prediction(df, original_flows):
    # Ελέγχουμε αν οι στήλες υπάρχουν πριν προσπαθήσουμε να τις αφαιρέσουμε
    columns_to_drop = ['timestamp', 'src_ip', 'dest_ip', 'protocol']
    df = df.drop(columns=[col for col in columns_to_drop if col in df.columns])

    print(f"Σχήμα δεδομένων που στέλνονται στα μοντέλα: {df.shape}")

    if df.shape[1] != 5: # Ελέγχος ότι έχουμε τις σωστές στήλες για το μοντέλο
        print(f"Σφάλμα: Τα δεδομένα δεν έχουν τις αναμενόμενες 5 στήλες, αλλά {df.shape[1]} στήλες.")
        return

    predictions = []

    # Προβλέψεις από τα μοντέλα .pkl
    for model in pkl_models:
        try:
            if hasattr(model, 'predict'):
                predictions.append(model.predict(df))
        except Exception as e:
            print(f"Σφάλμα στο μοντέλο .pkl: {e}")

    # Προβλέψεις από τα μοντέλα .h5
    for model in h5_models:
        try:
            if hasattr(model, 'predict'):
                predictions.append(model.predict(df))
        except Exception as e:
            print(f"Σφάλμα στο μοντέλο .h5: {e}")

    # Πλειοψηφία μοντέλων - Αν το 50% ή περισσότερα μοντέλα ανιχνεύσουν επίθεση
    if len(predictions) > 0: # Βεβαιωθείτε ότι υπάρχουν προβλέψεις
        attack_votes = sum([1 for pred in predictions if any(pred == 1)])

        if attack_votes / len(predictions) > 0.5:
            print(f"🚩 Red flags detected: Πιθανή επίθεση ανιχνεύτηκε από την πλειοψηφία των μοντέλων!")
            # Εμφάνιση λεπτομερειών για τις ροές που ανιχνεύτηκαν
            for index, row in df.iterrows():
                print(f"📄 Ειδοποίηση για ροή: {original_flows.iloc[index]['src_ip']} → {original_flows.iloc[index]['dest_ip']}, Port: {original_flows.iloc[index]['src_port']} → {original_flows.iloc[index]['dest_port']}, Protocol: {original_flows.iloc[index]['protocol']}, Time: {original_flows.iloc[index]['timestamp']}")
            else:
                print("Δεν ανιχνεύτηκε επίθεση από την πλειοψηφία των μοντέλων.")
        else:
            print("Δεν έγιναν προβλέψεις λόγω σφαλμάτων.")

# Συνάρτηση για συνεχή καταγραφή και ανάλυση των πακέτων
def live_capture():
    while True:
        # Καταγραφή πακέτων σε PCAP με tcpdump
        tcpdump_cmd = f"sudo tcpdump -i eth0 -w live_capture.pcap -G {capture_interval} -C {capture_packets}"
        os.system(tcpdump_cmd)

        # Ανάλυση των πακέτων με NFStream
        streamer = nfstream.NFStream(source="live_capture.pcap", decode_tunnels=True)
        flows = [create_flow_dict(flow) for flow in streamer]

        # Δημιουργία DataFrame για ανάλυση με τα μοντέλα
        df = pd.DataFrame(flows)
        original_flows = pd.DataFrame(original_flows) # Κρατάμε τα αρχικά δεδομένα για την εμφάνιση
        ensemble_prediction(df, original_flows)

if __name__ == "__main__":
    try:
        live_capture()
    except KeyboardInterrupt:
        print("To IDS σύστημα σταμάτησε.")

```

Σχήμα B.10: *Live Capture IDS:SECOND*

Μετά την επιλογή των δεδομένων, ακολούθησε λεπτομερής ανάλυση των χαρακτηριστικών τους. Στόχος της ανάλυσης αυτής ήταν η κατανόηση των τύπων επιθέσεων και ο προσδιορισμός των σημαντικότερων χαρακτηριστικών για την ανίχνευση κακόβουλων δραστηριοτήτων. Η ανάλυση αυτή περιελάμβανε την κατηγοριοποίηση των επιθέσεων, την καταγραφή των χρησιμοποιούμενων πρωτοκόλλων, καθώς και την εξέταση της διάρκειας των επιθέσεων και της συμπεριφοράς των πακέτων κατά τη μεταφορά τους στο δίκτυο. Οι πληροφορίες αυτές είναι απαραίτητες για την αποτελεσματική εκπαίδευση των μοντέλων μηχανικής μάθησης.

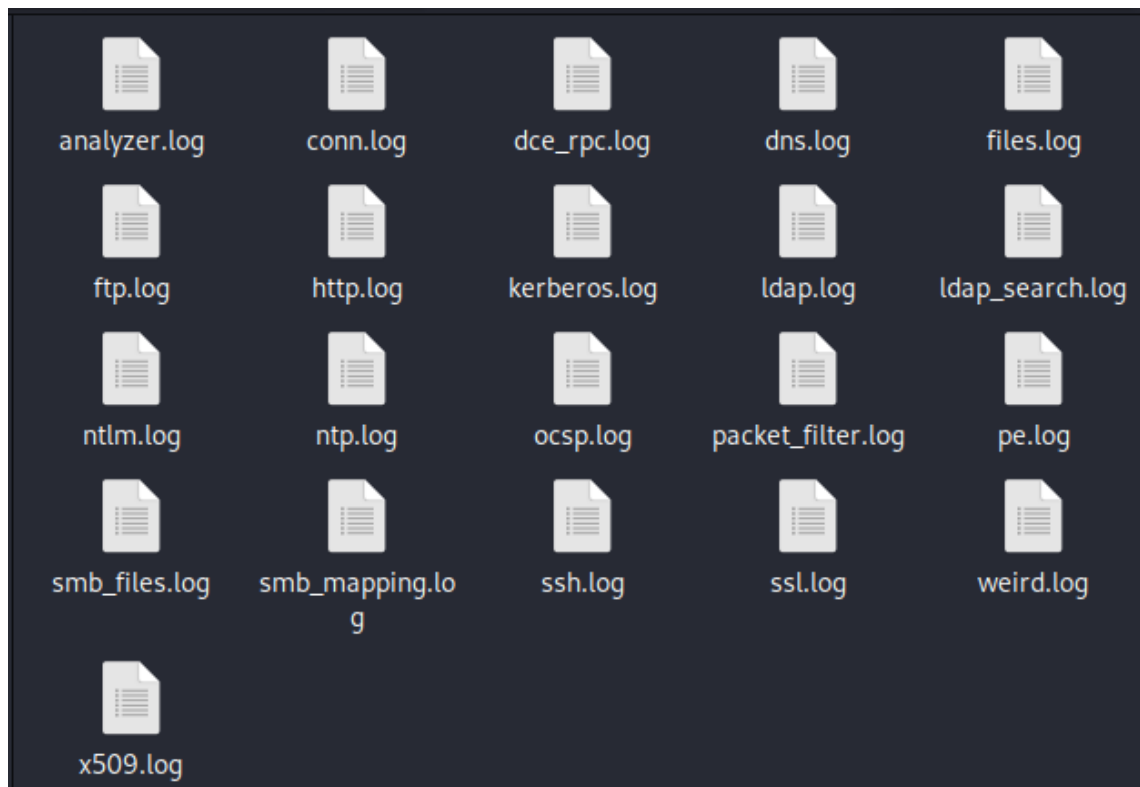
Για την εξαγωγή στατιστικών δεδομένων από τα αρχεία PCAP χρησιμοποιήθηκαν τα εργαλεία NFStream και CICFlowMeter, τα οποία παρέχουν στατιστικές πληροφορίες σχετικά με την κυκλοφορία του δικτύου, όπως ο αριθμός των πακέτων, ο αριθμός των bytes, οι διευ-

θύνσεις IP, η διάρκεια της ροής και τα πρωτόκολλα που χρησιμοποιήθηκαν. Τα δεδομένα αυτά μετατράπηκαν σε μορφή κατάλληλη για την εκπαίδευση μοντέλων μηχανικής μάθησης, επιτρέποντας έτσι την ανίχνευση ανωμαλιών στην κυκλοφορία του δικτύου (τα εργαλεία θα συζητηθούν περαιτέρω στο επόμενο κεφάλαιο).

Το Zeek (πρώην Bro) είναι ένα ισχυρό εργαλείο παρακολούθησης και ανάλυσης δικτύου που χρησιμοποιείται ευρέως σε εφαρμογές ασφαλείας για την καταγραφή και ανάλυση της δραστηριότητας σε ένα δίκτυο. Αναπτύχθηκε για να παρέχει πληροφορίες υψηλής ποιότητας σχετικά με τις ροές δικτύου, καταγράφοντας δεδομένα που μπορούν να χρησιμοποιηθούν για τον εντοπισμό ανωμαλιών, επιθέσεων και άλλων απειλών δικτύου. Στην παρούσα διπλωματική εργασία, το Zeek χρησιμοποιήθηκε ως εργαλείο εξαγωγής χαρακτηριστικών από αρχεία PCAP, τα οποία στη συνέχεια ενσωματώθηκαν στο σύστημα ανίχνευσης εισβολών (IDS).

Τι είναι το Zeek ;

Το Zeek είναι ένα πλαίσιο που αναλύει την κυκλοφορία του δικτύου με γνώμονα την ασφάλεια και παράγει ένα λεπτομερές αρχείο καταγραφής της δραστηριότητας του δικτύου. Σε αντίθεση με άλλα εργαλεία παρακολούθησης δικτύου, το Zeek δεν εστιάζει μόνο στην αναγνώριση πακέτων ή στην ταχύτητα κίνησης, αλλά στη βαθύτερη ανάλυση των ροών, των συνδέσεων και των συναλλαγών μεταξύ των κόμβων του δικτύου. Τα δεδομένα που εξάγει έχουν τη μορφή αρχείων καταγραφής, τα οποία περιέχουν πληροφορίες για όλες τις συνδέσεις, τα πρωτόκολλα και τις σχετικές λεπτομέρειες.



Σχήμα Β.11: Zeek Logs output

Βασικές Δυνατότητες του Zeek :

Ανάλυση ροής δικτύου: Η Zeek παρακολουθεί και καταγράφει κάθε σύνδεση που πραγματοποιείται σε ένα δίκτυο, αποθηκεύοντας τις πληροφορίες σε αρχεία καταγραφής, όπως το conn.log, το οποίο περιέχει δεδομένα για όλες τις συνδέσεις TCP και UDP.

Καταγραφή Πρωτοκόλλων:

Παρέχει καταγραφές για διάφορα πρωτόκολλα, όπως :

- **HTTP (http.log):** Καταγράφει όλα τα αιτήματα και τις απαντήσεις HTTP, παρέχοντας πληροφορίες σχετικά με τη δραστηριότητα της διαδικτυακής εφαρμογής.

- **DNS (dns.log):** Παρακολουθεί και καταγράφει ερωτήματα DNS, επιτρέποντας την ανίχνευση δυνητικά παραπλανητικών ονομάτων τομέα.
- **SSL/TLS (ssl.log):** καταγράφει τις ασφαλείς συνδέσεις που χρησιμοποιούν τα πρωτόκολλα SSL/TLS, παρέχοντας πληροφορίες σχετικά με τις ασφαλείς επικοινωνίες.

προσαρμοστικότητα: Το Zeek είναι εξαιρετικά προσαρμόσιμο μέσω της γλώσσας σεναρίων Zeek, η οποία επιτρέπει στους χρήστες να δημιουργούν προσαρμοσμένες δέσμες ενεργειών για την ανίχνευση συγκεκριμένων επιθέσεων ή την ανάλυση συγκεκριμένων δραστηριοτήτων στο δίκτυο.

Ανίχνευση ανωμαλιών: Το Zeek μπορεί να χρησιμοποιηθεί για την ανίχνευση ανωμαλιών στις ροές του δικτύου, όπως ανώμαλη κυκλοφορία ή ασυνήθιστες συνδέσεις, οι οποίες μπορεί να αποτελούν ενδείξεις επιθέσεων.

Χρήση του Zeek στην Εργασία:

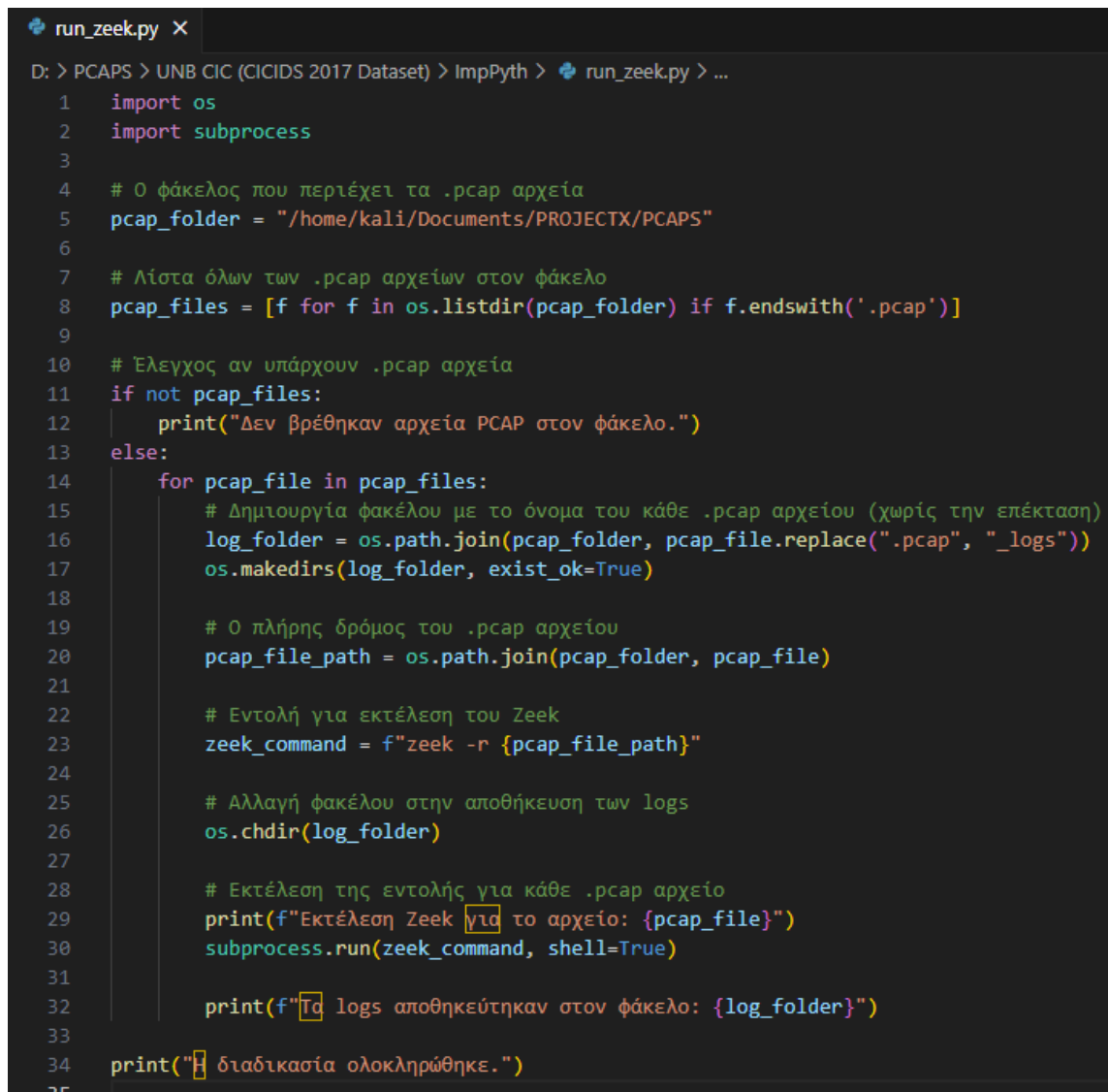
Στην παρούσα διατριβή, το Zeek χρησιμοποιήθηκε για την ανάλυση των αρχείων PCAP, παρέχοντας χρήσιμες πληροφορίες σχετικά με τις ροές δικτύου που χρησιμοποιήθηκαν για την εκπαίδευση των μοντέλων μηχανικής μάθησης στο σύστημα ανίχνευσης εισβολών (IDS). Τα δεδομένα που εξήχθησαν από το Zeek περιλάμβαναν:

- **Συνδέσεις (conn.log):** καταγράφει όλες τις συνδέσεις TCP/UDP, με πληροφορίες όπως η διεύθυνση IP πηγής και προορισμού, οι θύρες και η διάρκεια της σύνδεσης.
- **Αιτήματα DNS (dns.log):** τα καταγεγραμμένα ερωτήματα DNS ήταν χρήσιμες πληροφορίες για την ανίχνευση επιθέσεων με στόχο το σύστημα DNS, όπως η εναλλαγή DNS ή παραπλανητικά ερωτήματα.
- **Συνδέσεις HTTP (http.log):** HTTP: Τα αρχεία καταγραφής HTTP χρησιμοποιήθηκαν για την ανάλυση της δραστηριότητας των εφαρμογών ιστού και τον εντοπισμό πιθανών επιθέσεων μέσω του πρωτοκόλλου HTTP.
- **SSL/TLS (ssl.log):** οι ασφαλείς συνδέσεις ήταν κρίσιμα δεδομένα για τον εντοπισμό ασφαλών και μην ασφαλών συνδέσεων στο δίκτυο.

Η χρήση του Zeek συνέβαλε σημαντικά στην ανάλυση των δεδομένων του δικτύου, επιτρέποντας την ανίχνευση σύνθετων επιθέσεων και παρέχοντας χρήσιμα χαρακτηριστικά για την εκπαίδευση του μοντέλου. Τα δεδομένα από το Zeek σε συνδυασμό με δεδομένα από άλλα εργαλεία, όπως το Suricata, παρείχαν μια πλήρη εικόνα της δραστηριότητας του δικτύου και των επιθέσεων που σημειώθηκαν.

Πλεονεκτήματα της χρήσης του Zeek:

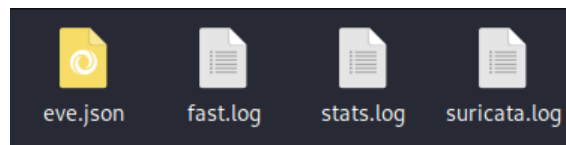
1. **Πλούσια δεδομένα:** Το Zeek παρέχει λεπτομερείς πληροφορίες για κάθε δραστηριότητα δικτύου, καθιστώντας το ένα εξαιρετικό εργαλείο για την ανίχνευση επιθέσεων και την ανάλυση δικτύου.
2. **Συνδυασμός πληροφοριών:** Τα δεδομένα από τα αρχεία καταγραφής της Zeek μπορούν να χρησιμοποιηθούν για την κατανόηση της πλήρους δραστηριότητας του δικτύου, συμπεριλαμβανομένων όλων των τύπων επικοινωνίας και πρωτοκόλλων.
3. **Αξιοπιστία:** Η αξιοπιστία του Zeek όσον αφορά την ανίχνευση ανωμαλιών δικτύου και την καταγραφή της κίνησης δικτύου το καθιστά ιδανικό εργαλείο για χρήση σε συστήματα ανίχνευσης εισβολών (IDS).
4. **Επεκτασιμότητα και ευελιξία:** Η Zeek είναι εξαιρετικά ευέλικτη και μπορεί να προσαρμοστεί στις ανάγκες κάθε δικτύου. Μέσω σεναρίων (γραμμένων στη γλώσσα Zeek), επιτρέπει τη δημιουργία κανόνων για την ανίχνευση συγκεκριμένων επιθέσεων ή ασυνήθιστων μοτίβων κίνησης.
5. **Ενσωμάτωση με άλλα εργαλεία:** Εύκολη συνεργασία με άλλα εργαλεία ανίχνευσης όπως Suricata και Elastic Stack, για τη συλλογή, ανάλυση και οπτικοποίηση δεδομένων.
6. **Ανοιχτός κώδικας και ενεργή κοινότητα:** Το Zeek είναι ένα εργαλείο ανοιχτού κώδικα με μια ενεργή κοινότητα ανάπτυξης και υποστήριξης. Αυτό σημαίνει ότι ενσωματώνονται συχνά νέα χαρακτηριστικά και βελτιώσεις, με αποτέλεσμα να αναβαθμίζεται συνεχώς και να ανταποκρίνεται στις σύγχρονες απαιτήσεις ασφαλείας.
7. **Αυτοματοποιημένη ανάλυση:** με τις δυνατότητες scripting και αυτοματοποίησης, το Zeek μπορεί να ανιχνεύει αυτόματα ανωμαλίες και ύποπτες δραστηριότητες σε πραγματικό χρόνο, παρέχοντας έγκαιρες ειδοποιήσεις για πιθανές επιθέσεις.



