

Ατομική Διπλωματική Εργασία

**ΣΥΣΤΗΜΑ ΑΝΙΧΝΕΥΣΗΣ ΕΠΙΘΕΣΕΩΝ ΜΕ ΤΗ ΧΡΗΣΗ
ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΣΕ ΔΙΚΤΥΑ IoT**

Χρίστος Ανδρέου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ



ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2020

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Σύστημα Ανίχνευσης Επιθέσεων με τη Χρήση Μηχανικής Μάθησης σε Δίκτυα IoT

Χρίστος Ανδρέου

Επιβλέπων Καθηγητής
Βάσος Βασιλείου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2020

Ευχαριστίες

Αυτή η έρευνα πραγματοποιήθηκε υπό την επίβλεψη του Αναπληρωτή Καθηγητή Δρ. Βάσου Βασιλείου του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου. Οφείλω πολλές ευχαριστίες αφού μου παρείχε πολλά υλικά για να μελετήσω που ήταν θεμελιώδη για το μέλλον μου διαδικασία σε αυτήν την έρευνα. Κατά τη διάρκεια των εβδομαδιαίων συναντήσεων με τον επιβλέπων καθηγητή μου συζητούσαμε μελλοντικά σχέδια και αποφασίζαμε για επιλογές για την περαιτέρω ανάπτυξη της έρευνας μου. Χωρίς τη βοήθεια και την αφοσίωσή του αυτό η έρευνα δεν θα μπορούσε να επιτευχθεί.

Ιδιαίτερες ευχαριστίες στο Μιχάλη Αγαθοκλέους, ερευνητή στο Τμήμα Πληροφορικής του Πανεπιστημίου Κύπρου. Με τη βοήθειά του ήμουν σε θέση να καταλάβω σχετικές πληροφορίες για διάφορους αλγορίθμους μηχανικής μάθησης. Θέλω να εκφράσω την ευγνωμοσύνη μου για την προσπάθεια του για παροχή βοηθητικού εργαλείου στο Τμήμα Πληροφορικής.

Επιπλέον, θέλω να πω ευχαριστώ στην οικογένειά μου που παρέμειναν δίπλα μου κατά τη διάρκεια της φοίτησης μου στο πανεπιστήμιο. Η ηθική τους υποστήριξη ήταν εξαιρετική και χωρίς αυτή δεν θα είχα τα ίδια αποτελέσματα και δύναμη για να συνεχίσω.

Περίληψη

Τα περισσότερα συστήματα ανίχνευσης εισβολής χρησιμοποιούν αλγόριθμους επιβλεπόμενης μηχανικής μάθησης που τους επιτρέπουν να εντοπίζουν μόνο καταγεγραμμένους τύπους κακόβουλων επιθέσεων. Αυτή η έρευνα εφαρμόζει μια θεμελιωδώς διαφορετική προσέγγιση στο πρόβλημα, εκμεταλλευόμενος τον Isolation Forest, έναν αλγόριθμο μη επιβλεπόμενης μηχανικής μάθησης σε ένα νέο πλαίσιο. Ένα από τα πιο σημαντικά πλεονεκτήματα του αλγορίθμου είναι ότι μπορεί να αναγνωρίσει και να καταγράψει νέα μοντέλα εισβολής [1].

Οι περισσότερες υπάρχουσες προσεγγίσεις βασισμένες στο μοντέλο για ανίχνευση ανωμαλιών δημιουργούν ένα προφίλ φυσιολογικών περιπτώσεων και, στη συνέχεια, εντοπίζουν περιπτώσεις που δεν συμμορφώνονται με το κανονικό προφίλ ως ανωμαλίες. Αυτή η έρευνα προτείνει μια βασικά διαφορετική μέθοδο βασισμένη στο μοντέλο που απομονώνει ρητά ανωμαλίες αντί να δημιουργεί προφίλ φυσιολογικών σημείων [2].