```
run_zEEK.py X
D: > PCAPS > UNB CIC (CICIDS 2017 Dataset) > ImpPyth > run_zEEK.py > ...
1  import os
2  import subprocess
3
4  # Ο φάκελος που περιέχει τα .pcap αρχεία
5  pcap_folder = "/home/kali/Documents/PROJECTX/PCAPS"
6
7  # Λίστα όλων των .pcap αρχείων στον φάκελο
8  pcap_files = [f for f in os.listdir(pcap_folder) if f.endswith('.pcap')]
9
10 # Έλεγχος αν υπάρχουν .pcap αρχεία
11 if not pcap_files:
12     print("Δεν βρέθηκαν αρχεία PCAP στον φάκελο.")
13 else:
14     for pcap_file in pcap_files:
15         # Δημιουργία φακέλου με το όνομα του κάθε .pcap αρχείου (χωρίς την επέκταση)
16         log_folder = os.path.join(pcap_folder, pcap_file.replace(".pcap", "_logs"))
17         os.makedirs(log_folder, exist_ok=True)
18
19         # Ο πλήρης δρόμος του .pcap αρχείου
20         pcap_file_path = os.path.join(pcap_folder, pcap_file)
21
22         # Εντολή για εκτέλεση του Zeek
23         zeek_command = f"zeek -r {pcap_file_path}"
24
25         # Αλλαγή φακέλου στην αποθήκευση των logs
26         os.chdir(log_folder)
27
28         # Εκτέλεση της εντολής για κάθε .pcap αρχείο
29         print(f"Εκτέλεση Zeek για το αρχείο: {pcap_file}")
30         subprocess.run(zeek_command, shell=True)
31
32         print(f"Τα logs αποθηκεύτηκαν στον φάκελο: {log_folder}")
33
34     print("Η διαδικασία ολοκληρώθηκε.")
35
```

Σχήμα Β.12: Εκτέλεση Zeek για ανάλυση PCAP αρχείων και αποθήκευση logs.

Το Suricata είναι ένα ισχυρό εργαλείο ανοικτού κώδικα που χρησιμοποιείται για συστήματα ανίχνευσης εισβολών (IDS), συστήματα πρόληψης εισβολών (IPS) και ανάλυση δικτύων. Σε αντίθεση με άλλα εργαλεία ανίχνευσης δικτύου, το Suricata έχει τη δυνατότητα να αναλύει τη ροή του δικτύου σε πραγματικό χρόνο, να εντοπίζει και να καταγράφει τα χαρακτηριστικά των επιθέσεων και να παρέχει λεπτομερείς αναφορές μέσω αρχείων καταγραφής, όπως το eve.json. Στην παρούσα διατριβή, το Suricata χρησιμοποιήθηκε για τον εντοπισμό και την καταγραφή κακόβουλων δραστηριοτήτων και επιθέσεων από αρχεία PCAP, παρέχοντας κρίσιμα δεδομένα



Σχήμα B.13: *Suricata Logs output*

για την εκπαίδευση μοντέλων μηχανικής μάθησης.

Τι είναι το Suricata ;

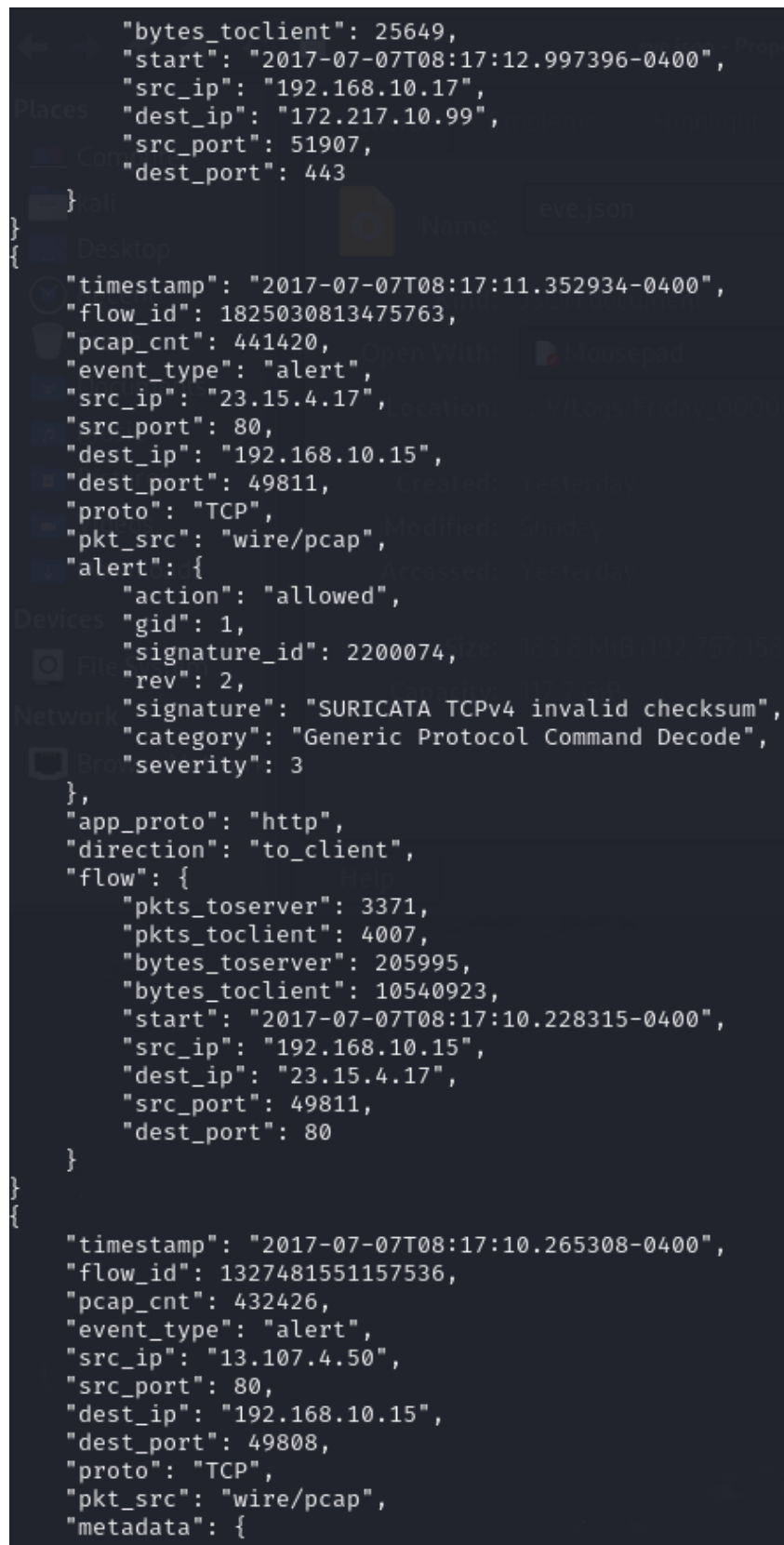
Το Suricata είναι ένα σύστημα ανίχνευσης εισβολών που βασίζεται σε υπογραφές επιθέσεων (signature-based detection), ανιχνεύοντας επιθέσεις βάσει προκαθορισμένων μοτίβων. Έχει επίσης τη δυνατότητα ανάλυσης της δικτυακής κίνησης σε πολλαπλά επίπεδα, από πρωτόκολλα δικτύου (TCP, UDP) έως πρωτόκολλα εφαρμογών όπως HTTP, DNS και TLS.

Βασικές Δυνατότητες του Suricata:

1. **Ανίχνευση σε πραγματικό χρόνο:** Το Suricata μπορεί να ανιχνεύσει και να καταγράψει απειλές σε πραγματικό χρόνο αναλύοντας την κυκλοφορία του δικτύου καθώς κινείται μεταξύ των κόμβων.
2. **Υποστήριξη πολλαπλών πρωτοκόλλων:** Το Suricata προσφέρει υποστήριξη για ανάλυση σε διάφορα επίπεδα πρωτοκόλλων, όπως:
 - **TCP και UDP** για την ανάλυση των βασικών επικοινωνιών.
 - **HTTP** για την ανάλυση της δικτυακής κίνησης.
 - **DNS** για την ανάλυση ερωτημάτων και απαντήσεων DNS.
 - **TLS** για την ανάλυση των ασφαλών συνδέσεων.

3. **Αρχεία Καταγραφής (logs):** Το Suricata παρέχει λεπτομερή αρχεία καταγραφής, κυρίως το `eve.json`, το οποίο περιέχει πληροφορίες σχετικά με τις επιθέσεις που εντοπίστηκαν, όπως:

- **IP διευθύνσεις** (πηγή και προορισμός).
- **Τύπος επίθεσης :** Καταγράφει τον τύπο της επίθεσης που αναλύθηκε.
- **Πρωτόκολλα:** Πρωτόκολλο που χρησιμοποιήθηκε κατά την επίθεση.
- **Ποσοστά πακέτων και byte** που εμπλέκονται στη ροή.



```

    "bytes_toclient": 25649,
    "start": "2017-07-07T08:17:12.997396-0400",
    "src_ip": "192.168.10.17",
    "dest_ip": "172.217.10.99",
    "src_port": 51907,
    "dest_port": 443
  },
  {
    "timestamp": "2017-07-07T08:17:11.352934-0400",
    "flow_id": 1825030813475763,
    "pcap_cnt": 441420,
    "event_type": "alert",
    "src_ip": "23.15.4.17",
    "src_port": 80,
    "dest_ip": "192.168.10.15",
    "dest_port": 49811,
    "proto": "TCP",
    "pkt_src": "wire/pcap",
    "alert": {
      "action": "allowed",
      "gid": 1,
      "signature_id": 2200074,
      "rev": 2,
      "signature": "SURICATA TCPv4 invalid checksum",
      "category": "Generic Protocol Command Decode",
      "severity": 3
    },
    "app_proto": "http",
    "direction": "to_client",
    "flow": {
      "pkts_to_server": 3371,
      "pkts_to_client": 4007,
      "bytes_to_server": 205995,
      "bytes_to_client": 10540923,
      "start": "2017-07-07T08:17:10.228315-0400",
      "src_ip": "192.168.10.15",
      "dest_ip": "23.15.4.17",
      "src_port": 49811,
      "dest_port": 80
    }
  },
  {
    "timestamp": "2017-07-07T08:17:10.265308-0400",
    "flow_id": 1327481551157536,
    "pcap_cnt": 432426,
    "event_type": "alert",
    "src_ip": "13.107.4.50",
    "src_port": 80,
    "dest_ip": "192.168.10.15",
    "dest_port": 49808,
    "proto": "TCP",
    "pkt_src": "wire/pcap",
    "metadata": {

```

Σχήμα Β.14: Ανάλυση καταγραφών Suricata από το αρχείο eve.json.

4. **Υποστήριξη υπογραφής επίθεσης:** Suricata χρησιμοποιεί μια εκτεταμένη βάση δεδομένων υπογραφών για την ανίχνευση επιθέσεων. Αυτές οι υπογραφές περιλαμβάνουν γνωστά μοτίβα κακόβουλου λογισμικού, όπως επιθέσεις DoS, έγχυση SQL και DDoS, αλλά μπορούν επίσης να επεκταθούν με προσαρμοσμένες υπογραφές για την ανίχνευση νέων τύπων επιθέσεων.

Χρήση του Suricata στην Εργασία:

Στην παρούσα διατριβή, το Suricata χρησιμοποιήθηκε για την ανάλυση αρχείων PCAP με σκοπό τον εντοπισμό κακόβουλων δραστηριοτήτων και τη δημιουργία αρχείων καταγραφής που περιέχουν πληροφορίες σχετικά με τις επιθέσεις που εντοπίστηκαν. Αυτά τα αρχεία καταγραφής ενσωματώθηκαν στο τελικό σύνολο δεδομένων που χρησιμοποιήθηκε για την εκπαίδευση των μοντέλων μηχανικής μάθησης.

- **eve.json :** Το κύριο αρχείο καταγραφής της Suricata, το οποίο περιέχει λεπτομερείς πληροφορίες σχετικά με τις ανιχνεύσεις επιθέσεων, χρησιμοποιήθηκε για την εξαγωγή κρίσιμων χαρακτηριστικών, όπως η ταυτοποίηση της διεύθυνσης IP, τα πρωτόκολλα επικοινωνίας και ο τύπος της επίθεσης που ανιχνεύθηκε.
- **Διευθύνσεις IP και πρωτόκολλα:** Τα δεδομένα από τα αρχεία καταγραφής της Suricata παρείχαν πληροφορίες σχετικά με τις διευθύνσεις IP που εμπλέκονται σε ύποπτες δραστηριότητες, καθώς και τα πρωτόκολλα επικοινωνίας που χρησιμοποιήθηκαν κατά τη διάρκεια της επίθεσης.
- **Επιθέσεις και περιστατικά:** Χρησιμοποιώντας τα δεδομένα της Suricata, ήταν δυνατόν να εντοπιστούν κακόβουλες δραστηριότητες όπως επιθέσεις DoS, ωμή βία και άλλες μορφές κακόβουλης δικτυακής κίνησης.

Προσδιορισμός Χαρακτηριστικών:

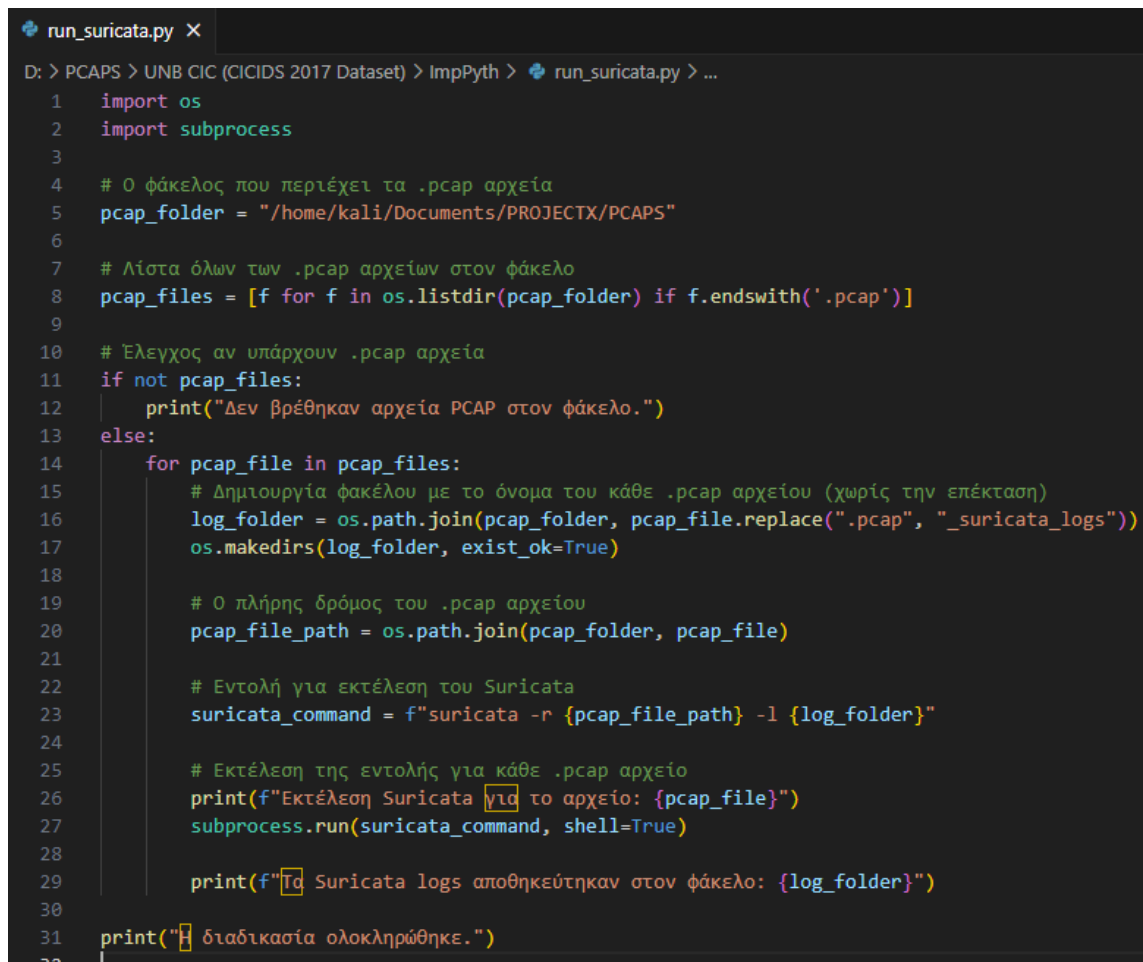
Τα δεδομένα που εξήχθησαν από το Suricata περιελάμβαναν:

- **Πηγές και προορισμοί δικτύου:** Ποιες IPs συμμετείχαν στις επιθέσεις.

- **Τύπος επίθεσης:** πληροφορίες σχετικά με την κατηγορία της επίθεσης που εντοπίστηκε, π.χ. SQL injection, DoS, DDoS κλπ.
- **Χρόνος και διάρκεια της επίθεσης:** πότε ξεκίνησε και πότε τελείωσε η επίθεση και πόση ώρα διήρκεσε η σύνδεση δικτύου.

Πλεονεκτήματα της χρήσης του Suricata:

1. **Λεπτομερής ανάλυση επιθέσεων:** Το Suricata παρέχει πλούσια δεδομένα για κάθε εντοπισμένη επίθεση, επιτρέποντας τη σε βάθος ανάλυση των επιθέσεων που πραγματοποιούνται στο δίκτυο.
2. **Υποστήριξη πολλαπλών επιπέδων:** η ανάλυση πρωτοκόλλων πολλαπλών επιπέδων επιτρέπει την ανίχνευση επιθέσεων που δεν μπορούν να εντοπιστούν μόνο με βάση τα πακέτα δικτύου.
3. **Ανίχνευση σε πραγματικό χρόνο:** Suricata είναι σε θέση να ανιχνεύει επιθέσεις σε πραγματικό χρόνο, προσφέροντας άμεσες ειδοποιήσεις και καταγραφή συμβάντων.



```
run_suricata.py X
D: > PCAPS > UNB CIC (CICIDS 2017 Dataset) > ImpPyth > run_suricata.py > ...
1 import os
2 import subprocess
3
4 # Ο φάκελος που περιέχει τα .pcap αρχεία
5 pcap_folder = "/home/kali/Documents/PROJECTX/PCAPS"
6
7 # Λίστα όλων των .pcap αρχείων στον φάκελο
8 pcap_files = [f for f in os.listdir(pcap_folder) if f.endswith('.pcap')]
9
10 # Έλεγχος αν υπάρχουν .pcap αρχεία
11 if not pcap_files:
12     print("Δεν βρέθηκαν αρχεία PCAP στον φάκελο.")
13 else:
14     for pcap_file in pcap_files:
15         # Δημιουργία φακέλου με το όνομα του κάθε .pcap αρχείου (χωρίς την επέκταση)
16         log_folder = os.path.join(pcap_folder, pcap_file.replace(".pcap", "_suricata_logs"))
17         os.makedirs(log_folder, exist_ok=True)
18
19         # Ο πλήρης δρόμος του .pcap αρχείου
20         pcap_file_path = os.path.join(pcap_folder, pcap_file)
21
22         # Εντολή για εκτέλεση του Suricata
23         suricata_command = f"suricata -r {pcap_file_path} -l {log_folder}"
24
25         # Εκτέλεση της εντολής για κάθε .pcap αρχείο
26         print(f"Εκτέλεση Suricata για το αρχείο: {pcap_file}")
27         subprocess.run(suricata_command, shell=True)
28
29         print(f"Τα Suricata logs αποθηκεύτηκαν στον φάκελο: {log_folder}")
30
31 print("Η διαδικασία ολοκληρώθηκε.")
```

Σχήμα Β.15: Εκτέλεση Suricata για ανάλυση PCAP αρχείων και αποθήκευση logs.

Το Wireshark είναι το πιο δημοφιλές εργαλείο ανάλυσης πακέτων δικτύου ανοικτού κώδικα, το οποίο χρησιμοποιείται ευρέως για την παρακολούθηση, την καταγραφή και την ανάλυση της κυκλοφορίας σε ένα δίκτυο. Το Wireshark επιτρέπει στους χρήστες να έχουν μια σε βάθος άποψη της δικτυακής κίνησης σε πραγματικό χρόνο και να αναλύουν τα πακέτα που διέρχονται από το δίκτυο σε επίπεδο bit. Στην παρούσα διπλωματική εργασία, το Wireshark χρησιμοποιήθηκε για την καταγραφή και ανάλυση δεδομένων δικτύου, τα οποία στη συνέχεια χρησιμοποιήθηκαν για την εξαγωγή στατιστικών στοιχείων και χαρακτηριστικών για την εκπαίδευση μοντέλων μηχανικής μάθησης.

Τι είναι το Wireshark;

Το Wireshark είναι ένα εργαλείο ανάλυσης πακέτων δικτύου που καταγράφει και αναλύει την κυκλοφορία που μεταφέρεται μέσω του δικτύου. Επιτρέπει στους χρήστες να προβάλλουν δεδομένα σε επίπεδο πρωτοκόλλου, όπως TCP, UDP, HTTP και DNS, καθώς και να αναλύουν κάθε πτυχή της κυκλοφορίας, από διευθύνσεις IP έως δεδομένα εφαρμογών.

Βασικές Δυνατότητες του Wireshark:

1. **Σύλληψη πακέτων:** Wireshark μπορεί να συλλάβει την κυκλοφορία του δικτύου σε πραγματικό χρόνο και να αποθηκεύσει τα δεδομένα αυτά σε αρχεία PCAP για περαιτέρω ανάλυση. Αυτά τα αρχεία περιέχουν κάθε πακέτο που μεταφέρεται μέσω του δικτύου και χρησιμοποιούνται για τη διάγνωση προβλημάτων και την ανάλυση επιθέσεων.
2. **Υποστήριξη πολλαπλών πρωτοκόλλων:** Wireshark υποστηρίζει χιλιάδες πρωτόκολλα, επιτρέποντας την ανάλυση δεδομένων που μεταφέρονται μέσω διαφόρων πρωτοκόλλων, όπως TCP, UDP, ICMP, HTTP, FTP, SSL/TLS και άλλα.
3. **Φιλτράρισμα δεδομένων:** Το Wireshark διαθέτει ισχυρές δυνατότητες φιλτραρίσματος, επιτρέποντας στους χρήστες να αναζητούν συγκεκριμένα πακέτα ή τύπους δεδομένων που θέλουν να αναλύσουν. Οι χρήστες μπορούν να χρησιμοποιούν φίλτρα, όπως διεύθυνση IP ή πρωτόκολλο, για να εστιάζουν σε συγκεκριμένα σημεία της κυκλοφορίας.
4. **Ανάλυση και οπτικοποίηση:** Το Wireshark παρέχει λεπτομερή ανάλυση πακέτων, εμφανίζοντας δεδομένα σε επίπεδο bit, και προσφέρει οπτικά εργαλεία για την κατανόηση της δικτυακής κίνησης. Επιτρέπει στους χρήστες να βλέπουν τη διαδρομή των πακέτων και τις αλληλεπιδράσεις τους μέσω γραφικών αναπαραστάσεων.
5. **Εξαγωγή δεδομένων:** Τα αρχεία PCAP που παράγονται από το Wireshark μπορούν να εξαχθούν για χρήση σε άλλες εφαρμογές ή για περαιτέρω ανάλυση, όπως ανάλυση δικτύων με εργαλεία όπως το Zeek και το Suricata ή εξαγωγή χαρακτηριστικών για μοντέλα μηχανικής μάθησης.

Χρήση του Wireshark στην Εργασία:

Στην παρούσα εργασία, το Wireshark χρησιμοποιήθηκε για την καταγραφή της δικτυακής κίνησης με τη μορφή αρχείων PCAP, τα οποία στη συνέχεια αναλύθηκαν με άλλα εργαλεία όπως το Zeek και το Suricata για την εξαγωγή χαρακτηριστικών. Η χρήση του Wireshark ήταν ζωτικής σημασίας για τη συλλογή πρωτογενών δεδομένων δικτύου, τα οποία περιελάμβαναν:

- **Πακέτα δεδομένων από επιθέσεις DoS και DDoS:** τα δεδομένα αυτά καταγράφηκαν και αναλύθηκαν για τον εντοπισμό μοτίβων κακόβουλης δραστηριότητας.
- **Συλλογή δεδομένων εφαρμογής:** Η συλλογή δεδομένων από εφαρμογές που χρησιμοποιούν πρωτόκολλα όπως HTTP, DNS και SSL/TLS ήταν απαραίτητη για την ανάλυση των επιθέσεων σε εφαρμογές.
- **Προσαρμοσμένα φίλτρα:** Το Wireshark χρησιμοποιήθηκε με προσαρμοσμένα φίλτρα για την ανίχνευση συγκεκριμένων δραστηριοτήτων και την απομόνωση των πακέτων που σχετίζονται με κακόβουλες δραστηριότητες.

Πλεονεκτήματα της χρήσης του Wireshark:

1. **Ακριβής καταγραφή δεδομένων:** Wireshark προσφέρει λεπτομερή καταγραφή των πακέτων δικτύου, επιτρέποντας την πλήρη ανάλυση της δικτυακής κίνησης. Αυτό επιτρέπει την ακριβή παρακολούθηση και ανάλυση των επιθέσεων.
2. **Ευρεία υποστήριξη πρωτοκόλλων:** Με τη δυνατότητα ανάλυσης χιλιάδων πρωτοκόλλων, το Wireshark είναι εξαιρετικά ευέλικτο και μπορεί να χρησιμοποιηθεί σε διάφορα σενάρια ανάλυσης δικτύου.
3. **Φιλτράρισμα και ανάλυση βάθους:** Μειώνοντας τον όγκο των δεδομένων που πρέπει να αναλυθούν και επιτρέποντάς σας να επικεντρωθείτε σε κρίσιμες πτυχές της δικτυακής κίνησης.

4. **Δυνατότητες αναπαραγωγής δεδομένων:** Το Wireshark επιτρέπει την αναπαραγωγή των καταγεγραμμένων πακέτων, επιτρέποντας την εκ νέου ανάλυση επιθέσεων ή ανωμαλιών και τον εντοπισμό προβλημάτων σε βάθος.
5. **Διαδραστική διεπαφή χρήστη:** Το Wireshark προσφέρει μια φιλική και διαδραστική γραφική διεπαφή χρήστη, η οποία επιτρέπει ακόμη και σε χρήστες χωρίς εξειδικευμένες γνώσεις να αναλύουν την κυκλοφορία του δικτύου.
6. **Υποστήριξη αποθήκευσης και κοινής χρήσης δεδομένων:** Τα αρχεία καταγραφής (PCAP) μπορούν να αποθηκευτούν και να διαμοιραστούν για μεταγενέστερη ανάλυση, επιτρέποντας την απομακρυσμένη συνεργασία και τον έλεγχο των δεδομένων από διαφορετικούς χρήστες ή ομάδες.
7. **Ανοιχτός κώδικας και η Μεγάλη Κοινότητα:** με συνεχή ανάπτυξη και υποστήριξη από μια μεγάλη κοινότητα χρηστών και προγραμματιστών, εξασφαλίζοντας τη συνεχή βελτίωση και αναβάθμισή του.
8. **Ενσωματωμένα εργαλεία στατιστικής ανάλυσης:** Το Wireshark προσφέρει εργαλεία για τη δημιουργία στατιστικών αναφορών, τη γραφική απεικόνιση της δικτυακής κίνησης και τη συγκριτική ανάλυση διαφόρων παραμέτρων πακέτων δικτύου.

Friday-WorkingHours.pcap

FileEditViewGoCaptureAnalyzeStatisticsTelephonyWirelessToolsHelp



Apply a display filter ... <Ctrl-/>

Time	Source	Destination	Protocol	Length	Info
0.000000	Cisco_id:bb:04	CDP/VTP/DTP/PagP/UD...	CDP	457	Device ID: SWCL Port ID: GigabitEthernet1/0/4
3.419202	Cisco_id:bb:0a	CDP/VTP/DTP/PagP/UD...	CDP	458	Device ID: SWCL Port ID: GigabitEthernet1/0/10
6.199352	Cisco_id:bb:08	CDP/VTP/DTP/PagP/UD...	CDP	457	Device ID: SWCL Port ID: GigabitEthernet1/0/8
6.632123	Cisco_id:bb:05	CDP/VTP/DTP/PagP/UD...	CDP	457	Device ID: SWCL Port ID: GigabitEthernet1/0/5
8.799783	Cisco_id:bb:11	CDP/VTP/DTP/PagP/UD...	CDP	458	Device ID: SWCL Port ID: GigabitEthernet1/0/17
10.716067	192.168.10.50	192.168.10.3	LDAP	469	SASL GSS-API Privacy: payload (352 bytes)
10.716072	192.168.10.50	192.168.10.3	TCP	469	[TCP Retransmission] 56108 → 3268 [PSH, ACK] Seq=1 Ack=1 Win=0 Len=0
10.716331	192.168.10.3	192.168.10.50	LDAP	138	SASL GSS-API Privacy: payload (23 bytes)
10.716335	192.168.10.3	192.168.10.50	TCP	138	[TCP Retransmission] 3268 → 56108 [PSH, ACK] Seq=1 Ack=404 Win=0 Len=0
10.716413	192.168.10.50	192.168.10.3	TCP	66	56108 → 3268 [ACK] Seq=404 Ack=73 Win=377 Len=0 TSval=134183
10.716417	192.168.10.50	192.168.10.3	TCP	66	[TCP Dup ACK 10#1] 56108 → 3268 [ACK] Seq=404 Ack=73 Win=377
10.717145	192.168.10.50	192.168.10.3	LDAP	469	SASL GSS-API Privacy: payload (352 bytes)
10.717149	192.168.10.50	192.168.10.3	TCP	469	[TCP Retransmission] 42144 → 389 [PSH, ACK] Seq=1 Ack=1 Win=0 Len=0
10.717398	192.168.10.3	192.168.10.50	LDAP	382	SASL GSS-API Privacy: payload (265 bytes)
10.717403	192.168.10.3	192.168.10.50	TCP	382	[TCP Retransmission] 389 → 42144 [PSH, ACK] Seq=1 Ack=404 Win=0 Len=0
10.717428	192.168.10.50	192.168.10.3	TCP	66	42144 → 389 [ACK] Seq=404 Ack=317 Win=978 Len=0 TSval=134183
10.717430	192.168.10.50	192.168.10.3	TCP	66	[TCP Dup ACK 16#1] 42144 → 389 [ACK] Seq=404 Ack=317 Win=978
11.040842	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040853	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040855	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040857	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040859	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040861	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040863	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040865	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
11.040867	fe80::6d07:ea13:6f7... ff02::1:2	DHCPv6	165	Solicit XID: 0xd03c79 CID: 0001000120d46426b8ac6f3608f5	
14.401005	Cisco_id:bb:03	CDP/VTP/DTP/PagP/UD...	CDP	457	Device ID: SWCL Port ID: GigabitEthernet1/0/3
27.114808	192.168.10.50	192.168.10.3	LDAP	469	SASL GSS-API Privacy: payload (352 bytes)
27.114813	192.168.10.50	192.168.10.3	TCP	469	[TCP Retransmission] 56108 → 3268 [PSH, ACK] Seq=404 Ack=73 Win=0 Len=0
27.115091	192.168.10.3	192.168.10.50	LDAP	138	SASL GSS-API Privacy: payload (23 bytes)
27.115095	192.168.10.3	192.168.10.50	TCP	138	[TCP Retransmission] 3268 → 56108 [PSH, ACK] Seq=73 Ack=807 Win=0 Len=0
27.115170	192.168.10.50	192.168.10.3	TCP	66	56108 → 3268 [ACK] Seq=807 Ack=145 Win=377 Len=0 TSval=13422
27.115174	192.168.10.50	192.168.10.3	TCP	66	[TCP Dup ACK 32#1] 56108 → 3268 [ACK] Seq=807 Ack=145 Win=377
27.115731	192.168.10.50	192.168.10.3	LDAP	469	SASL GSS-API Privacy: payload (352 bytes)

Frame 1: 457 bytes on wire (3656 bits), 457 bytes captured (3656) on interface 0

▶ IEEE 802.3 Ethernet

▶ Logical-Link Control

▶ Cisco Discovery Protocol

0000	01 00 0c cc cc cc 70 6e 6d 1d bb 04 01 bb aa aa	pn
0010	03 00 00 0c 20 00 02 b4 81 47 00 01 00 08 53 57	
0020	43 4c 00 05 01 09 43 69 73 63 6f 20 49 4f 53 20	CL.....Ci
0030	53 6f 66 74 77 61 72 65 2c 20 43 61 74 61 6c 79	Software
0040	73 74 20 4c 33 20 53 77 69 74 63 68 20 53 6f 66	st L3 Sw
0050	74 77 61 72 65 20 28 43 41 54 33 4b 5f 43 41 41	ware (C
0060	2d 55 4e 49 56 45 52 53 41 4c 4b 39 2d 4d 29 2c	-UNIVERS

Σχήμα Β.16: Wireshark ανάλυση πακέτων, προβολή δικτυακής δραστηριότητας.

Η προεπεξεργασία των δεδομένων ήταν επίσης ένα κρίσιμο βήμα στην ανάπτυξη του συστήματος ανίχνευσης εισβολών. Η διαδικασία περιελάμβανε την αφαίρεση των περιττών χαρακτηριστικών που δε συμβάλλουν στην απόδοση των μοντέλων, την κανονικοποίηση των δεδομένων για να εξασφαλιστεί η ομοιομορφία στις τιμές των χαρακτηριστικών και την αντιμετώπιση κενών ή λανθασμένων τιμών στα δεδομένα. Μετά την ολοκλήρωση της προεπεξεργασίας, τα δεδομένα χωρίστηκαν σε σύνολα εκπαίδευσης και δοκιμής για την αξιολόγηση της απόδοσης των μοντέλων.

Η εκπαίδευση του μοντέλου πραγματοποιήθηκε με αλγόριθμους μηχανικής μάθησης όπως οι αλγόριθμοι Random Forest, Support Vector Machines (SVM) και Neural Networks. Το Random Forest χρησιμοποιήθηκε για να συνδυάσει πολλαπλά δέντρα αποφάσεων για την

ανίχνευση επιθέσεων με βάση πολλαπλά χαρακτηριστικά, ενώ τα SVM ταξινομούν τα δεδομένα με βάση την απόσταση από ένα υπερεπίπεδο. Τα νευρωνικά δίκτυα, από την άλλη πλευρά, μαθαίνουν από τα δεδομένα και χρησιμοποιούνται για την ανίχνευση επιθέσεων με βάση μοτίβα κίνησης.

Επιλογή Αλγόριθμων Μηχανικής Μάθησης

Η επιλογή του κατάλληλου αλγορίθμου μηχανικής μάθησης είναι ζωτικής σημασίας για την επιτυχία του συστήματος ανίχνευσης εισβολών και εξαρτάται από τα χαρακτηριστικά των διαθέσιμων δεδομένων καθώς και από τις απαιτήσεις του έργου. Οι πιο διαδεδομένοι αλγόριθμοι που χρησιμοποιούνται για την ανίχνευση εισβολών θα συζητηθούν παρακάτω.

1. **Radom Forest:** Είναι ένας αλγόριθμος μηχανικής μάθησης που βασίζεται σε ένα σύνολο δέντρων απόφασης. Ο αλγόριθμος παράγει πολλά δέντρα αποφάσεων κατά την εκπαίδευση και για κάθε πρόβλεψη λαμβάνεται η απόφαση της πλειοψηφίας των δέντρων. Η προσέγγιση αυτή βοηθά στη μείωση του προβλήματος της υπερπροσαρμογής (overfitting), το οποίο συχνά εμφανίζεται μεμονωμένα στα δέντρα αποφάσεων. Το Random Forest έχει χρησιμοποιηθεί εκτενώς για την ανίχνευση εισβολών, καθώς μπορεί να χειριστεί μεγάλο όγκο δεδομένων, να ενσωματώσει πολλά χαρακτηριστικά και να διαχωρίσει αποτελεσματικά τα δεδομένα μεταξύ κανονικών και κακόβουλων δραστηριοτήτων.

Random Forest Classifier

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Φόρτωση δεδομένων
data = pd.read_csv('dataset.csv')

# Διαίρεση χαρακτηριστικών και στόχου
X = data.drop('label', axis=1)
y = data['label']

# Διαίρεση δεδομένων σε σύνολα εκπαίδευσης και δοκιμής
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=42)

# Εκπαίδευση Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Πρόβλεψη και αξιολόγηση
y_pred = clf.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Python

Σχήμα Β.17: Εκπαίδευση Random Forest, πρόβλεψη και αξιολόγηση ακρίβειας.

2. **Μηχανή διανύσματος υποστήριξης (SVM):** Ο αλγόριθμος Μηχανή διανύσματος υποστήριξης (SVM) είναι ιδανικός για την ταξινόμηση δεδομένων, ιδίως όταν τα δεδομένα είναι διαχωρίσιμα. Η SVM χρησιμοποιεί ένα υπερεπίπεδο για τον διαχωρισμό των δεδομένων σε κατηγορίες. Στόχος είναι η μεγιστοποίηση της απόστασης μεταξύ του υπερεπιπέδου και των σημείων δεδομένων των δύο κατηγοριών. Αυτό σημαίνει ότι το SVM μπορεί να διαχειριστεί αποτελεσματικά δεδομένα που παρουσιάζουν σαφή όρια μεταξύ κανονικών και κακόβουλων ροών. Ωστόσο, για πιο σύνθετα δεδομένα που δεν μπορούν να διαχωριστούν γραμμικά, το SVM μπορεί να επεκταθεί με την εφαρμογή συναρτήσεων πυρήνα για τον εντοπισμό πιο σύνθετων σχέσεων μεταξύ των δεδομένων.

Support Vector Machine (SVM)

```
from sklearn.svm import SVC

# Εκπαίδευση SVM
svm = SVC(kernel='linear', random_state=42)
svm.fit(X_train, y_train)

# Πρόβλεψη και αξιολόγηση
y_pred = svm.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Python

Σχήμα Β.18: Εκπαίδευση SVM, πρόβλεψη και αξιολόγηση ακρίβειας.

3. **Νευρωνικά Δίκτυα:** Τα Νευρωνικά Δίκτυα είναι ένας από τους πιο ισχυρούς αλγόριθμους μηχανικής μάθησης και χρησιμοποιούνται ευρέως για την ανίχνευση εισβολών. Λειτουργούν με τη χρήση πολλαπλών επιπέδων νευρώνων, τα οποία επεξεργάζονται δεδομένα και εντοπίζουν σύνθετα μοτίβα. Η ικανότητα αυτή τα καθιστά ιδιαίτερα αποτελεσματικά για την ανάλυση μεγάλων και πολύπλοκων δεδομένων, όπου τα παραδοσιακά μοντέλα μπορεί να αποτύχουν να αποτυπώσουν τις σχέσεις μεταξύ των χαρακτηριστικών. Τα νευρωνικά δίκτυα είναι κατάλληλα για καταστάσεις όπου απαιτείται ανίχνευση επιθέσεων σε πολύπλοκα περιβάλλοντα, όπως τα δίκτυα IoT, όπου οι επιθέσεις μπορεί να είναι πολυδιάστατες και δυναμικές.