Η χρήση της απομόνωσης επιτρέπει στην προτεινόμενη μέθοδο, τον iForest, να εκμεταλλευτεί την υπο-δειγματοληψία σε βαθμό που δεν είναι εφικτό σε υπάρχουσες μεθόδους, δημιουργώντας έναν αλγόριθμο που έχει γραμμική πολυπλοκότητα χρόνου με χαμηλή σταθερά και χαμηλή απαίτηση μνήμης.

Ο iForest λειτουργεί επίσης καλά σε προβλήματα υψηλών διαστάσεων που έχουν μεγάλο αριθμό άσχετων χαρακτηριστικών και σε καταστάσεις όπου το σετ εκπαίδευσης δεν περιέχει ανωμαλίες.

Περιεχόμενα

Κεφάλαιο 1	Εισαγωγή.....	1
	1.1 Γενική Εισαγωγή	1
	1.2 Ορισμός του προβλήματος	1
	1.3 Σκοπός	3
	1.4 Σχετική δουλειά	3
	1.5 Μεθοδολογία και αποτελέσματα	4
Κεφάλαιο 2	Υπόβαθρο.....	5
	2.1 Κυκλοφορία δικτύων IoT	5
	2.2 Αλγόριθμοι Μηχανικής Μάθησης	7
	2.3 Ο αλγόριθμος Isolation Forest	9
Κεφάλαιο 3	Δομή προγράμματος.....	11
	3.1 Δομή Καταγραφής κυκλοφορίας δικτύου	11
	3.2 Δομή Προετοιμασίας Δεδομένων	13
	3.3 Δομή Διαδικασίας Εκπαίδευσης	14
	3.4 Δομή Ανίχνευσης επιθέσεων	15
	3.5 Δημιουργία Εργαλείου	16
	3.6 Πολυπλοκότητα	17
Κεφάλαιο 4	Υλοποίηση Συστήματος.....	18
	4.1 Καταγραφή κυκλοφορίας δικτύου	18
	4.2 Προετοιμασία Δεδομένων	25
	4.3 Διαδικασία Εκπαίδευσης	25
	4.4 Ανίχνευση επιθέσεων	31
Κεφάλαιο 5	Αποτελέσματα.....	38
	5.1 Μετρήσεις	38
	5.2 Σύγκριση αποτελεσμάτων	39

Κεφάλαιο 6	Συμπεράσματα.....	40
6.1	Παρατηρήσεις	40
6.2	Συμπεράσματα	40
6.3	Μελλοντική Εργασία	41
Βιβλιογραφία.....		42
Παράρτημα Α.....		A-1

Κεφάλαιο 1

Εισαγωγή

1.1 Γενική Εισαγωγή	1
1.2 Ορισμός του προβλήματος	1
1.3 Σκοπός	3
1.4 Σχετική δουλειά	3
1.5 Μεθοδολογία και αποτελέσματα	4

1.1 Γενική Εισαγωγή

Σε αυτό το κεφάλαιο περιγράφεται ο ορισμός του προβλήματος που διερευνήθηκε και ο λεπτομερής σκοπός για τον οποίο δημιουργήθηκε αυτό το έργο.

Το σύστημα ανίχνευσης εισβολής (IDS) είναι ένα σημαντικό συστατικό σε ένα σύστημα ασφαλείας υπολογιστή. Το καθήκον ενός IDS είναι να εντοπίζει κακόβουλες δραστηριότητες στο σύστημα του υπολογιστή, επομένως να επιτρέπει γρήγορη αντίδραση στις επιθέσεις. Η χρήση της τεχνολογίας εμπορικών συστημάτων ανίχνευσης εισβολής (IDS) είναι πολύ υψηλή για τα παραδοσιακά δίκτυα και τα IDS είναι πλέον τυπικός εξοπλισμός για μεγάλα δίκτυα. Ωστόσο, όταν πρόκειται για δίκτυα IoT, η χρήση τους είναι συνήθως περιορισμένη, καθώς οι περισσότερες συσκευές IoT που είναι συνδεδεμένες είναι μηχανήματα χαμηλής ισχύος με περιορισμένες δυνατότητες και υπολογιστική ισχύ.

1.2 Ορισμός του προβλήματος

Τα IDS χρησιμοποιούνται για την ανάλυση δεδομένων δικτύου και τον εντοπισμό ανωμαλιών συμπεριφορών στο δίκτυο. Τα IDS χωρίζονται γενικά σε δύο κατηγορίες: συστήματα ανίχνευσης βάσει υπογραφής (signature-based) και ανωμαλίας (anomaly-based)

[3]. Τα συστήματα ανίχνευσης εισβολής βάσει υπογραφής (signature-based) έχουν σχεδιαστεί για να ανιχνεύουν την εισβολή δημιουργώντας βιβλιοθήκες χαρακτήρων συμπεριφοράς ανωμαλίας και ταιριάζοντας δεδομένα δικτύου. Αυτά τα IDS έχουν υψηλό ποσοστό ανίχνευσης, αλλά είναι δύσκολο να εντοπιστούν νέες επιθέσεις στο δίκτυο. Τα συστήματα ανίχνευσης εισβολής που βασίζονται σε ανωμαλίες (anomaly-based) καθιερώνουν μοντέλα σύμφωνα με την κανονική συμπεριφορά του δικτύου και διεξάγουν ανίχνευση εισβολής βάσει του εάν οι συμπεριφορές αποκλίνουν από την κανονική συμπεριφορά. Τέτοια IDS έχουν εξαιρετική αποδοτικότητα αναγνώρισης για άγνωστους τύπους συμπεριφοράς ανωμαλίας, αλλά το συνολικό ποσοστό ανίχνευσής τους είναι χαμηλό και έχει υψηλό ποσοστό ψευδούς συναγερμού.

Προκειμένου να βελτιωθεί το ποσοστό ανίχνευσης των IDS και να μειωθεί ο ψευδής ρυθμός συναγερμού, οι ερευνητές έχουν κάνει πολλή δουλειά, προσπαθώντας να εφαρμόσουν μια ποικιλία μεθόδων εξόρυξης δεδομένων και μηχανικής μάθησης σε IDS.

Ωστόσο, η τεράστια ποσότητα δεδομένων δικτύου και η μη ισορροπημένη κατανομή φυσιολογικών και ανώμαλων συμπεριφορών οδηγούν σε προβλήματα χαμηλού ρυθμού ανίχνευσης και υψηλού ψευδούς ρυθμού συναγερμού στα περισσότερα IDS. Σε αυτή την έρευνα, το προτεινόμενο IDS χρησιμοποιεί επιλογή χαρακτηριστικών (feature selection) στα αρχικά δεδομένα για να μειώσει την ποσότητα των πληροφοριών. Η επιλογή χαρακτηριστικών (feature selection) είναι η αναζήτηση χαρακτηριστικών που αντικατοπτρίζουν καλύτερα τη διαφορά μεταξύ ανώμαλων συμπεριφορών και κανονικών συμπεριφορών και διαγράφουν άχρηστα χαρακτηριστικά για να βελτιώσουν την απόδοση ανίχνευσης του IDS.

Δυστυχώς, υπάρχουν περισσότερα προβλήματα που πρέπει να αντιμετωπιστούν τα οποία αφορούν ειδικά τα δίκτυα IoT. Η φύση αυτών των συσκευών (IoT) είναι ότι κάθε μία έχει σχεδιαστεί λαμβάνοντας υπόψη πολύ συγκεκριμένους σκοπούς και η λειτουργικότητά τους προσαρμόζεται στις ανάγκες αυτής της συγκεκριμένης εφαρμογής.

Αυτό φυσικά επεκτείνεται πέρα από τη λειτουργικότητά τους αλλά και με τον τρόπο επικοινωνίας τους, τόσο στο δίκτυο όσο και μεταξύ τους. Αυτό σημαίνει ότι δεν υπάρχει ομοιομορφία ή μοτίβα στην κίνηση του δικτύου τους (network traffic).