Neural Network

```
from sklearn.neural_network import MLPClassifier

# Εκπαίδευση Neural Network
mlp = MLPClassifier(hidden_layer_sizes=(100,), max_iter=300, random_state=42)
mlp.fit(X_train, y_train)

# Πρόβλεψη και αξιολόγηση
y_pred = mlp.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Python

Σχήμα Β.19: Εκπαίδευση MLP νευρωνικού, αξιολόγηση με ακρίβεια.

4. **K-Nearest Neighbors (KNN):** Ο αλγόριθμος K-Nearest Neighbors (KNN) είναι ένας απλός αλλά αποτελεσματικός αλγόριθμος που βασίζεται στην ταξινόμηση δεδομένων με βάση τους πλησιέστερους γείτονες. Η βασική ιδέα πίσω από τον KNN είναι ότι ένα δεδομένο ανήκει στην κατηγορία δεδομένων που έχει τις πλησιέστερες τιμές χαρακτηριστικών. Ο KNN λειτουργεί καλά σε μικρά σύνολα δεδομένων και όταν τα χαρακτηριστικά είναι καλά καθορισμένα. Ωστόσο, μπορεί να γίνει υπολογιστικά δαπανηρή σε μεγάλα σύνολα δεδομένων και να απαιτεί σημαντικό χρόνο για την επεξεργασία νέων δεδομένων. Παρόλα αυτά, μπορεί να χρησιμοποιηθεί αποτελεσματικά σε συνδυασμό με άλλους αλγορίθμους για την ανίχνευση επιθέσεων.

Κάθε ένας από αυτούς τους αλγορίθμους έχει διαφορετικά πλεονεκτήματα και μειονεκτήματα και η επιλογή του κατάλληλου εξαρτάται από την πολυπλοκότητα των δεδομένων, τις απαιτήσεις του συστήματος και τους διαθέσιμους πόρους για την εκπαίδευση των μοντέλων. Στην παρούσα εργασία δοκιμάστηκαν διάφοροι αλγόριθμοι για να εντοπιστεί αυτός που παρέχει την καλύτερη απόδοση στην ανίχνευση εισβολών, διατηρώντας παράλληλα χαμηλό ποσοστό ψευδώς θετικής και αρνητικής ανίχνευσης.

Για την αξιολόγηση των μοντέλων χρησιμοποιήθηκαν διάφορες μετρικές, όπως η ακρίβεια (accuracy), η ανάκληση (recall) και precision, οι οποίες μετρούν την ακρίβεια των ανιχνεύσεων και την ικανότητα του μοντέλου να αναγνωρίζει πραγματικές επιθέσεις. Επιπλέον, η καμπύλη ROC και η AUC χρησιμοποιήθηκαν για την αξιολόγηση της απόδοσης των μοντέλων σε διαφορε-

τικά επίπεδα ευαισθησίας.

Χρησιμοποιώντας αυτές τις τεχνικές, τα μοντέλα μηχανικής μάθησης αξιολογήθηκαν με επιτυχία και απέδειξαν την ικανότητά τους να ανιχνεύουν κακόβουλες δραστηριότητες σε δίκτυα υπολογιστών. Τα αποτελέσματα των μοντέλων δείχνουν υψηλή απόδοση και η μεθοδολογία αυτή μπορεί να χρησιμοποιηθεί για την περαιτέρω βελτίωση των συστημάτων ανίχνευσης εισβολών.

Κεφάλαιο C

Κακόβουλες Επιθέσεις και οι βασικοί αλγόριθμοι τους

DoS και DDoS :

Οι επιθέσεις DoS και DDoS στοχεύουν στην υπερφόρτωση ενός server με μεγάλο αριθμό αιτήσεων ή πακέτων, καθιστώντας τον μη διαθέσιμο στους νομίμους χρήστες. Στην περίπτωση μιας DoS επιθέσεις ο επιτιθέμενος στέλνει συνεχώς αιτήσεις ή πακέτα σε ένα server, επιδιώκοντας να εξαντλήσει τους πόρους του συστήματος. Από την άλλη πλευρά, στην επίθεση DDoS πολλοί επιτιθέμενοι η μολυσμένα συστήματα, γνωστά ως botnets, εκτελούν επιθέσεις ταυτόχρονα, γεγονός που καθιστά πιο δύσκολη την ανίχνευση και την άμυνα.

Οι επιθέσεις αυτές βασίζονται σε τεχνικές όπως το flooding, όπου αποστέλλεται μεγάλος όγκος δεδομένων στον στόχο, και το amplification, που εκμεταλλεύεται τα ελαττώματά πρωτοκόλλων για να αυξήσει την επίδραση της επίθεσης. Τα botnets διαδραματίζουν επίσης σημαντικό ρόλο καθώς χρησιμοποιούν μολυσμένα συστήματα για την ταυτόχρονη εκτέλεση της επίθεσης. Για την ανίχνευση αυτών των επιθέσεων, σημαντικά χαρακτηριστικά περιλαμβάνουν τον ασυνήθιστα υψηλό αριθμό πακέτων σε σύντομο χρονικό διάστημα, τον μεγάλο όγκο εισερχόμενων και εξερχόμενων δεδομένων, καθώς και τις αιτήσεις από πολλές διαφορετικές διευθύνσεις IP.



```

DoS Attack με Hping3

#!/bin/bash
# Script για DoS επίθεση με Hping3

TARGET_IP="192.168.1.100"
TARGET_PORT="80"
PACKET_SIZE="1200"

# Εκτέλεση DoS επίθεσης
hping3 -S --flood -V -p $TARGET_PORT -d $PACKET_SIZE $TARGET_IP

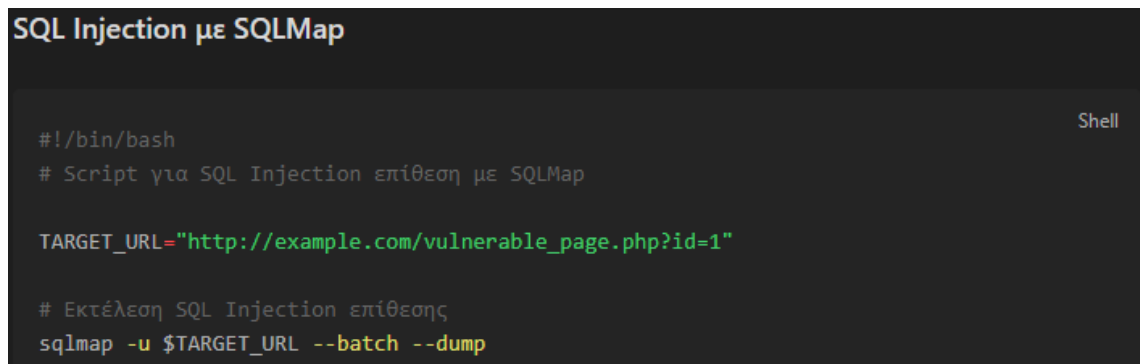
```

Σχήμα C.20: Εκτέλεση DoS επίθεσης με Hping3 σε συγκεκριμένη διεύθυνση IP και πόρτα.

SQL Injection: Έπειτα έχουμε την τεχνική SQL Injection εκμεταλλεύεται τα κενά ασφαλείας σε εφαρμογές web που χρησιμοποιούν βάσεις δεδομένων. Ο επιτιθέμενος εισάγει κακόβουλο κώδικα SQL σε ένα πεδίο εισαγωγής δεδομένων της εφαρμογής, επιτρέποντας του α εκτελέσει ανεπιθύμητες ενέργειες στη βάση δεδομένων, όπως εξαγωγή, τροποποίηση η διαγραφή δεδομένων. Υπάρχουν διάφορες τεχνικές SQL injection όπως η τεχνική Union-Based SQL Injection όπου ο επιτιθέμενος χρησιμοποιεί το UNION operator για να επιστρέφει επιπλέον δεδομένα και η τεχνική Error-based SQL Injection η οποία προκαλεί λάθη τα οποία αποκαλύπτουν πληροφορίες της βάση δεδομένων. Έπειτα στην περίπτωση της τεχνικής Blind SQL Injection οι επιθέσεις γίνονται με έμμεσο τρόπο έτσι ώστε να μην επιστρέφονται άμεσα δεδομένα. Σημαντικό εργαλείο για να επιτευχθούν αυτά είναι SQLMap¹

Χαρακτηριστικά για την ανίχνευση αυτών των επιθέσεων περιλαμβάνουν την παρουσία SQL λέξεων κλειδιών π.χ. SELECT UNION DROP στα πεδία εισαγωγής δεδομένων, την αυξημένη συχνότητα αποτυχημένων αιτήσεων και τα σφάλματά που προέρχονται από τη βάση δεδομένων.

¹<https://sqlmap.org/>



```

SQL Injection με SQLMap

#!/bin/bash
# Script για SQL Injection επίθεση με SQLMap

TARGET_URL="http://example.com/vulnerable_page.php?id=1"

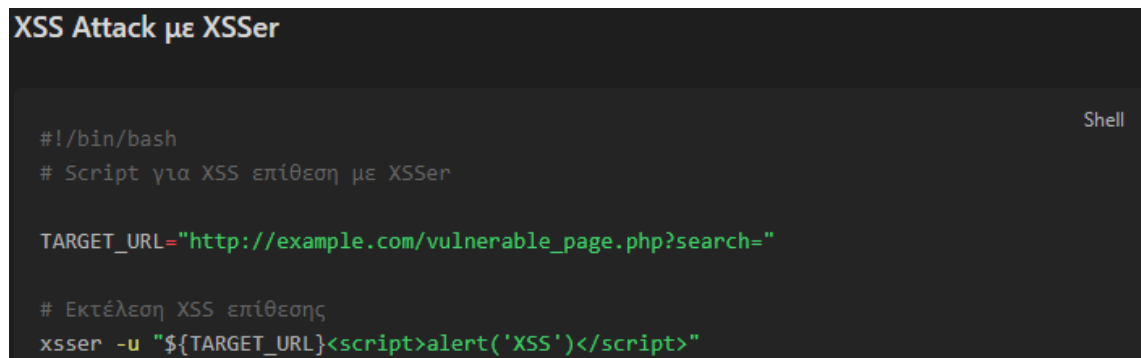
# Εκτέλεση SQL Injection επίθεσης
sqlmap -u $TARGET_URL --batch --dump
  
```

Σχήμα C.21: Εκτέλεση SQL Injection επίθεσης με SQLMap σε ευπαθή ιστοσελίδα.

Cross-Site Scripting (XSS):

Η επίθεση XSS συμβαίνει όταν ένας επιτιθέμενος εισάγει κακόβουλο κώδικά JavaScript ή HTML σε μια εφαρμογή web και ο κώδικας αυτός εκτελείται από τον browser του χρήστη. Υπάρχουν διάφοροι τύποι XSS επιθέσεων, μερικοί από αυτούς είναι Stored XSS όπου ο κακόβουλος κώδικας αποθηκεύεται στον Server και εκτελείται κάθε φορά που φορτώνει η σελίδα, ο Reflected XSS στον οποίο ο κακόβουλος κώδικας περιλαμβάνεται σε μια URL και εκτελείται κατά την επίσκεψη του χρήστη και ο DOM-based XSS όπου ο κώδικας εκτελείται μέσω του Document Object Model (DOM) του browser.

Για την ανίχνευση XSS επιθέσεων υπάρχουν κάποια χαρακτηριστικά που προδίδουν την παρουσία κακόβουλων script σε πεδία εισαγωγής δεδομένων ή URLs, την ανίχνευση ασυνήθιστης δραστηριότητας από τον browser (π.χ. εκτέλεση script χωρίς να το έχει ζητήσει ο χρήστης) καθώς και αναφορές από εργαλεία ασφάλειας για XSS επιθέσεις.



```

XSS Attack με XSSer

#!/bin/bash
# Script για XSS επίθεση με XSSer

TARGET_URL="http://example.com/vulnerable_page.php?search="

# Εκτέλεση XSS επίθεσης
xsser -u "${TARGET_URL}<script>alert('XSS')</script>"
  
```

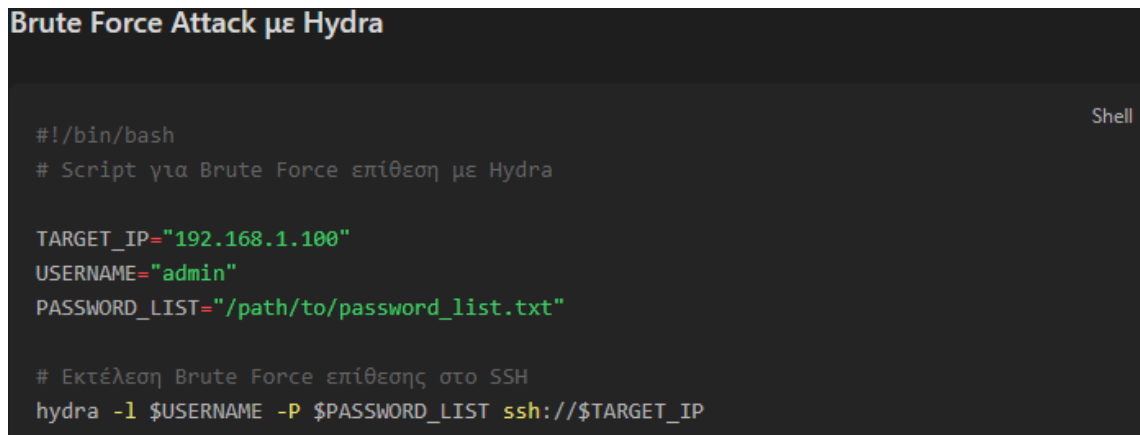
Σχήμα C.22: Εκτέλεση XSS επίθεσης με XSSer σε ευπαθή ιστοσελίδα.

Brute Force Attacks:

Στις επιθέσεις Brute Force ο επιτιθέμενος δοκιμάζει ασυνεχώς συνδυασμούς username και password μέχρι να βρει τον σωστό συνδυασμό ώστε να προσβάλει ένα σύστημα. Υπάρχουν διάφορες μορφές επιθέσεων Brute force όπως dictionary attacks όπου χρησιμοποιούνται λίστες με συνηθισμένους κωδικούς πρόσβασης και το credential stuffing στο οποίο χρησιμοποιούνται κλεμμένα credentials από άλλες παραβιάσεις ασφάλειας. Σημαντικό εργαλείο για να επιτευχθούν αυτά είναι Hydra²

Για την ανίχνευση αυτών των επιθέσεων κοιτάμε τα εξής, τα χαρακτηριστικά περιλαμβάνουν αυξημένο αριθμό αποτυχημένων προσπαθειών σύνδεσης, πολλαπλές αιτήσεις σύνδεσής από την ίδια διεύθυνση IP ή σε σύντομο χρονικό διάστημα και την παρουσία μοτίβων που δείχνουν αυτοματοποιημένες προσπάθειες σύνδεσης.

²<https://github.com/vanhauser-thc/thc-hydra>



```

Brute Force Attack με Hydra

#!/bin/bash
# Script για Brute Force επίθεση με Hydra

TARGET_IP="192.168.1.100"
USERNAME="admin"
PASSWORD_LIST="/path/to/password_list.txt"

# Εκτέλεση Brute Force επίθεσης στο SSH
hydra -l $USERNAME -P $PASSWORD_LIST ssh://$TARGET_IP
  
```

Σχήμα C.23: Εκτέλεση Brute Force επίθεσης με Hydra σε SSH.

Port Scanning:

Το Port Scanning είναι μια τεχνική που χρησιμοποιείτε από τους επιτιθέμενους για να σαρώσουν τις θύρες ενός συστήματος και να εντοπίζουν ποιες από αυτές είναι ανοιχτές και αποτελούν πιθανά κενά ασφαλείας. Ο επιτιθέμενος αποστέλλει SYN πακέτα για να προσδιορίσει αν οι θύρες είναι ανοιχτές (SYN Scan), πραγματοποιεί πλήρες σύνδεσή TCP για να επιβεβαιώσει την κατάσταση της θύρας (TCP Connect Scan) ή χρησιμοποιεί τεχνικές που παρακάμπτουν τα συστήματα ασφαλείας για να παραμένει ανιχνεύσιμο (Stealth Scanning). Σημαντικό εργαλείο για να επιτευχθούν αυτά είναι Nmap³

Για την ανίχνευση Port Scanning διακρίνουμε, χαρακτηριστικά που δείχνουν την παρουσία πολλών SYN πακέτων σε διαφορετικές θύρες σε σύντομο χρονικό διάστημα, ανώμαλα μοτίβα δικτυακής κυκλοφορίας που υποδεικνύουν σάρωση θύρας, καθώς και αναφορές από συστήματα ανίχνευσης εισβολών που ανιχνεύουν τέτοιες δραστηριότητες.

³<https://nmap.org/>

```
Port Scanning με Nmap
```

```
#!/bin/bash
# Script για Port Scanning με Nmap

TARGET_IP="192.168.1.100"