Επεκτείνεται επίσης στις δυνατότητές τους και πιο συγκεκριμένα στις υπολογιστικές τους δυνατότητες. Αυτές οι συσκευές είναι συνήθως μηχανήματα χαμηλής ισχύος και, στις περισσότερες περιπτώσεις, η υπολογιστική ισχύς τους είναι περιορισμένη καθιστώντας

ακόμη πιο δύσκολο το να υπάρχει ένα επιπλέον επίπεδο ασφάλειας στο παρασκήνιο, όπως ένα IDS.

1.3 Σκοπός

Ο σκοπός αυτής της έρευνάς είναι να βρει μια βιώσιμη λύση στο πρόβλημα της επικοινωνίας συσκευών IoT. Πιο συγκεκριμένα, ένα IDS που να μπορεί να αναγνωρίσει και να ανιχνεύσει τόσο γνωστές όσο και άγνωστες επιθέσεις στην κυκλοφορία του δικτύου και να είναι αρκετά ελαφρύ και γρήγορο για να τρέχει στις πραγματικές συσκευές IoT.

Ως μέρος αυτού του έργου, ένας από τους στόχους είναι η έρευνα διαφόρων συνδυασμών χαρακτηριστικών που εξάγονται από τη διαδικασία εξαγωγής χαρακτηριστικών (feature extraction) στα αρχικά δεδομένα, την κυκλοφορία δικτύου, για τη μείωση του όγκου των πληροφοριών που χρησιμοποιούνται.

Τέλος, αφού αποφασίσει ο συνδυασμός των χαρακτηριστικών που ταιριάζουν καλύτερα και η ρύθμιση του αλγορίθμου, πρέπει να πραγματοποιηθεί δοκιμή αυτού του μοντέλου για να εξασφαλιστεί η επιτυχία της προτεινόμενης προσέγγισης για ασφαλή επικοινωνία σε δίκτυα IoT.

1.4 Σχετική δουλειά

Στο παρελθόν έχει γίνει πολλή δουλειά σχετικά με τα IDS τόσο για τα παραδοσιακά όσο και για τα δίκτυα IoT χρησιμοποιώντας τεχνικές μηχανικής μάθησης. Παρόλο που ο αλγόριθμος Isolation Forest έχει χρησιμοποιηθεί σε παραδοσιακά δίκτυα σε διάφορες μορφές από πολλούς ερευνητές, δεν έχει γίνει έρευνα για τη χρήση αυτού του αλγορίθμου σε δίκτυα IoT.

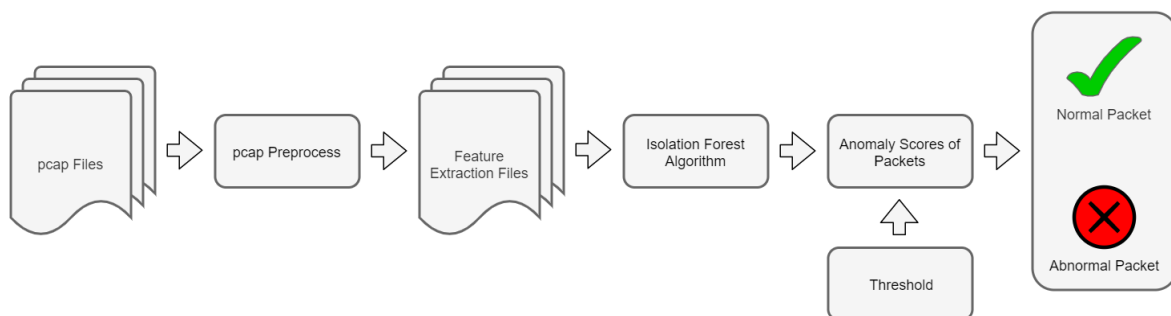
Μερικές από τις σχετικές έρευνες για το iForest που έχουν γίνει, είναι το απλό μοντέλο του αλγορίθμου Isolation Forest για ανίχνευσης ανωμαλιών [2].

Επίσης, μία πιο πρόσφατη έρευνα που έχει γίνει, είναι ένα εκτεταμένο μοντέλο του αλγορίθμου Isolation Forest για εντοπισμό ανώμαλης συμπεριφοράς χρήστη [4]. Παρόλου που η συγκεκριμένη έρευνα δεν έχει σχέση με δίκτυα, η χρήση του αλγορίθμου εξακολουθεί να είναι σχετική.

Ακόμα μια πρόσφατη έρευνα που έχει γίνει, είναι ένα υβριδικό μοντέλο του αλγορίθμου Isolation Forest για σύστημα ανίχνευσης επιθέσεων σε παραδοσιακά δίκτυα [5].

1.5 Μεθοδολογία και αποτελέσματα

Σε αυτή την έρευνα, αναπτύχθηκε ένα σύστημα εντοπισμού εισβολής (IDS) χρησιμοποιώντας τον αλγόριθμο Isolation Forest. Το προτεινόμενο πλαίσιο φαίνεται στο Σχήμα 1.1. Το πρόγραμμα ανάλυσης pcap προ-επεξεργάζεται όλη την κυκλοφορία και το σύστημα εξαγάγει τα χαρακτηριστικά που έχουν οριστεί και δημιουργεί ένα βασικό μοντέλο κυκλοφορίας δημιουργώντας μια συλλογή δέντρων Isolation Forest. Όταν έρχεται ένα νέο πακέτο, χαρτογραφείται σε καθένα από αυτά τα δέντρα του Isolation Forest και υπολογίζεται μια βαθμολογία ανωμαλίας (anomaly score). Εάν η βαθμολογία της ανωμαλίας (anomaly score) είναι κάτω από το όριο, θεωρείται φυσιολογικό, διαφορετικά καταγράφεται μια ανώμαλη συμπεριφορά και αυτό το πακέτο επισημαίνεται ως ανώμαλο.



Σχήμα. 1.1: Το προτεινόμενο πλαίσιο

Κεφάλαιο 2

Υπόβαθρο

2.1 Κυκλοφορία δικτύων IoT	5
2.2 Αλγόριθμοι Μηχανικής Μάθησης	7
2.3 Ο αλγόριθμος Isolation Forest	9

2.1 Κυκλοφορία δικτύων IoT

Πώς διαφέρει το Διαδίκτυο των πραγμάτων (IoT) από το παραδοσιακό Διαδίκτυο; Αυτή η ενότητα συνοψίζει τις κύριες διαφορές μεταξύ των δύο.

Η πρώτη και πιο κατανοητή διαφορά μεταξύ του παραδοσιακού Διαδικτύου και του IoT είναι η ταυτότητα της δημιουργίας περιεχομένου (όπως σημειώνεται στον Πίνακα 2.1 παρακάτω). Ωστόσο, υπάρχουν πολλές άλλες αξιοσημείωτες διαφορές.

Για παράδειγμα, το περιεχόμενο στο παραδοσιακό Διαδίκτυο καταναλώνεται κατόπιν αιτήματος. Δηλαδή, κάποιος πρέπει να υποβάλει ερώτημα, να πραγματοποιήσει αναζήτηση ή να στείλει ένα αίτημα για μια υπηρεσία Ιστού για να καταναλώσει το περιεχόμενο. Αντίθετα, στο IoT, το περιεχόμενο καταναλώνεται συνήθως μέσω της προώθησης της τεχνολογίας ως ειδοποίηση ή την ενεργοποίηση μιας ενέργειας όταν εντοπίζεται μια κατάσταση ενδιαφέροντος. Σε πολλές περιπτώσεις, η κατανάλωση σημαίνει συνδυασμό δεδομένων από διαφορετικές πηγές. Αυτό ισχύει για το παραδοσιακό Διαδίκτυο καθώς και για το IoT.

Στο παραδοσιακό Διαδίκτυο, η σύνδεση γίνεται μέσω φυσικών συνδέσμων μεταξύ ιστοσελίδων. Στο IoT, ο συνδυασμός δεδομένων απαιτείται για την ανίχνευση καταστάσεων. Αυτό εκδηλώνεται με το συνδυασμό δεδομένων με τη μορφή προτύπων συμβάντων που

βασίζονται στο περιβάλλον, όπου ορισμένα από τα δεδομένα καθορίζουν το πλαίσιο και άλλα καθορίζουν το ίδιο το μοτίβο.