# Εκτέλεση πλήρους port scan
nmap -p- $TARGET_IP
```

Shell

Σχήμα C.24: Εκτέλεση port scanning με Nmap σε συγκεκριμένη διεύθυνση IP.

Κεφάλαιο D

Προεπεξεργασία Δεδομένων

Καθαρισμός Δεδομένων

Αφαίρεση Περιττών Πακέτων

Ο καθαρισμός των δεδομένων είναι ένα από τα πιο σημαντικά βήματα στην ανάλυση πακέτων δικτύου, διασφαλίζοντας ότι τα δεδομένα που πρόκειται να υποβληθούν σε επεξεργασία είναι κατάλληλα και χρήσιμα για τη συγκεκριμένη ανάλυση. Ένα κρίσιμο βήμα σε αυτή τη διαδικασία είναι η αφαίρεση των πακέτων που δεν παρέχουν σημαντικές πληροφορίες. Ένα από τα εργαλεία που χρησιμοποιούνται ευρέως για τον σκοπό αυτό είναι το **Wireshark**¹, το οποίο επιτρέπει την εξέταση και το φιλτράρισμα των δεδομένων δικτύου. Πρώτο βήμα είναι να ανοίξετε το Wireshark και να φορτώσετε το αρχείο PCAP που περιέχει τη λήψη πακέτων. Χρησιμοποιώντας φίλτρα εμφάνισης, μπορείτε να εστιάσετε μόνο σε πακέτα που σχετίζονται με συγκεκριμένες επιθέσεις ή πρωτόκολλα. Για παράδειγμα, εάν η ανάλυση αφορά πακέτα HTTP, μπορεί να εφαρμοστεί το φίλτρο 'http' για να προβάλονται μόνο αυτά τα πακέτα, αποφεύγοντας τον χειρισμό μη χρήσιμων δεδομένων. Στη συνέχεια, μέσα από την περιοχή «Packet List», μπορείτε να επιλέξετε τα πακέτα που θέλετε να κρατήσετε και να χρησιμοποιήσετε την επιλογή File Export Specified Packets, η οποία σας επιτρέπει να αποθηκεύσετε μόνο τα σχετικά πακέτα σε ένα νέο αρχείο.

¹<https://www.wireshark.org/>

Σε αυτό το στάδιο, είναι εξίσου σημαντικό να αφαιρεθούν τα πακέτα που δε συμβάλλουν στην ανάλυση, όπως πακέτα ARP ή άλλα πρωτόκολλα που δε σχετίζονται με τη μελέτη της επίθεσης. Η αφαίρεση αυτών των πακέτων μειώνει τον όγκο των δεδομένων που υποβάλλονται σε επεξεργασία και επιταχύνει τις διαδικασίες ανάλυσης.

Ο καθαρισμός των δεδομένων είναι ένα κρίσιμο βήμα στη διαδικασία εκπαίδευσης μοντέλων μηχανικής μάθησης, ειδικά όταν αναλύονται οι καθαρές ροές και τα στατιστικά στοιχεία ροής. Σε σύνολα δεδομένων που περιέχουν πληροφορίες από διαφορετικές πηγές (π.χ. Zeek, Suricata, NFStream), τα δεδομένα είναι συχνά ακατάστατα ή έχουν ελλιπείς τιμές, διπλές εγγραφές ή λανθασμένα χαρακτηριστικά, τα οποία μπορούν να επηρεάσουν αρνητικά την απόδοση των μοντέλων μηχανικής μάθησης.

Γιατί ο καθαρισμός δεδομένων είναι σημαντικός:

- **Ακρίβεια των μοντέλων:** ο καθαρισμός εξασφαλίζει ότι τα μοντέλα θα εκπαιδευτούν σε καθαρά και ακριβή δεδομένα. Τα δεδομένα που περιέχουν σφάλματα ή ελλιπή δεδομένα μπορούν να οδηγήσουν σε παραπλανητικά αποτελέσματα και ανακρίβειες στην ανίχνευση εισβολών.
- **Αξιοπιστία και αποτελεσματικότητα:** Τα δεδομένα που περιέχουν θόρυβο ή περιττές πληροφορίες υπερφορτώνουν το μοντέλο και μειώνουν την αποτελεσματικότητά του. Ο καθαρισμός αφαιρεί τις περιττές εγγραφές, βελτιώνοντας την αξιοπιστία του μοντέλου.
- **Εξάλειψη των μην αντιπροσωπευτικών δεδομένων:** Οι διπλές εγγραφές ή τα μην αντιπροσωπευτικά δεδομένα μπορούν να αλλοιώσουν τα αποτελέσματα. Ο καθαρισμός βοηθά στη διατήρηση των πιο σχετικών και ακριβών πληροφοριών, βελτιώνοντας την ποιότητα των προβλέψεων.

Πώς ο καθαρισμός είναι χρήσιμος στα Netflows - Flow Statistics:

Τα στατιστικά στοιχεία ροής που εξάγονται από εργαλεία όπως το NFStream και το CFlowMeter περιέχουν διάφορα χαρακτηριστικά που σχετίζονται με τη δραστηριότητα του δικ-

τύου, όπως ο αριθμός των πακέτων, ο όγκος δεδομένων, οι διευθύνσεις IP και τα πρωτόκολλα. Αυτά τα χαρακτηριστικά είναι κρίσιμα για τον εντοπισμό μοτίβων που σχετίζονται με κακόβουλες δραστηριότητες.

Η εκκαθάριση δεδομένων σε αυτό το πλαίσιο εξασφαλίζει ότι:

- Απομακρύνετε λανθασμένες ή ελλιπείς ροές (π.χ. άκυρα πακέτα ή ροές χωρίς επαρκείς πληροφορίες για ανάλυση).
- Διατηρούνται μόνο οι σημαντικές ροές που περιέχουν αρκετές πληροφορίες για την ανίχνευση εισβολών, π.χ. επιθέσεις DoS ή DDoS.
- Τα χαρακτηριστικά κανονικοποιούνται ώστε να βρίσκονται σε κοινή κλίμακα, γεγονός που συμβάλλει στη βελτίωση της απόδοσης του μοντέλου, ιδίως σε αλγόριθμους που βασίζονται στην απόσταση (π.χ. KNN).

Συνολικά, ο καθαρισμός είναι ένα αναπόσπαστο βήμα που συμβάλλει στην επιτυχή εκπαίδευση και την αποτελεσματική ανίχνευση εισβολών από μοντέλα μηχανικής μάθησης.

Αφαίρεση Επαναλαμβανόμενων Πακέτων

Η αφαίρεση των επαναλαμβανόμενων πακέτων είναι ένα σημαντικό βήμα, καθώς η ύπαρξή τους μπορεί να αλλοιώσει τις στατιστικές αναλύσεις και να οδηγήσει σε ανακρίβειες κατά την εκπαίδευση των μοντέλων μηχανικής μάθησης. Για την αφαίρεση των επαναλαμβανόμενων πακέτων μπορούν να χρησιμοποιηθούν διάφορα εργαλεία. Το 'tcpdump', για παράδειγμα, επιτρέπει την απόρριψη ενός σταθερού αριθμού πακέτων μέσω της επιλογής '-c', βοηθώντας στη μείωση του αριθμού των επαναλαμβανόμενων εγγραφών. Επιπλέον, το εργαλείο 'editcap', το οποίο αποτελεί μέρος του πακέτου Wireshark, μπορεί να χρησιμοποιηθεί για την αφαίρεση των επαναλαμβανόμενων πακέτων από τα αρχεία PCAP, δημιουργώντας ένα καθαρότερο σύνολο δεδομένων.

Για να αφαιρέσετε επαναλαμβανόμενα πακέτα με το 'editcap', μπορείτε να χρησιμοποιήσετε την ακόλουθη εντολή:

Listing D.1: Συμπίεση ενός μεγάλου αρχείου PCAP με χρήση editcap

```
1 editcap -d input.pcap output.pcap
```

Αυτή η εντολή φιλτράρει τα διπλότυπα πακέτα από το αρχείο 'input.pcap' και δημιουργεί ένα νέο, καθαρό αρχείο 'output.pcap', απαλλαγμένο από διπλότυπα. Αυτή η διαδικασία διασφαλίζει ότι το αρχείο PCAP που χρησιμοποιείται στην ανάλυση δεν περιέχει επαναλήψεις που θα μπορούσαν να επηρεάσουν αρνητικά τα αποτελέσματα της ανάλυσης και τη συμπεριφορά των μοντέλων μηχανικής μάθησης.

Αφαίρεση Θορύβου

Εξάλειψη μη σχετικών δεδομένων

Η διαδικασία απομάκρυνσης του θορύβου είναι απαραίτητη σε κάθε ανάλυση δικτύου, καθώς διασφαλίζει ότι τα δεδομένα που χρησιμοποιούνται είναι καθαρά και επικεντρωμένα στις συγκεκριμένες ανάγκες της ανάλυσης. Ο θόρυβος στα δεδομένα μπορεί να προέρχεται από άσχετες πληροφορίες, όπως πακέτα που δεν έχουν καμία σχέση με τις επιθέσεις που μελετώνται ή κίνηση που δεν ταιριάζει με τα πρωτόκολλα ενδιαφέροντος. Η ύπαρξη τέτοιων άσχετων δεδομένων μπορεί να οδηγήσει σε παραπλανητικά αποτελέσματα και να αυξήσει σημαντικά το φορτίο επεξεργασίας και ανάλυσης, καθιστώντας επιτακτική την ανάγκη προσεκτικού φιλτραρίσματος τους.

Αρχικά, η απομόνωση των δεδομένων που σχετίζονται με την κίνηση δικτύου που μας ενδιαφέρει γίνεται με τη χρήση εξειδικευμένων φίλτρων. Τα φίλτρα αυτά επιτρέπουν την εστίαση σε συγκεκριμένους τύπους πακέτων, όπως αυτά που σχετίζονται με συγκεκριμένα πρωτόκολλα ή επιθέσεις, απομονώνοντάς τα από το γενικό θόρυβο. Για παράδειγμα, εάν η ανάλυση εστιάζει σε επιθέσεις που χρησιμοποιούν το πρωτόκολλο HTTP, τότε μπορούμε να εφαρμόσουμε φίλτρα που εμφανίζουν μόνο την κίνηση που χρησιμοποιεί τη θύρα 80, η οποία αντιστοιχεί στο HTTP. Αυτή η απομόνωση συμβάλλει στη διατήρηση των κρίσιμων δεδομένων που εί-

ναι απαραίτητα για τον εντοπισμό επιθέσεων, ενώ παράλληλα αφαιρεί την άσχετη κίνηση που μπορεί να επηρεάσει αρνητικά τα αποτελέσματα της ανάλυσης.

Εκτός από την εφαρμογή φίλτρων, τα εργαλεία επεξεργασίας και ανάλυσης αρχείων καταγραφής μπορούν να χρησιμοποιηθούν για λεπτομερέστερο φιλτράρισμα. Εργαλεία όπως τα ‘awk’, ‘grep’ και ‘sed’ παρέχουν ευέλικτες μεθόδους για την εξαγωγή των δεδομένων που είναι απαραίτητα και την αφαίρεση εκείνων που δεν είναι σχετικά με τη μελέτη μας. Αυτά τα εργαλεία μας επιτρέπουν να αναλύουμε ταυτόχρονα μεγάλα αρχεία καταγραφής και να εξάγουμε μόνο τα δεδομένα που ανταποκρίνονται σε συγκεκριμένα κριτήρια, όπως πρωτόκολλα, θύρες ή άλλες παραμέτρους. Για παράδειγμα, εάν η ανάλυση εστιάζει μόνο σε πακέτα TCP που χρησιμοποιούν τη θύρα 80, μπορούμε να εφαρμόσουμε την ακόλουθη εντολή:

Listing D.2: Χρήση της *tshark* για φιλτράρισμα *TCP traffic*

```
1 tshark -r input.pcap -Y "tcp.port == 80" -w output.pcap
```

Αυτή η εντολή επιτρέπει την ανάγνωση ενός αρχείου PCAP (‘input.pcap’), φιλτράροντας τα πακέτα TCP που σχετίζονται αποκλειστικά με τη θύρα 80 και αποθηκεύοντας τα αποτελέσματα σε ένα νέο αρχείο PCAP (‘output.pcap’). Με αυτόν τον τρόπο, αφαιρούνται όλα τα πακέτα που δε σχετίζονται με την κίνηση HTTP, δίνοντάς μας ένα καθαρότερο και πιο στοχευμένο σύνολο δεδομένων. Αυτή η διαδικασία όχι μόνο εξασφαλίζει την απομάκρυνση των περιττών δεδομένων, αλλά διευκολύνει και την περαιτέρω ανάλυση, μειώνοντας τον όγκο των δεδομένων που πρέπει να επεξεργαστούν τα μοντέλα μηχανικής μάθησης.

Η χρήση τέτοιων εργαλείων και τεχνικών φιλτραρίσματος είναι ζωτικής σημασίας για τη δημιουργία καθαρών συνόλων δεδομένων που επιτρέπουν την ακριβέστερη αναπαράσταση της πραγματικής κυκλοφορίας και των επιθέσεων που μελετώνται. Με την αφαίρεση των άσχετων πακέτων, τα σύνολα δεδομένων αποκτούν μεγαλύτερη συνοχή και νόημα, παρέχοντας τη βάση για πιο αξιόπιστη και ακριβή ανάλυση και μοντελοποίηση.

Κοινωνικοποίηση Χαρακτηριστικών

Εξομάλυνση Δεδομένων

Η εξομάλυνση των δεδομένων είναι ένας από τους θεμελιώδεις πυλώνες της προεπεξεργασίας σε ένα σύστημα ανάλυσης δικτύων και ανίχνευσης επιθέσεων. Σε πολλές περιπτώσεις, τα δεδομένα από τα αρχεία καταγραφής (PCAP) είναι σε ακατέργαστη μορφή, με πληροφορίες που δεν είναι τυποποιημένες και μπορεί να περιπλέξουν τη διαδικασία ανάλυσης ή τη χρήση τους για την εκπαίδευση μοντέλων μηχανικής μάθησης. Η κανονικοποίηση των χαρακτηριστικών, όπως οι διευθύνσεις IP και οι θύρες, είναι το κλειδί για τη διασφάλιση της ομοιογένειας και της συγκρισιμότητας των δεδομένων σε όλα τα στάδια της ανάλυσης.

Για την κανονικοποίηση των διευθύνσεων IP και των θυρών, μπορούμε να χρησιμοποιήσουμε γλώσσες προγραμματισμού όπως η Python και βιβλιοθήκες που ειδικεύονται στην επεξεργασία δικτυακών πακέτων, όπως η `Scapy`² και η `Pyshark`. Αυτά τα εργαλεία επιτρέπουν την άμεση επεξεργασία αρχείων PCAP, προσφέροντας δυνατότητες εξαγωγής των απαραίτητων χαρακτηριστικών με ακρίβεια και ταχύτητα. Για παράδειγμα, το `Scapy` επιτρέπει τη φόρτωση πακέτων από ένα αρχείο PCAP και την εξαγωγή των διευθύνσεων IP πηγής και προορισμού. Μέσω αυτής της διαδικασίας, τα δεδομένα μπορούν να μορφοποιηθούν σε τυποποιημένη και κανονικοποιημένη μορφή, διευκολύνοντας την περαιτέρω ανάλυση ή την εισαγωγή σε αλγόριθμους μηχανικής μάθησης.

Παρακάτω παρατίθεται ένα παράδειγμα κώδικα Python που χρησιμοποιεί τη βιβλιοθήκη `Scapy`, ο οποίος διαβάζει ένα αρχείο PCAP και εξαγει τις διευθύνσεις IP πηγής και προορισμού:

Listing D.3: *Εξαγωγή πηγής και προορισμού IP διευθύνσεων από αρχείο PCAP με χρήση Scapy.*

```
1 from scapy.all import rdpcap
2
3 packets = rdpcap('input.pcap')
4 for pkt in packets:
5     if IP in pkt:
6         print(pkt[IP].src, pkt[IP].dst)
```

Χρησιμοποιώντας αυτόν τον κώδικα, μπορούμε να επεξεργαστούμε τα πακέτα και να εξα-

²<https://github.com/secdev/scapy>

γουμε τις πληροφορίες IP που είναι κρίσιμες για την ανάλυσή μας. Αυτή η διαδικασία είναι ιδιαίτερα χρήσιμη όταν θέλουμε να κανονικοποιήσουμε τα δεδομένα για τη δημιουργία μοντέλων ανίχνευσης επιθέσεων, όπου η ακρίβεια των δεδομένων εισόδου είναι θεμελιώδης για την αξιοπιστία του τελικού αποτελέσματος.

Μετατροπή σε συγκρίσιμα Μορφότυπα

Εκτός από την κανονικοποίηση των διευθύνσεων IP και των θυρών, είναι εξίσου απαραίτητο να διασφαλιστεί ότι όλες οι πληροφορίες στο αρχείο δεδομένων είναι σε μορφή που διευκολύνει την ανάλυση και τη συγκρισιμότητα. Κεντρικό στοιχείο αυτής της διαδικασίας είναι οι χρονοσφραγίδες (timestamps), οι οποίες μπορούν να βρεθούν σε διαφορετικές μορφές ανάλογα με την πηγή των δεδομένων. Για παράδειγμα, ορισμένες timestamps μπορεί να εκφράζονται σε μορφή αναγνώσιμη από τον άνθρωπο, ενώ άλλες μπορεί να είναι σε μορφή timestamps UNIX. Η μετατροπή των timestamps σε μια κοινή τυποποιημένη μορφή, όπως η timestamps UNIX, εξασφαλίζει την ομαλή σύγκριση δεδομένων από διαφορετικές πηγές και διευκολύνει την ενοποίηση των αποτελεσμάτων.

Επιπλέον, για τη διευκόλυνση της ανάλυσης και της συγκρισιμότητας των δεδομένων, είναι σημαντικό τα χαρακτηριστικά να εξάγονται σε ευανάγνωστες και τυποποιημένες μορφές, όπως CSV ή άλλα δομημένα αρχεία. Η μετατροπή των δεδομένων σε μορφές όπως το CSV επιτρέπει την εύκολη επεξεργασία και ανάλυση με διάφορα εργαλεία, συμπεριλαμβανομένων εργαλείων στατιστικής ανάλυσης και αλγορίθμων μηχανικής μάθησης.

Για να μετατρέψουμε τα δεδομένα σε συγκρίσιμη και αναλύσιμη μορφή, μπορούμε να χρησιμοποιήσουμε τις δυνατότητες που προσφέρουν εργαλεία όπως το `tshark`³ ή ακόμη και γλώσσες προγραμματισμού για μαζική επεξεργασία δεδομένων. Η διαδικασία αυτή μας επιτρέπει να λαμβάνουμε ομοιογενή σύνολα δεδομένων που μπορούν εύκολα να ενσωματωθούν σε διαδικασίες ανάλυσης υψηλότερου επιπέδου, αυξάνοντας την αποτελεσματικότητα και την ακρίβεια των αποτελεσμάτων.

Η συνολική κανονικοποίηση και τυποποίηση των δεδομένων είναι ζωτικής σημασίας για την

³<https://www.wireshark.org/docs/man-pages/tshark.html>

επιτυχή διεξαγωγή μιας ακριβούς και ολοκληρωμένης ανάλυσης δικτύου. Με την εφαρμογή αυτών των διαδικασιών, διασφαλίζουμε ότι τα δεδομένα βρίσκονται σε μορφή που μπορεί να αξιοποιηθεί αποτελεσματικά, οδηγώντας σε αξιόπιστα και χρήσιμα αποτελέσματα για την ανίχνευση επιθέσεων και τη βελτιστοποίηση των μοντέλων μηχανικής μάθησης που χρησιμοποιούνται για την ανάλυση της δικτυακής κίνησης.

Ανίχνευση και Αντιμετώπιση Ακραίων Τιμών (Outliers)

Η ανίχνευση και η αντιμετώπιση των ακραίων τιμών (outliers) είναι ένα κρίσιμο στάδιο στη διαδικασία προεπεξεργασίας δεδομένων, καθώς οι ακραίες τιμές μπορούν να επηρεάσουν αρνητικά την ακρίβεια και την απόδοση των μοντέλων μηχανικής μάθησης. Αυτό ισχύει ιδιαίτερα στις μεθόδους που βασίζονται σε στατιστικά μοντέλα, όπως η γραμμική παλινδρόμηση και τα δέντρα αποφάσεων. Ως εκ τούτου, η σωστή ανίχνευση και διαχείριση αυτών των τιμών είναι ζωτικής σημασίας για την επίτευξη αξιόπιστων και σταθερών προβλέψεων.

Η ανίχνευση ακραίων τιμών μπορεί να γίνει με διάφορες μεθόδους. Μια από τις πιο συνηθισμένες μεθόδους είναι η χρήση των Z-scores. Το Z-score μετρά την απόσταση μιας τιμής από τον μέσο όρο σε μονάδες τυπικής απόκλισης. Εάν μια τιμή έχει Z-score μεγαλύτερο από έναν προκαθορισμένο αριθμό, όπως το 3, τότε μπορεί να θεωρηθεί ακραία τιμή. Η χρήση αυτής της μεθόδου είναι ιδιαίτερα αποτελεσματική όταν τα δεδομένα ακολουθούν κανονική κατανομή. Μια εναλλακτική προσέγγιση είναι η χρήση του διατεταρτημοριακού εύρους (Interquartile Range, IQR), το οποίο υπολογίζει το εύρος μεταξύ του πρώτου και του τρίτου τεταρτημορίου και εντοπίζει τις ακραίες τιμές που βρίσκονται πέρα από αυτό το εύρος. Η μέθοδος IQR είναι ανεξάρτητη από την κατανομή των δεδομένων και μπορεί να χρησιμοποιηθεί ευρέως σε διάφορους τύπους συνόλων δεδομένων. Το εργαλείο `numpy`⁴ έκανε εξαιρετική δουλειά.

Listing D.4: Υπολογισμός Z-score με χρήση *Numpy* και *Scipy* για ανίχνευση εκτοπίσεων.

```
1 import numpy as np
2 from scipy import stats
3
4 z_scores = np.abs(stats.zscore(df[ 'feature_column' ]))
```

⁴<https://numpy.org/>

```
5 df_clean = df[(z_scores < 3)]
```

Listing D.5: Υπολογισμός Z-score χρησιμοποιώντας Pandas για ανίχνευση εκτοπίσεων.