Θέμα	Παραδοσιακό Internet	The Internet of Things (IoT)
Ποιος δημιουργεί περιεχόμενο;	Άνθρωπος	Μηχανή
Πώς καταναλώνεται το περιεχόμενο;	Κατόπιν αιτήματος	Ωθώντας πληροφορίες και ενεργοποιώντας ενέργειες
Πώς συνδυάζεται το περιεχόμενο;	Χρησιμοποιώντας ρητά καθορισμένους συνδέσμους	Μέσω σαφώς καθορισμένων χειριστών
Ποια είναι η αξία;	Απάντηση ερωτήσεων	Δράση και έγκαιρη ενημέρωση
Τι γινόταν μέχρι τώρα;	Τόσο δημιουργία περιεχομένου (HTML) όσο και κατανάλωση περιεχομένου (μηχανές αναζήτησης)	Κυρίως δημιουργία περιεχομένου

Πίνακας 2.1 Σύγκριση μεταξύ του παραδοσιακού Διαδικτύου και του Διαδικτύου των πραγμάτων (IoT)

Υπάρχει επίσης μια διαφορά στην αξία για τον καταναλωτή. Στο παραδοσιακό Διαδίκτυο, η αξία έγκειται στην απάντηση μιας ερώτησης που τίθεται από τον καταναλωτή, σε πολλές περιπτώσεις κατά την αναζήτηση πληροφοριών ή την ενεργοποίηση υπηρεσιών. Στο IoT, η τιμή είναι έγκαιρη ενέργεια ή ειδοποίηση βάσει ανιχνευόμενων καταστάσεων.

Αυτό μας φέρνει στη σύγκριση της τελευταίας τεχνολογίας και στους δύο τομείς. Το παραδοσιακό Διαδίκτυο είναι μια ώριμη τεχνολογία. Έχει πρότυπα σε διάφορους τομείς και μηχανές αναζήτησης με τα οποία μπορεί κανείς να επικοινωνήσει χρησιμοποιώντας φυσικές γλώσσες. Το καθαρό αποτέλεσμα είναι ότι η κατανάλωση του παραδοσιακού Διαδικτύου μπορεί να γίνει από όλους χωρίς την ανάγκη τεχνικών δεξιοτήτων.

Αλλά στον τομέα IoT, η κατάσταση είναι πολύ διαφορετική. Η προσπάθεια τυποποίησης είναι μόνο στις πρώτες φάσεις της, επομένως [6], η ολοκλήρωση δεδομένων γίνεται ad hoc και απαιτεί εξειδικευμένους προγραμματιστές για την εφαρμογή μιας εφαρμογής.

2.2 Αλγόριθμοι Μηχανικής Μάθησης

Η ανίχνευση ανωμαλιών υπάρχει ως ένα σημαντικό θέμα για αρκετές δεκαετίες και οδήγησε σε πολλές εφαρμογές σε ένα ευρύ φάσμα τομέων, όπως ανοχή σφαλμάτων στη βιομηχανία, ανίχνευση κρίσεων στη χρηματοδότηση και οικονομία, διάγνωση υγείας, ακραία φαινόμενα στην επιστήμη της γης και μετεωρολογία, ανίχνευση άτυπων ουράνιων αντικειμένων στην αστρονομία ή την αστροφυσική, την εισβολή των συστημάτων στην ασφάλεια του κυβερνοχώρου κ.λπ. Η ανίχνευση ανωμαλιών ορίζεται γενικά ως το πρόβλημα του εντοπισμού προτύπων που αποκλίνουν από ένα μοντέλο συμπεριφοράς «κανονικότητας». Στη βιβλιογραφία, οι περισσότερες προσεγγίσεις μπορούν να κατηγοριοποιηθούν είτε σύμφωνα με το μοντέλο της κανονικότητας που εμπλέκεται είτε με τον τρόπο με τον οποίο αντιμετωπίζουν τον χαρακτηρισμό της ανωμαλίας και την ταυτοποίησή τους. Μια αρκετά εξαντλητική, αν και λίγο παλιά, μελέτη είχε προταθεί που αναθεωρούσε την ανίχνευση ανωμαλιών, η οποία ολοκληρώθηκε με μια πιο πρόσφατη συγκριτική μελέτη [7, 8]. Σύμφωνα με αυτές τις μελέτες, οι προηγμένες μέθοδοι μπορούν να καταταχθούν σε πέντε κύριες κατηγορίες:

(1) Κοντινοί γείτονες (near neighbors) και μέθοδοι που βασίζονται σε ομαδοποίηση (clustering-based): Οι μέθοδοι Near Neighbors βασίζονται στην υπόθεση ότι μια «φυσιολογική» εμφάνιση εμφανίζεται κοντά στους κοντινούς γείτονές της, ενώ μια ανωμαλία εμφανίζεται πολύ μακριά από τους γείτονές της. Παρομοίως, οι μέθοδοι που βασίζονται σε σύμπλεγμα (clusters) βασίζονται στην υπόθεση ότι μια «κανονική» εμφάνιση εμφανίζεται κοντά στο πλησιέστερο κεντροειδές του συμπλέγματος ενώ μια ανωμαλία θα συμβεί πολύ μακριά από το πλησιέστερο κεντροειδές του συμπλέγματος [9].

(2) Μέθοδος βάσει ταξινόμησης (classification-based): σε αυτό το παράδειγμα, πολλές κλάσεις «κανονικών» δεδομένων μαθαίνονται από ένα σύνολο έναντι όλων των ταξινομητών (κάθε ταξινομητής συνδέεται με μια κλάση και εκπαιδεύεται για να την διαχωρίσει από τις άλλες κλάσεις). Μια περίπτωση που δεν κατηγοριοποιείται ως «κανονική» από κανέναν από αυτούς τους ταξινομητές θεωρείται ανωμαλία. Μια περίεργη περίπτωση συμβαίνει όταν χρησιμοποιείται μία κλάση για τη μοντελοποίηση των «κανονικών» δεδομένων. Ο Random Forest, συμπεριλαμβανομένων των πρόσφατων εξελίξεων σε ένα Random Forest μιας κλάσης, ο Support Vector Machine (SVM) με ταξινομητές πολλαπλών τάξεων (multi-class) και με ταξινομητή μιας κατηγορίας (one-class) και τα νευρωνικά δίκτυα (Neural Networks), είναι οι πιο χρησιμοποιημένοι ταξινομητές για ανίχνευση ανωμαλιών [10].

(3) Οι στατιστικές μέθοδοι (statistical-based) βασίζονται στην υπόθεση ότι τα «κανονικά» δεδομένα σχετίζονται με καταστάσεις υψηλής πιθανότητας μιας υποκείμενης στοχαστικής διαδικασίας, ενώ οι ανωμαλίες σχετίζονται με καταστάσεις χαμηλής πιθανότητας αυτής της διαδικασίας. Δημοφιλείς προσεγγίσεις σε αυτήν την κατηγορία είναι τα μοντέλα πυκνότητας με βάση τον πυρήνα (kernel-based) και το Gaussian Mixture Model (GMM), συμπεριλαμβανομένης της πρόσφατης προόδου στο GMM μιας κλάσης [11].

(4) Οι μέθοδοι θεωρητικής πληροφόρησης (Information theoretic based) χρησιμοποιούν θεωρητικά μέτρα πληροφόρησης, όπως η εντροπία, η πολυπλοκότητα του Kolmogorov, το ελάχιστο μήκος περιγραφής κ.λπ., για την εκτίμηση της «πολυπλοκότητας» του «κανονικού» συνόλου δεδομένων (ή ισοδύναμα της πολυπλοκότητας της διαδικασίας πίσω από την παραγωγή αυτών των δεδομένων). Εάν το $C(D)$ εκτιμήσει την πολυπλοκότητα του συνόλου δεδομένων D , το ελάχιστο υποσύνολο της παρουσίας I που μεγιστοποιεί το $C(D) - C(I)$ θεωρείται ως το υποσύνολο ανωμαλίας [12].