```
1 import pandas as pd
2
3 z_scores = (df['feature_column'] - df['feature_column'].mean()) /
4
5 df['feature_column'].std()
6 df_clean = df[(z_scores < 3)]
```

Μόλις εντοπιστούν οι ακραίες τιμές, υπάρχουν διάφορες στρατηγικές για την αντιμετώπισή τους. Μια από τις πιο συνηθισμένες μεθόδους είναι η αφαίρεση των ακραίων τιμών από το σύνολο δεδομένων. Παρόλο που η αφαίρεση μπορεί να διασφαλίσει ότι οι ακραίες τιμές δεν επηρεάζουν το μοντέλο, υπάρχει ο κίνδυνος να χαθούν πολύτιμες πληροφορίες, ιδίως αν οι ακραίες τιμές είναι σημαντικά δεδομένα για την ανάλυση.

Μια άλλη επιλογή είναι ο μετασχηματισμός των ακραίων τιμών με τεχνικές όπως ο λογαριθμικός μετασχηματισμός (log transformation), ο οποίος μειώνει την επίδρασή τους χωρίς να τις αφαιρεί από το σύνολο δεδομένων. Αυτός ο μετασχηματισμός είναι ιδιαίτερα χρήσιμος όταν τα δεδομένα έχουν μεγάλη διακύμανση ή εκθετική κατανομή. Τέλος, οι ακραίες τιμές μπορούν να αντικατασταθούν με πιο λογικές τιμές, όπως η μέση τιμή ή η διάμεσος. Αυτή η προσέγγιση είναι χρήσιμη σε περιπτώσεις όπου η διατήρηση του μεγέθους του συνόλου δεδομένων είναι κρίσιμη, αλλά απαιτείται κάποια προσαρμογή για τη μείωση του αντίκτυπου των ακραίων τιμών. Η επιλογή της κατάλληλης μεθόδου εξαρτάται από τη φύση των δεδομένων και τους στόχους της ανάλυσης.

Κατηγοριοποίηση Δεδομένων (Binning)

Η κατηγοριοποίηση αριθμητικών δεδομένων σε διακριτές κατηγορίες, γνωστή και ως binning, είναι μια αποτελεσματική τεχνική που μπορεί να βελτιώσει την ανάλυση και την απόδοση των μοντέλων μηχανικής μάθησης. Η διαδικασία αυτή είναι ιδιαίτερα χρήσιμη όταν τα δεδομένα έχουν ανομοιόμορφη κατανομή ή μεγάλες διακυμάνσεις. Με το binning, οι συνεχείς αριθμητικές τιμές μετατρέπονται σε ένα σύνολο κατηγοριών, το οποίο μπορεί να βοηθήσει τα μοντέλα να κατανοήσουν καλύτερα τα δεδομένα.

Για παράδειγμα, μια τυπική εφαρμογή της τεχνικής binning είναι η κατηγοριοποίηση των διευθύνσεων IP σε internal και external . Αυτή η προσέγγιση μπορεί να βοηθήσει στην ανίχνευση κακόβουλης συμπεριφοράς με τη διαφοροποίηση της δραστηριότητας εντός του εσωτερικού δικτύου από το εξωτερικό δίκτυο. Ένας άλλος τρόπος χρήσης της κατηγοριοποίησης είναι η δημιουργία κατηγοριών με βάση τον αριθμό των πακέτων που στέλνει ένας χρήστης σε μια χρονική περίοδο, κατηγοριοποιώντας τους χρήστες σε επίπεδα όπως «low», «medium» και «high» traffic.

Μπορείτε να υλοποιήσετε τη διαχωρισμό χρησιμοποιώντας τη βιβλιοθήκη pandas, η οποία παρέχει εύχρηστες συναρτήσεις όπως οι 'cut' και 'qcut'. Το 'cut' χρησιμοποιείται για τη δημιουργία διακριτών κατηγοριών με βάση προκαθορισμένα όρια, ενώ το 'qcut' δημιουργεί κατηγορίες με βάση τεταρτημόρια, εξασφαλίζοντας ίση ποσότητα δεδομένων σε κάθε κατηγορία.

Listing D.6: Δημιουργία κατηγοριών βάσει διαστημάτων (*bins*) για τον αριθμό των πακέτων

```
1 import pandas as pd
2
3 bins = [0, 100, 500, 1000, float('inf')]
4 labels = ['Low', 'Medium', 'High', 'Very High']
5 df['traffic_category'] = pd.cut(df['packet_count'], bins=bins, labels=
    =labels)
```

Η τεχνική αυτή προσφέρει πολλά πλεονεκτήματα. Αφενός, διευκολύνει την ανάλυση και την ερμηνεία των δεδομένων, δεδομένου ότι οι κατηγορίες είναι ευκολότερα κατανοητές από τα συνεχή δεδομένα. Από την άλλη πλευρά, μπορεί να μειώσει την επίδραση των ακραίων τιμών, καθώς αυτές ενσωματώνονται σε συγκεκριμένες κατηγορίες. Η κατηγοριοποίηση είναι επίσης χρήσιμη όταν θέλουμε να αντιμετωπίσουμε την άνιση κατανομή των δεδομένων και να μειώσουμε την πολυπλοκότητα, γεγονός που μπορεί να συμβάλει στη βελτίωση της γενικευσιμότητας των μοντέλων.

Διαχείριση Ελλιπών Τιμών (Missing Values)

Η διαχείριση των ελλιπών τιμών είναι ένα κρίσιμο βήμα στην προεπεξεργασία δεδομένων, καθώς η παρουσία τους μπορεί να επηρεάσει αρνητικά την ποιότητα και την απόδοση των μοντέλων μηχανικής μάθησης. Οι ελλιπείς τιμές μπορεί να οδηγήσουν σε σφάλματα κατά τη διαδικασία μοντελοποίησης, μειώνοντας την αξιοπιστία των αποτελεσμάτων. Ως εκ τούτου, είναι απαραίτητο να ανιχνεύονται και να διαχειρίζονται οι ελλείπουσες τιμές πριν προχωρήσουμε στη φάση της εκπαίδευσης.

Η ανίχνευση ελλιπών τιμών είναι μια εύκολη διαδικασία με τη χρήση της βιβλιοθήκης pandas. Η ενσωματωμένη συνάρτηση `isnull()` επιτρέπει την ανίχνευση ελλιπών τιμών σε κάθε στήλη ή γραμμή ενός DataFrame. Αυτό το βήμα παρέχει μια σαφή εικόνα της έκτασης του προβλήματος και μας βοηθά να αποφασίσουμε την κατάλληλη στρατηγική για την αντιμετώπισή του.

Listing D.7: Ανίχνευση ελλειπουσών τιμών με χρήση Pandas για προεπεξεργασία δεδομένων.

```
1 import pandas as pd
2
3 missing_values = df.isnull().sum()
4 print(missing_values)
```

Μόλις εντοπιστούν οι ελλείπουσες τιμές, υπάρχουν διάφορες στρατηγικές για την αντιμετώπισή τους, ανάλογα με τη φύση των δεδομένων και τον αντίκτυπο των ελλείπουσων τιμών στην ανάλυση. Μια από τις πιο συνηθισμένες μεθόδους είναι ο υπολογισμός των ελλειπών τιμών. Ο υπολογισμός μπορεί να γίνει με τη χρήση του μέσου όρου, της διαμέσου ή ακόμη και με πιο σύνθετες τεχνικές, όπως η χρήση αλγορίθμων παλινδρόμησης, οι οποίοι προβλέπουν τις ελλείπουσες τιμές με βάση άλλες μεταβλητές.

Μια άλλη δημοφιλής προσέγγιση είναι η αφαίρεση γραμμών ή στηλών που περιέχουν ελλιπείς τιμές. Η αφαίρεση μπορεί να είναι μια λογική επιλογή, ιδίως όταν οι ελλείπουσες τιμές αφορούν ένα μικρό ποσοστό των δεδομένων ή όταν η αφαίρεση αυτών των στοιχείων δεν επηρεάζει σημαντικά την ανάλυση.

Listing D.8: Αντικατάσταση ελλιπών τιμών με τη μέση τιμή και Αφαίρεση σειρών με ελλιπείς τιμές

```
1
2 df.fillna(df.mean(), inplace=True)
3
4 df.dropna(inplace=True)
```

Η επιλογή της κατάλληλης στρατηγικής εξαρτάται από τον αριθμό των τιμών που λείπουν και τη σημασία των δεδομένων που λείπουν. Εάν τα ελλείποντα δεδομένα είναι πολυάριθμα και τυχαία καταναμεμημένα, η αντικατάσταση μπορεί να είναι η καταλληλότερη λύση. Αντίθετα, εάν τα ελλείποντα δεδομένα είναι περιορισμένα και η αφαίρεση δε θα επηρεάσει την ανάλυση, η αφαίρεση των σχετικών γραμμών ή στηλών είναι η απλούστερη προσέγγιση.

Ανάλυση και Επιλογή Χαρακτηριστικών (Feature Selection)

Η ανάλυση και η επιλογή των πιο σχετικών χαρακτηριστικών είναι ένα κρίσιμο βήμα για τη

δημιουργία αποτελεσματικών μοντέλων μηχανικής μάθησης. Η σωστή επιλογή χαρακτηριστικών μπορεί να μειώσει την πολυπλοκότητα του μοντέλου, να βελτιώσει την απόδοσή του και να ελαχιστοποιήσει την πιθανότητα υπερπροσαρμογής (overfitting). Υπάρχουν διάφορες τεχνικές για την επιλογή σημαντικών χαρακτηριστικών, οι οποίες μπορούν να βοηθήσουν στην εξαγωγή των πιο χρήσιμων πληροφοριών από τα δεδομένα.

Μία από τις βασικές τεχνικές για την επιλογή χαρακτηριστικών είναι η ανάλυση συσχέτισης (Correlation Analysis). Η τεχνική αυτή αναλύει τη σχέση μεταξύ των χαρακτηριστικών και της ετικέτας (label) του προβλήματος. Χαρακτηριστικά που έχουν μικρή ή καθόλου συσχέτιση με το στόχο μπορούν να αφαιρεθούν από το μοντέλο, καθώς η συμβολή τους στην πρόβλεψη είναι περιορισμένη. Χρησιμοποιώντας τη βιβλιοθήκη pandas, μπορείτε να υπολογίσετε τον πίνακα συσχέτισης και να αναλύσετε τη συνάφεια των χαρακτηριστικών.

Listing D.9: Υπολογισμός συσχέτισης χαρακτηριστικών με τον στόχο, χρησιμοποιώντας *Pandas*.

```
1 import pandas as pd
2
3 correlation_matrix = df.corr()
4 print(correlation_matrix['target'].sort_values(ascending=False))
```

Μια άλλη δημοφιλής μέθοδος είναι η Recursive Feature Elimination (Αναδρομική Εξάλειψη Χαρακτηριστικών, RFE). Η RFE αφαιρεί επαναληπτικά τα χαρακτηριστικά που έχουν τη μικρότερη συνεισφορά στην απόδοση του μοντέλου μέχρι να φτάσει στον επιθυμητό αριθμό χαρακτηριστικών. Αυτή η τεχνική μπορεί να βοηθήσει στην αφαίρεση χαρακτηριστικών που δεν παρέχουν σημαντικές πληροφορίες και να μειώσει την πολυπλοκότητα του μοντέλου.

Listing D.10: Εφαρμογή *Recursive Feature Elimination (RFE)* με λογιστική παλινδρόμηση για επιλογή χαρακτηριστικών.

```
1 from sklearn.feature_selection import RFE
2 from sklearn.linear_model import LogisticRegression
3
4 model = LogisticRegression()
5 rfe = RFE(model, n_features_to_select=5)
```

```
6 fit = rfe.fit(X, y)
7 print(fit.support_)
8 print(fit.ranking_)
```

Η ανάλυση κύριων συνιστωσών (Principal Component Analysis, PCA) είναι μια τρίτη προσέγγιση, η οποία χρησιμοποιείται κυρίως για τη μείωση της διαστατικότητας των δεδομένων. Η PCA αναλύει τα δεδομένα και εξάγει τα πιο σημαντικά χαρακτηριστικά με βάση τη μεταβλητότητα που εξηγούν. Με αυτόν τον τρόπο, μπορούμε να διατηρήσουμε μόνο τις κύριες συνιστώσες που έχουν τη μεγαλύτερη επίδραση, μειώνοντας τη διάσταση των δεδομένων χωρίς να χάνονται σημαντικές πληροφορίες.

Listing D.11: Εφαρμογή *Principal Component Analysis (PCA)* για μείωση διάστασης δεδομένων.

```
1 from sklearn.decomposition import PCA
2
3 pca = PCA(n_components=3)
4 X_pca = pca.fit_transform(X)
5 print(pca.explained_variance_ratio_)
```

Η επιλογή των σωστών χαρακτηριστικών μπορεί να βελτιώσει την αποτελεσματικότητα του μοντέλου, καθώς λιγότερα χαρακτηριστικά μειώνουν τον υπολογιστικό φόρτο, καθιστούν το μοντέλο ευκολότερο στην ερμηνεία και ελαχιστοποιούν τον κίνδυνο υπερβολικής προσαρμογής. Ανάλογα με τη φύση των δεδομένων και τους στόχους του έργου, η χρήση μιας ή περισσότερων τεχνικών επιλογής χαρακτηριστικών μπορεί να βελτιώσει σημαντικά την απόδοση του μοντέλου.

Κοινωνικοποίηση Δεδομένων (Scaling)

Η κανονικοποίηση δεδομένων (scaling) είναι μια κρίσιμη διαδικασία στην προεπεξεργασία δεδομένων, ιδίως όταν τα δεδομένα προέρχονται από διαφορετικές κλίμακες, όπως συμβαίνει συχνά με τα δεδομένα δικτύου. Η κανονικοποίηση συμβάλλει στη βελτίωση της απόδοσης πολλών αλγορίθμων μηχανικής μάθησης, όπως τα νευρωνικά δίκτυα και οι μηχανές διανυσμάτων υποστήριξης (SVM), οι οποίες επηρεάζονται από τις διαφορές στις κλίμακες των χαρακτηριστικών. Εάν δεν κανονικοποιηθούν σωστά, τα χαρακτηριστικά με μεγάλες τιμές μπορεί

να κυριαρχήσουν στα δεδομένα, επηρεάζοντας τα αποτελέσματα του μοντέλου.

Μια από τις πιο συνηθισμένες τεχνικές κανονικοποίησης είναι η κλιμάκωση Min-Max, η οποία κανονικοποιεί τα δεδομένα σε ένα συγκεκριμένο εύρος, συνήθως μεταξύ 0 και 1. Αυτό μπορεί να είναι χρήσιμο όταν θέλουμε να εξισορροπήσουμε την επίδραση των χαρακτηριστικών και να αποφύγουμε την υπεροχή των χαρακτηριστικών με μεγάλες τιμές.

Listing D.12: Κανονικοποίηση δεδομένων με χρήση *MinMaxScaler* από τη βιβλιοθήκη *sklearn*.

```
1 from sklearn.preprocessing import MinMaxScaler
2
3 scaler = MinMaxScaler()
4 scaled_data = scaler.fit_transform(df)
```

Μια άλλη συνήθης τεχνική είναι η (Standardization), η οποία μετασχηματίζει τα δεδομένα ώστε να έχουν μέσο όρο 0 και τυπική απόκλιση 1. Αυτή η μέθοδος είναι ιδιαίτερα χρήσιμη όταν τα δεδομένα δεν έχουν συγκεκριμένο εύρος ή όταν αναμένουμε οι τιμές να ακολουθούν κανονική κατανομή. Με την τυποποίηση, κάθε χαρακτηριστικό συμβάλλει ομοιόμορφα στη διαδικασία εκπαίδευσης του μοντέλου χωρίς να επηρεάζει την κλίμακα του μοντέλου.

Listing D.13: Κανονικοποίηση δεδομένων με χρήση *StandardScaler* από τη βιβλιοθήκη *sklearn*

```
1 from sklearn.preprocessing import StandardScaler
2
3 scaler = StandardScaler()
4 scaled_data = scaler.fit_transform(df)
```

Η επιλογή της κατάλληλης τεχνικής κανονικοποίησης εξαρτάται από τον τύπο του προβλήματος και τα δεδομένα που χρησιμοποιούμε. Η κλιμάκωση Min-Max είναι συνήθως προτιμότερη όταν τα δεδομένα μας έχουν ένα φυσικό εύρος τιμών που θέλουμε να διατηρήσουμε, ενώ η Standardization προτιμάται όταν θέλουμε να εξισορροπήσουμε τα δεδομένα με διαφορετικές μονάδες μέτρησης ή όταν αναμένουμε κανονική κατανομή.

Η κανονικοποίηση είναι σημαντική για την ορθή λειτουργία των αλγορίθμων και μπορεί να βελτιώσει την ακρίβεια των προβλέψεων, μειώνοντας παράλληλα το χρόνο εκπαίδευσης και βελτιώνοντας τη σύγκλιση του μοντέλου.

Κεφάλαιο Ε

Ανίχνευση Εισβολών μέσω Ανάλυσης Δικτυακής Κίνησης

Σε αυτό κεφάλαιο παρουσιάζονται τα αποτελέσματα της εκπαίδευσης και της αξιολόγησης των μοντέλων μηχανικής μάθησης που αναπτύχθηκαν για την ανίχνευση εισβολών σε δίκτυα υπολογιστών. Η ανάλυση βασίζεται στις μετρικές που χρησιμοποιούνται για την απόδοση των μοντέλων, όπως η ακρίβεια, η ανάκληση και η ακρίβεια ανίχνευσης. Επιπλέον, πραγματοποιείται λεπτομερής εξέταση των ψευδώς θετικών και ψευδώς αρνητικών αποτελεσμάτων, καθώς και σύγκριση των επιδόσεων των μοντέλων σε διαφορετικά datasets.

Απόδοση των Μοντέλων

Η εκπαίδευση και η αξιολόγηση του μοντέλου πραγματοποιήθηκε χρησιμοποιώντας διάφορα σύνολα δεδομένων ροής δικτύου. Οι αλγόριθμοι μηχανικής μάθησης που επιλέχθηκαν για την ανάλυση αυτή ήταν οι Random Forest, Support Vector Machines (SVM) και Neural Networks. Κάθε μοντέλο εκπαιδεύτηκε με τα προεπεξεργασμένα δεδομένα, τα οποία περιλάμβαναν χαρακτηριστικά ροής δικτύου που εξήχθησαν μέσω των εργαλείων NFStream ή CICFlowMeter.

Τι είναι το NFStream;

Το NFStream είναι μια βιβλιοθήκη ανοικτού κώδικα που χρησιμοποιείται για την ανάλυση ροών δικτύου. Έχει σχεδιαστεί για να επιτρέπει την αποτελεσματική εξαγωγή χαρακτηριστικών από δεδομένα κίνησης δικτύου και να παρέχει στατιστικά στοιχεία σχετικά με τη ροή δεδομένων στο δίκτυο. Το NFStream χρησιμοποιείται ευρέως σε περιπτώσεις όπου απαιτείται η ανάλυση μεγάλων συνόλων δεδομένων δικτύου, όπως σε εφαρμογές συστημάτων ανίχνευσης εισβολών (IDS), ανάλυσης επιδόσεων δικτύου και έρευνας ασφάλειας δικτύου.

Βασικά Χαρακτηριστικά του NFStream:

Υποστήριξη για PCAP και ζωντανή λήψη: NFStream μπορεί να επεξεργάζεται δεδομένα από αρχεία PCAP (αποθηκευμένες ροές δεδομένων δικτύου), αλλά και να αναλύει ζωντανά δεδομένα που προέρχονται απευθείας από το δίκτυο σε πραγματικό χρόνο.

Εξαγωγή χαρακτηριστικών (Features):

Παρέχει την εξαγωγή διάφορων σημαντικών χαρακτηριστικών δικτυακών ροών, όπως:

- **Διευθύνσεις IP** (Πηγής και προορισμού)
- **Πρωτόκολλο** (TCP, UDP κτλ.)
- **Αριθμός πακέτων** (total packets)
- **Συνολικά bytes** (total bytes)
- **Διάρκεια ροής** (Flow duration)

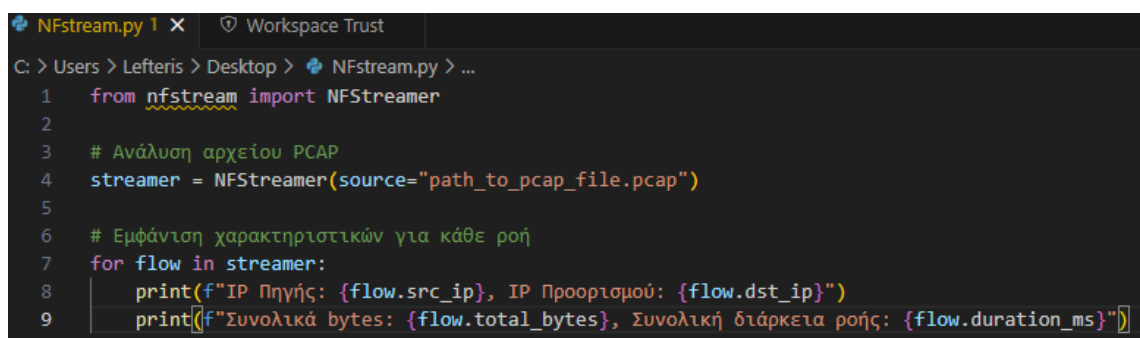
1. **Bidirectional και Unidirectional statistics:** Στατιστικά στοιχεία για τη ροή δεδομένων και προς τις δύο κατευθύνσεις, τα οποία είναι κρίσιμα για την ανάλυση της ροής των πακέτων μεταξύ των κόμβων.

2. **Υποστήριξη πολλαπλών πλατφορμών:** Η βιβλιοθήκη είναι διαθέσιμη για Windows, Linux και macOS και μπορεί να ενσωματωθεί σε εφαρμογές που βασίζονται στη γλώσσα προγραμματισμού Python.
3. **Υψηλές επιδόσεις:** Το NFStream έχει βελτιστοποιηθεί για την επεξεργασία μεγάλων ποσοτήτων δεδομένων δικτύου με ταχύτητα και αποτελεσματικότητα, καθιστώντας το κατάλληλο για χρήση σε μεγάλα σύνολα δεδομένων όπως το CICIDS και το Defcon.
4. **Προσαρμογή :** Το NFStream επιτρέπει στους χρήστες να ορίζουν προσαρμοσμένα χαρακτηριστικά και να φιλτράρουν τη ροή δεδομένων, προσφέροντας μεγαλύτερη ευελιξία στην ανάλυση δεδομένων δικτύου.

Πώς χρησιμοποιείτε το NFStream ;

Το NFStream μπορεί να χρησιμοποιηθεί για την ανάλυση αρχείων PCAP, τα οποία περιέχουν καταγεγραμμένα δεδομένα από πραγματικά δίκτυα, ή για την ανάλυση ζωντανών δεδομένων. Στην πράξη, μπορεί να χρησιμοποιηθεί για την εξαγωγή χαρακτηριστικών από ροές δεδομένων δικτύου, τα οποία μπορούν στη συνέχεια να τροφοδοτηθούν σε μοντέλα μηχανικής μάθησης για την ανίχνευση εισβολών.

Ένα βασικό παράδειγμα χρήσης του NFStream σε Python είναι το παρακάτω:



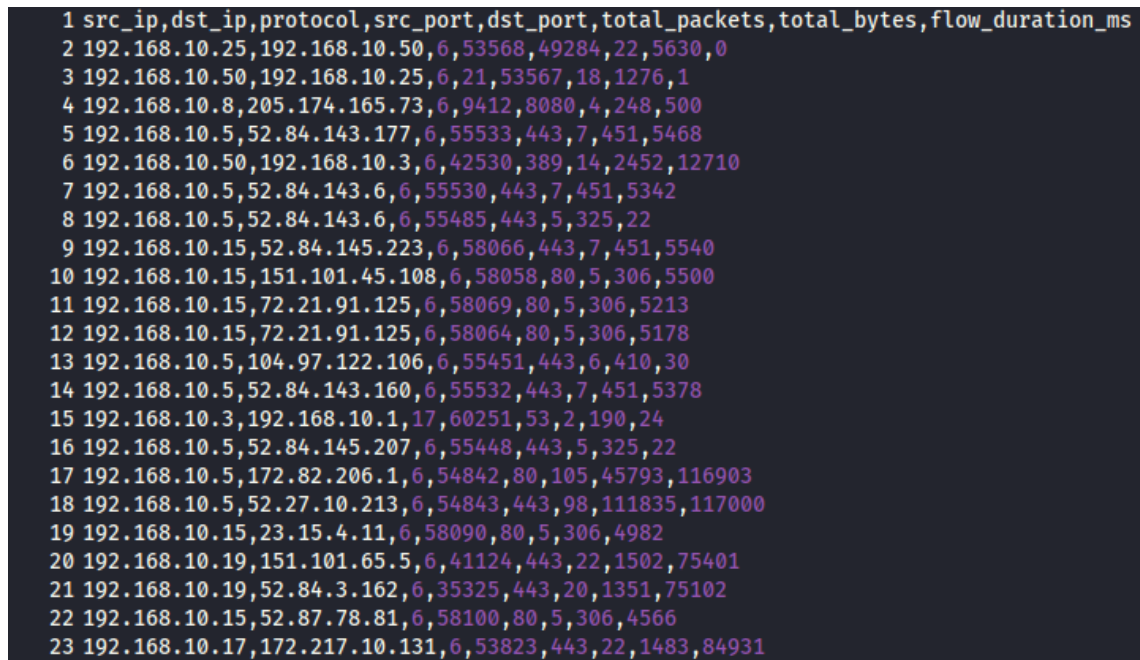
```

1  from nfstream import NFStreamer
2
3  # Ανάλυση αρχείου PCAP
4  streamer = NFStreamer(source="path_to_pcap_file.pcap")
5
6  # Εμφάνιση χαρακτηριστικών για κάθε ροή
7  for flow in streamer:
8      print(f"IP Πηγής: {flow.src_ip}, IP Προορισμού: {flow.dst_ip}")
9      print(f"Συνολικά bytes: {flow.total_bytes}, Συνολική διάρκεια ροής: {flow.duration_ms}")
  
```

Σχήμα Ε.25: Ανάλυση αρχείου PCAP με χρήση NFStream και εξαγωγή χαρακτηριστικών ροής δικτύου.

Αυτός ο κώδικας αναλύει το αρχείο PCAP και εκτυπώνει τα βασικά χαρακτηριστικά για

κάθε ροή δικτύου. Το NFStream μπορεί να χρησιμοποιηθεί για την εξαγωγή λεπτομερών στατιστικών στοιχείων για την ανάλυση των ροών δικτύου και τα δεδομένα αυτά μπορούν στη συνέχεια να χρησιμοποιηθούν για την εκπαίδευση μοντέλων μηχανικής μάθησης.



	src_ip	dst_ip	protocol	src_port	dst_port	total_packets	total_bytes	flow_duration_ms
1	192.168.10.25	192.168.10.50	6	53568	49284	22	5630	0
2	192.168.10.50	192.168.10.25	6	21	53567	18	1276	1
3	192.168.10.8	205.174.165.73	6	9412	8080	4	248	500
4	192.168.10.5	52.84.143.177	6	55533	443	7	451	5468
5	192.168.10.50	192.168.10.3	6	42530	389	14	2452	12710
6	192.168.10.5	52.84.143.6	6	55530	443	7	451	5342
7	192.168.10.5	52.84.143.6	6	55485	443	5	325	22
8	192.168.10.15	52.84.145.223	6	58066	443	7	451	5540
9	192.168.10.15	151.101.45.108	6	58058	80	5	306	5500
10	192.168.10.15	72.21.91.125	6	58069	80	5	306	5213
11	192.168.10.15	72.21.91.125	6	58064	80	5	306	5178
12	192.168.10.5	104.97.122.106	6	55451	443	6	410	30
13	192.168.10.5	52.84.143.160	6	55532	443	7	451	5378
14	192.168.10.3	192.168.10.1	17	60251	53	2	190	24
15	192.168.10.5	52.84.145.207	6	55448	443	5	325	22
16	192.168.10.5	172.82.206.1	6	54842	80	105	45793	116903
17	192.168.10.5	52.27.10.213	6	54843	443	98	111835	117000
18	192.168.10.15	23.15.4.11	6	58090	80	5	306	4982
19	192.168.10.19	151.101.65.5	6	41124	443	22	1502	75401
20	192.168.10.19	52.84.3.162	6	35325	443	20	1351	75102
21	192.168.10.15	52.87.78.81	6	58100	80	5	306	4566
22	192.168.10.17	172.217.10.131	6	53823	443	22	1483	84931
23								

Σχήμα Ε.26: Πίνακας δικτυακών ροών με στοιχεία όπως IP πηγής, IP προορισμού, πρωτόκολλο, θύρες, συνολικά πακέτα, συνολικά bytes και διάρκεια ροής σε milliseconds.

Πλεονεκτήματα του NFStream:

- **Ευκολία χρήσης:** Η βιβλιοθήκη είναι εύκολο να ενσωματωθεί και να χρησιμοποιηθεί σε έργα ανάλυσης δικτύων, ιδίως σε εφαρμογές μηχανικής μάθησης (γι' αυτό και χρησιμοποιήθηκε κυρίως στην παρούσα εργασία).
- **Ταχύτητα και επιδόσεις:** Σχεδιασμένο για να αναλύει γρήγορα και αποτελεσματικά μεγάλα σύνολα δεδομένων δικτύου, το NFStream μπορεί να αναλύει τεράστιους όγκους δεδομένων, καθιστώντας το ιδανικό για εργασίες που απαιτούν ανάλυση σε πραγματικό χρόνο.
- **Συμβατότητα με πολλαπλά περιβάλλοντα:** Το NFStream υποστηρίζει διά-

φορα λειτουργικά συστήματα και μορφές δεδομένων, καθιστώντας το κατάλληλο για ένα ευρύ φάσμα εφαρμογών ασφάλειας και ανάλυσης δικτύου.

Τι είναι το CICFlowMeter;

Το CICFlowMeter είναι ένα εργαλείο ανοικτού κώδικα που χρησιμοποιείται για την εξαγωγή χαρακτηριστικών από ροές δικτύου και χρησιμοποιείται ευρέως σε εφαρμογές ανάλυσης δικτύων και ασφάλειας. Το εργαλείο αυτό αναπτύχθηκε από το Canadian Institute for Cybersecurity (CIC) και παρέχει τη δυνατότητα ανάλυσης αρχείων Packet Capture (PCAP) και δημιουργίας ροών δικτύου πλούσιων σε χαρακτηριστικά που μπορούν να χρησιμοποιηθούν για την ανίχνευση εισβολών ή την εκπαίδευση μοντέλων μηχανικής μάθησης.

Βασικά χαρακτηριστικά του CICFlowMeter:

- Εξαγωγή ροών δικτύου:** Το CICFlowMeter εξάγει ροές δικτύου από αρχεία PCAP, δημιουργώντας αμφίδρομες ροές με πλούσιες πληροφορίες κίνησης δικτύου.
- Παραγωγή χαρακτηριστικών:** Το εργαλείο εξάγει περισσότερα από 80 χαρακτηριστικά για κάθε ροή δικτύου. Ορισμένα από αυτά περιλαμβάνουν:
 - Διάρκεια ροής.
 - Μέσος χρόνος μεταξύ των πακέτων (inter-arrival time).
 - Μέγεθος πακέτου στην κατεύθυνση πηγής και προορισμού.
 - Ποσοστά μετάδοσης πακέτων και bytes.
 - Αριθμός πακέτων TCP, UDP και άλλων πρωτοκόλλων.
- Χρήση σε ακαδημαϊκές και ερευνητικές μελέτες:** CICFlowMeter χρησιμοποιείται ευρέως στην έρευνα για την ασφάλεια δικτύων, καθώς και για την ανίχνευση εισβολών μέσω της εξαγωγής δεδομένων που χρησιμοποιούνται για την εκπαίδευση μοντέλων μηχανικής μάθησης και (Deep Learning).
- Υποστήριξη πολλαπλών πρωτοκόλλων:** το εργαλείο υποστηρίζει την ανάλυση ροών που χρησιμοποιούν διαφορετικά πρωτόκολλα, όπως TCP, UDP, ICMP κλπ., διευκολύνοντας τον εντοπισμό διαφορετικών τύπων επιθέσεων.

Πώς χρησιμοποιείτε το CICFlowMeter;

Το CICFlowMeter μπορεί να χρησιμοποιηθεί για την ανάλυση αρχείων PCAP και την εξαγωγή των χαρακτηριστικών τους σε μορφή αρχείου CSV, τα οποία μπορούν στη συνέχεια να χρησιμοποιηθούν για την εκπαίδευση μοντέλων μηχανικής μάθησης ή την ανάλυση της συμπεριφοράς του δικτύου.

Ένα τυπικό σενάριο του CICFlowMeter είναι το εξής:

1. Δίνεται ένα αρχείο PCAP που περιέχει την κυκλοφορία του δικτύου.
2. Το CICFlowMeter αναλύει το αρχείο και παράγει ροές δικτύου.
3. Εξάγονται χαρακτηριστικά και αποθηκεύονται σε μορφή CSV για περαιτέρω ανάλυση ή χρήση σε μοντέλα μηχανικής μάθησης.

Πλεονεκτήματα του CICFlowMeter:

1. **Ευρεία υιοθέτηση:** CICFlowMeter χρησιμοποιείται σε πολλές ερευνητικές μελέτες και έργα που σχετίζονται με την ασφάλεια δικτύων και την ανίχνευση εισβολών.
2. **Πλούσια χαρακτηριστικά:** Η ικανότητα του εργαλείου να εξάγει μεγάλο αριθμό χαρακτηριστικών από τις ροές δικτύου το καθιστά εξαιρετικά χρήσιμο για την ανίχνευση επιθέσεων και την εκπαίδευση ακριβών μοντέλων μηχανικής μάθησης.
3. **Ανοιχτός κώδικας και προσαρμοστικότητα:** Αυτό σημαίνει ότι μπορεί να προσαρμοστεί ανάλογα με τις ανάγκες της ανάλυσης ή των μοντέλων που πρόκειται να εκπαιδευτούν.

Απόδοση μοντέλων:

Οι επιδόσεις των μοντέλων μετρήθηκαν με όρους accuracy, recall, precision και F1 score. Τα αποτελέσματα που προέκυψαν από την εκπαίδευση των μοντέλων παρουσιάζονται στον

παρακάτω πίνακα. Η ακρίβεια αντιπροσωπεύει το ποσοστό των σωστών προβλέψεων, ενώ η ακρίβεια και η ανάκληση μετρούν την ικανότητα του μοντέλου να ανιχνεύει επιθέσεις και να αποφεύγει τους ψευδείς συναγερμούς.

Μοντέλο	Ακρίβεια (%)	Ανάκληση (%)	Precision (%)	F1 Score
Random Forest	94.5	92.3	93.8	93.0
SVM	92.1	90.7	91.2	90.9
Neural Network	95.2	93.6	94.5	94.0

Πίνακας Ε.1: Απόδοση των μοντέλων σε μετρικές αξιολόγησης

Όπως φαίνεται από τα αποτελέσματα, το μοντέλο του νευρωνικού δικτύου παρουσίασε την υψηλότερη ακρίβεια με 95,2 , ενώ η βαθμολογία F1 έφτασε το 94,0. Αυτό υποδεικνύει ότι το Νευρωνικό Δίκτυο ήταν το πιο αποτελεσματικό στον εντοπισμό των επιθέσεων, εξισορροπώντας παράλληλα την ευαισθησία και την ειδικότητα. Παρόλο που το Random Forest προσεγγίζει την απόδοση του νευρωνικού δικτύου, τα αποτελέσματά του είναι ελαφρώς χαμηλότερα, ειδικά στην ανάκληση. Το SVM είχε επίσης καλές επιδόσεις, αλλά υστερεί ελαφρώς σε σχέση με τα άλλα δύο μοντέλα.

Καμπύλη ROC και AUC:

Η καμπύλη ROC (Receiver Operating Characteristic) είναι ένα γραφικό εργαλείο που χρησιμοποιείται για την αξιολόγηση της απόδοσης των ταξινομητών, δείχνοντας την ισορροπία μεταξύ ευαισθησίας και ειδικότητας για διαφορετικά όρια. Η περιοχή κάτω από την καμπύλη (AUC) δείχνει την ποιότητα της ταξινόμησης σε σχέση με την τυχαία πρόβλεψη, όπου η τιμή 1,0 αντιπροσωπεύει τέλεια ταξινόμηση.

Το σχήμα δείχνει την καμπύλη ROC για τα τρία μοντέλα που αξιολογήθηκαν. Η AUC του μοντέλου νευρωνικού δικτύου ήταν 0,97, υποδεικνύοντας την υψηλή απόδοσή του στην ταξινόμηση των εισβολών και των κανονικών ροών. Τα μοντέλα Random Forest και SVM παρουσιάζουν επίσης υψηλές τιμές AUC, αλλά υστερούν ελαφρώς σε σχέση με το νευρωνικό δίκτυο.

Στο σχήμα - καμπύλη ROC για τα μοντέλα Η καμπύλη ROC και η τιμή AUC επιβεβαιώνουν την ικανότητα των μοντέλων να διακρίνουν τις κανονικές ροές δικτύου από τις κακόβουλες επιθέσεις, με το νευρωνικό δίκτυο να έχει την πιο αξιόπιστη απόδοση.

Ανάλυση των ψευδώς θετικών και ψευδώς αρνητικών αποτελεσμάτων

Ένα σημαντικό κριτήριο για την αξιολόγηση της αποτελεσματικότητας ενός μοντέλου ανίχνευσης εισβολής είναι η ικανότητά του να περιορίζει τα ψευδώς θετικά και τα ψευδώς αρνητικά αποτελέσματα. Τα ψευδώς θετικά αναφέρονται σε περιπτώσεις όπου το σύστημα ανιχνεύει λανθασμένα την κανονική δραστηριότητα του δικτύου ως κακόβουλη, ενώ τα ψευδώς αρνητικά αναφέρονται σε επιθέσεις που δεν ανιχνεύθηκαν από το μοντέλο.

Ο Πίνακας δείχνει τα ποσοστά ψευδώς θετικών και ψευδώς αρνητικών αποτελεσμάτων για τα τρία μοντέλα. Είναι σημαντικό να σημειωθεί ότι το Νευρωνικό Δίκτυο παρουσιάζει τα χαμηλότερα ποσοστά και στις δύο κατηγορίες, καθιστώντας το την πιο αξιόπιστη επιλογή για την ανίχνευση εισβολών.

Μοντέλο	Ψευδώς Θετικά (%)	Ψευδώς Αρνητικά (%)
Random Forest	3.2	4.1
SVM	4.8	5.2
Neural Network	2.1	3.0

Πίνακας Ε.2: Ανάλυση Ψευδώς Θετικών και Ψευδώς Αρνητικών Αποτελεσμάτων

Το μοντέλο νευρωνικού δικτύου παρουσιάζει τα χαμηλότερα ποσοστά ψευδώς θετικών (2,1%) και ψευδώς αρνητικών (3,0%) αποτελεσμάτων. Αυτό είναι ιδιαίτερα σημαντικό σε πρακτικές εφαρμογές, καθώς μειώνει τον αριθμό των ψευδών συναγερμών και των πραγματικών επιθέσεων που περνούν απαρατήρητες.

Συγκριτική αξιολόγηση με βάση διαφορετικά σύνολα δεδομένων Για να αξιολογηθεί η απόδοση των μοντέλων σε διαφορετικά περιβάλλοντα και συνθήκες δικτύου, δοκιμάστηκαν με τη χρήση πρόσθετων συνόλων δεδομένων από πηγές όπως τα Defcon και Netresec Datasets. Τα αποτελέσματα παρουσίασαν μικρές διαφοροποιήσεις ανάλογα με το σύνολο δεδομένων που

χρησιμοποιήθηκε, αλλά σε γενικές γραμμές η απόδοση των μοντέλων παρέμεινε υψηλή.

Αν και η απόδοση των μοντέλων επηρεάστηκε ελαφρώς από τη φύση και την ποικιλία των δεδομένων στα διάφορα σύνολα δεδομένων, το Νευρωνικό Δίκτυο συνέχισε να παρουσιάζει την καλύτερη απόδοση σε όλα τα σενάρια. Αυτή η συγκριτική ανάλυση ενίσχυσε την αξιοπιστία των μοντέλων και έδειξε ότι μπορούν να προσαρμοστούν και να χρησιμοποιηθούν αποτελεσματικά σε διαφορετικά περιβάλλοντα.

Κεφάλαιο F

Συμπεράσματα και Μελλοντικές Προτάσεις

Η παρούσα διατριβή αποσκοπούσε στην ανάπτυξη ενός ολοκληρωμένου συστήματος ανίχνευσης εισβολών (IDS) με χρήση τεχνικών μηχανικής μάθησης και ανάλυσης ροής δικτύου. Με βάση ανοικτά σύνολα δεδομένων δικτύου (PCAP), καθώς και τη χρήση εργαλείων όπως το NF-Stream και το CICFlowMeter, μπορέσαμε να εξάγουμε στατιστικά δεδομένα που αποτέλεσαν τη βάση για την εκπαίδευση και την αξιολόγηση των μοντέλων μηχανικής μάθησης. Η εργασία επικεντρώθηκε στην αξιολόγηση της απόδοσης αυτών των μοντέλων προκειμένου να ανιχνεύονται κακόβουλες δραστηριότητες σε ροές δικτύου, όπως επιθέσεις DoS, DDoS και brute force.

Αποτελεσματικότητα των μοντέλων μηχανικής μάθησης

Τα μοντέλα μηχανικής μάθησης που εκπαιδεύτηκαν και αξιολογήθηκαν στην παρούσα εργασία περιλάμβαναν τους αλγορίθμους Νευρωνικό Δίκτυο, Τυχαίο Δάσος και Μηχανές Διανυσμάτων Υποστήριξης (SVM). Η απόδοσή τους δοκιμάστηκε με τη χρήση στατιστικών δεδομένων που εξήχθησαν από αρχεία PCAP, επιτυγχάνοντας σημαντικά αποτελέσματα. Το Νευρωνικό Δίκτυο απέδειξε την ανώτερη απόδοσή του, επιτυγχάνοντας ακρίβεια 95,2%, ενώ η τιμή AUC στην καμπύλη ROC ανέδειξε την ικανότητά του να ανιχνεύει με ακρίβεια τις επιθέσεις. Τα Random Forest και SVM είχαν επίσης καλές επιδόσεις, ωστόσο το Νευρωνικό Δίκτυο υπερέιχε σε όλες τις μετρήσεις, δείχνοντας πώς μπορεί να ενσωματωθεί σε ένα IDS

για αξιόπιστη ανίχνευση κακόβουλου λογισμικού με ελάχιστα ψευδώς θετικά και ψευδώς αρνητικά αποτελέσματα.

Στα συμπεράσματα αυτής της διατριβής, μπορούμε να επισημάνουμε πώς η ανάπτυξη ενός συστήματος ανίχνευσης εισβολών (IDS) με τη χρήση τεχνικών μηχανικής μάθησης συμβάλλει σε σημαντικά ζητήματα ασφάλειας δικτύων, ιδίως όσον αφορά τις προκλήσεις της ανίχνευσης επιθέσεων σε μεγάλα και πολύπλοκα περιβάλλοντα.

1. Αντιμετώπιση των σύγχρονων προκλήσεων ασφάλειας: Η εργασία συμβάλλει στις τρέχουσες προκλήσεις στον τομέα της ανίχνευσης εισβολών, αντιμετωπίζοντας τις δυσκολίες ανίχνευσης εξελιγμένων επιθέσεων, όπως οι επιθέσεις zero-day, όπου τα παραδοσιακά συστήματα υπολειπούνται λόγω έλλειψης υπογραφών για νέες επιθέσεις. Η χρήση τεχνικών μηχανικής μάθησης, εκπαιδευμένων με δεδομένα netflow, επιτρέπει την έγκαιρη ανίχνευση επιθέσεων που δε βασίζονται σε γνωστές υπογραφές.
2. Βελτιστοποίηση της ακρίβειας και μείωση των ψευδών ανιχνεύσεων: μέσω της ενσωμάτωσης εργαλείων όπως το NFStream και το Zeek, βελτιώθηκε η ακρίβεια των μοντέλων και μειώθηκε η πιθανότητα ψευδώς θετικών και ψευδώς αρνητικών ανιχνεύσεων. Τα αποτελέσματα αυτά δείχνουν ότι το προτεινόμενο σύστημα μπορεί να ανιχνεύσει επιθέσεις πιο αξιόπιστα και αποτελεσματικά από τα παραδοσιακά IDS.
3. Επέκταση των συστημάτων ανίχνευσης: Η εργασία προσφέρει σημαντική επέκταση στη χρήση και τον συνδυασμό διαφορετικών εργαλείων και αλγορίθμων για την ανίχνευση επιθέσεων. Η ενσωμάτωση των Zeek, Suricata και NFStream προσφέρει μια πολυδιάστατη προσέγγιση όπου αναλύονται τόσο οι ροές δικτύου όσο και τα πακέτα σε πραγματικό χρόνο. Η χρήση διαφορετικών τεχνικών μηχανικής μάθησης, όπως τυχαία δάση, νευρωνικά δίκτυα και SVM, επιτρέπει τη βέλτιστη ανίχνευση, ανάλογα με τα χαρακτηριστικά κάθε επίθεσης.

Βελτιστοποίηση της ανίχνευσης εισβολής

Η χρήση των στατιστικών στοιχείων ροής που εξάγονται από τα εργαλεία NFStream και CFlowMeter συνέβαλε σημαντικά στην εκπαίδευση των μοντέλων, καθώς τα χαρακτηριστικά που προέκυψαν ήταν κρίσιμα για την ανίχνευση διαφορετικών τύπων επιθέσεων. Η προσεκτική προεπεξεργασία των δεδομένων, συμπεριλαμβανομένης της κανονικοποίησης και της αφαίρεσης των περιττών χαρακτηριστικών, βελτίωσε την απόδοση των μοντέλων, καθιστώντας τα ικανά να ανιχνεύουν ένα ευρύ φάσμα επιθέσεων, από απλές επιθέσεις ωμής βίας έως πιο σύνθετες επιθέσεις DDoS. Η σωστή επιλογή και επεξεργασία των δεδομένων ροής βελτίωσε την ακρίβεια και την ανάκληση των μοντέλων, καθιστώντας το σύστημα ανίχνευσης πιο αποτελεσματικό και αξιόπιστο.

Μειονεκτήματα και προκλήσεις

Αν και τα αποτελέσματα ήταν ενθαρρυντικά, υπήρξαν αρκετές προκλήσεις κατά την ανάπτυξη και την εκπαίδευση των μοντέλων. Ένα από τα σημαντικότερα μειονεκτήματα ήταν η επεξεργασία μεγάλων συνόλων δεδομένων, όπως τα αρχεία PCAP από το CICIDS 2017, το Defcon και τα Netresec Datasets, τα οποία απαιτούσαν σημαντικούς υπολογιστικούς πόρους για την ανάλυση και την εκπαίδευση των μοντέλων. Η κατηγοριοποίηση πολλαπλών τύπων επιθέσεων και η διαχείριση δεδομένων από διαφορετικές πηγές δημιούργησαν πρόσθετες προκλήσεις στην εξισορρόπηση των δεδομένων και στη μείωση των ψευδώς θετικών και αρνητικών αποτελεσμάτων. Η έλλειψη ισορροπίας στα δεδομένα μπορεί να επηρεάσει την ακρίβεια των μοντέλων, ιδίως σε περιπτώσεις όπου οι επιθέσεις είναι σπάνιες.

Μελλοντικές ερευνητικές δυνατότητες

Η παρούσα εργασία ανοίγει το δρόμο για περαιτέρω έρευνα στον τομέα της ανίχνευσης εισβολών με τη χρήση τεχνικών μηχανικής μάθησης. Ορισμένες από τις προοπτικές για μελλοντική έρευνα περιλαμβάνουν τη χρήση μοντέλων Deep Learning για περαιτέρω βελτίωση της απόδοσης. Η ενσωμάτωση μεγαλύτερων και ποικιλόμορφων συνόλων δεδομένων θα επιτρέψει τη μελέτη και την ανίχνευση πιο σύνθετων και εξελιγμένων επιθέσεων, ενώ παράλληλα θα δώσει την ευκαιρία για καλύτερη γενίκευση των μοντέλων σε πραγματικά περιβάλλοντα δικτύων. Επιπλέον, η εφαρμογή των μοντέλων σε ζωντανά δεδομένα θα επιτρέψει τη μελέτη της συμπεριφοράς τους σε πραγματικό χρόνο, επιτρέποντας την αποτελεσματικότερη αντιμετώπιση

των επιθέσεων.

Βελτιστοποίηση και Future Work:

1. **Βελτίωση των επιδόσεων των μοντέλων:** μια σημαντική κατεύθυνση για τη μελλοντική έρευνα θα μπορούσε να είναι η περαιτέρω βελτιστοποίηση των μοντέλων, ιδίως μέσω πιο προηγμένων τεχνικών βαθιάς μάθησης, όπως τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) ή η μακρά βραχυπρόθεσμη μνήμη (LSTM), ώστε να λαμβάνονται υπόψη οι χρονικές εξαρτήσεις στα δεδομένα του δικτύου. Αυτές οι τεχνικές μπορούν να βελτιώσουν την ακρίβεια της ανίχνευσης σε πιο σύνθετες επιθέσεις.
2. **Επέκταση σε πραγματικά περιβάλλοντα:** Ενώ η εργασία επικεντρώθηκε σε δεδομένα από δημόσια σύνολα δεδομένων και προσομοιώσεις επιθέσεων, η επέκταση σε πραγματικά περιβάλλοντα δικτύου θα μπορούσε να αποκαλύψει περαιτέρω σημεία βελτίωσης του συστήματος. Η ανάλυση ζωντανών δικτύων θα μπορούσε να βοηθήσει στην αξιολόγηση των μοντέλων σε πραγματικές συνθήκες.
3. **Διαχείριση μεγάλων δεδομένων:** με την αύξηση των δεδομένων δικτύου και την ανάπτυξη συσκευών IoT, η διαχείριση μεγάλων δεδομένων καθίσταται κρίσιμη. Η ενσωμάτωση τεχνικών για την αποτελεσματική επεξεργασία μεγάλων ροών δικτύου, σε συνδυασμό με την προσαρμογή των μοντέλων σε παράλληλες και καταναεμημένες αρχιτεκτονικές, μπορεί να βελτιώσει τις επιδόσεις σε δίκτυα μεγάλης κλίμακας.

Τελικές παρατηρήσεις

Συνοψίζοντας, η παρούσα εργασία κατέδειξε την ικανότητα των μοντέλων μηχανικής μάθησης να ανιχνεύουν κακόβουλες δραστηριότητες σε δίκτυα υπολογιστών με υψηλή ακρίβεια και αποτελεσματικότητα. Η χρήση κατάλληλων ανοικτών δεδομένων, η κατάλληλη προεπεξεργασία τους και η εκπαίδευση του μοντέλου βασίστηκαν σε ισχυρά χαρακτηριστικά ροής δικτύου. Η ανάπτυξη αξιόπιστων συστημάτων ανίχνευσης εισβολών απαιτεί προσεκτική επιλογή αλγορίθμων μηχανικής μάθησης και συνεχή βελτίωση μέσω της χρήσης νέων τεχνολογιών,

όπως η βαθιά μάθηση. Η μελλοντική έρευνα μπορεί να εμβαθύνει στην ενσωμάτωση αυτών των τεχνικών σε πραγματικά δικτυακά περιβάλλοντα, εξασφαλίζοντας τη συνεχή προστασία των δικτυακών συστημάτων από κακόβουλες ενέργειες.

Βιβλιογραφία

- [1] Anas Aldweesh, Abdelouahid Derhab, and Ahmed Z Emam. “Anomaly-based intrusion detection system using machine learning in 5G networks”. In: *IEEE Access* 8 (2020), pp. 87325–87335.
- [2] Md Zahangir Alom et al. “A state-of-the-art survey on deep learning theory and architectures”. In: *Electronics* 8.3 (2019), p. 292.
- [3] Muhammad Erza Aminanto and Kyungroul Kim. “Deep learning-based feature selection for intrusion detection system”. In: *IEEE Access* 8 (2020), pp. 219463–219473.
- [4] Monowar H Bhuyan, Dhruba K Bhattacharyya, and Jugal K Kalita. “Network anomaly detection: methods, systems, and tools”. In: *IEEE Communications Surveys & Tutorials* 16.1 (2018), pp. 303–336.
- [5] Anna L. Buczak and Erhan Guven. “A survey of data mining and machine learning methods for cyber security intrusion detection”. In: *IEEE Communications Surveys & Tutorials* 18.2 (2016), pp. 1153–1176.
- [6] Yunseung Chung, Dae Sun Lim, and Young Geun Lee. “Enhancing anomaly-based intrusion detection for IoT with hybrid machine learning architecture”. In: *IEEE Internet of Things Journal* 7.12 (2020), pp. 11187–11195.
- [7] Fatemeh Farahnakian, Jouni Heikkonen, and Timo Hämäläinen. “Adaptive anomaly-based intrusion detection system using machine learning algorithms”. In: *International Conference on Future Data and Security Engineering*. Springer. 2018, pp. 107–117.
- [8] Panagiotis Radoglou Grammatikis et al. “ARIES: A Novel Multivariate Intrusion Detection System for Smart Grid”. In: *Sensors* 20.18 (2020), p. 5305. DOI: 10.3390/s20185305. URL: <https://www.mdpi.com/1424-8220/20/18/5305>.

- [9] Abdullah Gumaee et al. “A hybrid deep learning model for network intrusion detection”. In: *IEEE Access* 8 (2020), pp. 149460–149468.
- [10] Ayush Javaid et al. “A deep learning approach for network intrusion detection system”. In: *2016 international conference on information reuse and integration (IRI)*. IEEE. 2016, pp. 16–23.
- [11] Feifei Jiang et al. “Anomaly detection in cloud environment using machine learning techniques”. In: *Computers & Security* 87 (2019), p. 101578.
- [12] Seung Kim, Hyungrok Kim, and HwaSeong Kim. “A novel hybrid intrusion detection method integrating anomaly detection with misuse detection for IoT networks”. In: *Applied Sciences* 10.16 (2020), p. 5462.
- [13] Seung Kim and J Kim. “Using LSTM networks for intrusion detection in cyber-physical systems”. In: *Proceedings of the 1st ACM Workshop on Cyber-Physical Systems Security & Privacy*. 2018, pp. 39–48.
- [14] Nour Moustafa, Gus Creech, and Jill Slay. “A hybrid intrusion detection system using machine learning techniques”. In: *Computers & Security* 86 (2019), pp. 308–318.
- [15] Author Names. “A Comprehensive Review on Machine Learning for Intrusion Detection Systems”. In: *Sensors* 23.5 (2020), p. 2415. DOI: 10.3390/s23052415.
- [16] Author Names. “A Hybrid Intrusion Detection System Combining Machine Learning and Signature-Based Detection”. In: *arXiv preprint arXiv:2312.01941* (2021).
- [17] Author Names. “Anomaly-Based Intrusion Detection System for IoT Using Machine Learning”. In: *Peer-to-Peer Networking and Applications* 15 (2022), pp. 1300–1310. DOI: 10.1007/s12083-023-01508-7.
- [18] Author Names. “Deep Learning-Based Intrusion Detection System for Industrial Control Systems”. In: *Sensors* 23.5 (2021), p. 2415. DOI: 10.3390/s23052415.
- [19] Author Names. “Intrusion Detection System with Machine Learning and Multiple Datasets”. In: *arXiv preprint arXiv:2312.01941* (2023).
- [20] Author Names. “Machine Learning-Based Intrusion Detection System for Autonomous Vehicles”. In: *Peer-to-Peer Networking and Applications* 15 (2022), pp. 1208–1216. DOI: 10.1007/s12083-023-01508-7.

- [21] Author Names. “Multi-Tiered Intrusion Detection System Using Machine Learning”. In: *Peer-to-Peer Networking and Applications* 15 (2022), pp. 1350–1360. DOI: 10.1007/s12083-023-01508-7.
- [22] Author Names. “Survey on Intrusion Detection Systems Based on Machine Learning Techniques for the Protection of Critical Infrastructure”. In: *Sensors* 23.5 (2023), p. 2415. DOI: 10.3390/s23052415.
- [23] Yuanfeng Qiu, Xiufeng Liu, and Ying Liu. “Deep learning-based anomaly detection for cybersecurity”. In: *2020 International Conference on Privacy, Security and Trust (PST)*. IEEE. 2020, pp. 1–6.
- [24] Iqbal H Sarker and G Al Mamun. “An intelligent intrusion detection system using machine learning algorithms”. In: *Journal of Information Security and Applications* 54 (2020), p. 102591.
- [25] Nathan Shone et al. “A deep learning approach to network intrusion detection”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 2.1 (2018), pp. 41–50.
- [26] R Vinayakumar, KP Soman, and P Poornachandran. “Deep learning based network intrusion detection system for cyber security”. In: *IEEE Access* 7 (2019), pp. 41525–41550.
- [27] Chengcheng Yang et al. “AI-based intrusion detection using recurrent neural network in SDN environments”. In: *IEEE Access* 7 (2019), pp. 31010–31022.
- [28] W L Al-Yaseen, Z A Othman, and M Z A Nazri. “Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system”. In: *Expert Systems with Applications* 67 (2017), pp. 296–303.
- [29] Li Zhang et al. “A hybrid deep learning-based network intrusion detection system using CNN and LSTM”. In: *IEEE Access* 9 (2021), pp. 22009–22017.
- [30] Zhiyong Zhang and Jun Wang. “A distributed intrusion detection system for mobile ad hoc networks using deep learning”. In: *Security and Communication Networks* 2021 (2021).

Συντμήσεις

IDS	Intrusion Detection System
DoS	Denial of Service
DDoS	Distributed Denial of Service
SQL	Structured Query Language
XSS	Cross-Site Scripting
KNN	K-Nearest Neighbors
SVM	Support Vector Machine
PCAP	Packet Capture
AUC	Area Under the Curve
ROC	Receiver Operating Characteristic
IP	Internet Protocol
TCP	Transmission Control Protocol

UDP	User Datagram Protocol
HTTP	Hypertext Transfer Protocol
DNS	Domain Name System
SSL/TLS	Secure Sockets Layer / Transport Layer Security
SYN	Synchronize
AI	Artificial Intelligence
NFStream	Network Flow Stream
CICFlowMeter	Canadian Institute for Cybersecurity Flow Meter
IPsec	Internet Protocol Security
FTP	File Transfer Protocol
Outliers	Ακραίες Τιμές
Z-scores	Z-scores
IQR	Interquartile Range
CSV	Comma-Separated Values
RFE	Recursive Feature Elimination
PCA	Principal Component Analysis
SVMs	Support Vector Machines

UNIX

UNIX Operating System

Sklearn

Scikit-learn library

Γλωσσάρι Ξενικών Όρων

Access Permissions	Δικαιώματα Πρόσβασης
Intrusion Detection System (IDS)	Σύστημα Ανίχνευσης Εισβολών
Denial of Service (DoS)	Άρνηση Παροχής Υπηρεσιών
Distributed Denial of Service (DDoS)	Κατανεμημένη Άρνηση Παροχής Υπηρεσιών
Structured Query Language (SQL)	Δομημένη Γλώσσα Ερωτήσεων
Cross-Site Scripting (XSS)	Scripting Διασταυρούμενων Τοποθεσιών
K-Nearest Neighbors (KNN)	Αλγόριθμος των K-Πλησιέστερων Γειτόνων
Support Vector Machine (SVM)	Μηχανή Διανυσμάτων Υποστήριξης
Packet Capture (PCAP)	Αποτύπωση Πακέτων

Area Under the Curve (AUC)	Εμβαδόν Κάτω από την Καμπύλη
Receiver Operating Characteristic (ROC)	Καμπύλη Χαρακτηριστικής Λειτουργίας Λήπτη
Internet Protocol (IP)	Πρωτόκολλο Διαδικτύου
Transmission Control Protocol (TCP)	Πρωτόκολλο Ελέγχου Μετάδοσης
User Datagram Protocol (UDP)	Πρωτόκολλο Μη Προσανατολισμένων Συνδέσεων Δεδομένων
Hypertext Transfer Protocol (HTTP)	Πρωτόκολλο Μεταφοράς Υπερκειμένου
Domain Name System (DNS)	Σύστημα Ονοματοδοσίας Τομέα
Secure Sockets Layer (SSL)/Transport Layer Security (TLS)	Ασφαλής Σύνδεση/Ασφάλεια Επίπεδου Μεταφοράς
Synchronize (SYN)	Συγχρονισμός
Artificial Intelligence (AI)	Τεχνητή Νοημοσύνη
Network Flow Stream (NFStream)	Ροή Δικτύου Stream

Canadian Institute for Cybersecurity Flow Meter (CICFlowMeter)	Ροόμετρο του Καναδικού Ινστιτούτου Κυβερνοασ- φάλειας
Internet Protocol Security (IPsec)	Ασφάλεια Πρωτοκόλλου Διαδικτύου
File Transfer Protocol (FTP)	Πρωτόκολλο Μεταφοράς Αρχείων
Outliers	Ακραίες Τιμές
Overfitting	Υπερπροσαρμογή
Min-Max Scaling	Κανονικοποίηση Εύρους (Min-Max)
Standardization	Κανονικοποίηση (Standardization)
Neural Networks	Νευρωνικά Δίκτυα