(5) Η μέθοδος με βάση το φάσμα (spectral-based) βασίζεται στην υπόθεση ότι είναι δυνατή η ενσωμάτωση των δεδομένων σε μία υπο-περιοχή χαμηλότερης διάστασης στην οποία τα «κανονικά» δεδομένα και οι ανωμαλίες υποτίθεται ότι είναι καλά διαχωρισμένες. Η PCA και η ομαδοποίηση γραφήματος (ομοιότητας) είναι από τις πιο δημοφιλείς μεθόδους αυτής της κατηγορίας [13].

Μια πρόσφατη ανασκόπηση, αν και πολύ λιγότερο λεπτομερής, προσδιορίζει μια συμπληρωματική κατηγορία που βρίσκεται ανάμεσα σε επιβλεπόμενες και μη επιβλεπόμενες μεθόδους και την οποία οι συγγραφείς αναφέρονται ως «υβριδικές μέθοδοι» [14]. Τέτοιες προσεγγίσεις συνδυάζουν επιβλεπόμενες, ημι-επιβλεπόμενες και μη επιβλεπόμενες τεχνικές. Το 2008, το Isolation Forest (IF), μια εννοιολογικά διαφορετική προσέγγιση στις μεθόδους που αναφέρθηκαν προηγουμένως, προτάθηκε που παράξενα ξέφυγε από το ραντάρ της προηγούμενης επισκόπησης [2]. Το παράδειγμα IF βασίζεται στη δυσκολία απομόνωσης μιας συγκεκριμένης παρουσίας σε ολόκληρο το σύνολο παρουσιών κατά τη χρήση (τυχαίων) δομών δέντρων. Βασίζεται στην υπόθεση ότι μια ανωμαλία είναι γενικά πολύ πιο εύκολη στην απομόνωση από μια «κανονική» παρουσία δεδομένων. Ως εκ τούτου, το IF είναι μια μη εποπτευόμενη προσέγγιση που σχετίζεται κάπως με τις θεωρητικές μεθόδους της πληροφορίας, δεδομένου ότι η δυσκολία απομόνωσης αντιμετωπίζεται μέσω ενός αλγοριθμικού σχήματος πολυπλοκότητας. Το IF έχει χρησιμοποιηθεί με επιτυχία σε ορισμένες εφαρμογές και επεκτάθηκε πρόσφατα για να βελτιώσει την επιλογή των χαρακτηριστικών και των τιμών διαχωρισμού τους κατά την κατασκευή του δέντρου [15]. Το κύριο πλεονέκτημα των αλγορίθμων που βασίζονται σε IF είναι η ικανότητά τους να

επεξεργάζονται μεγάλη ποσότητα δεδομένων σε χώρους υψηλής διάστασης με υπολογιστική και χωρική ελεγχόμενη απόδοση σε σύγκριση με άλλες μη εποπτευόμενες μεθόδους. Στην επόμενη ενότητα ο αλγόριθμος Isolation Forest θα συζητηθεί με μεγαλύτερη λεπτομέρεια.

2.3 Ο αλγόριθμος Isolation Forest

Ο αλγόριθμος IForest προτάθηκε από τους Liu, Fei et al. το 2012; Αυτός ο αλγόριθμος είναι ένα δένδροειδές (tree-based) μοντέλο ανίχνευσης των περιπτώσεων άτυπης συμπεριφοράς (outlier detection model) με γραμμική πολυπλοκότητα χρόνου και υψηλή ακρίβεια και κατάλληλο για υψηλής διαστάσεων και μεγάλη ποσότητα δεδομένων.

Επειδή οι ανωμαλίες είναι «λιγότερες και διαφορετικές», είναι πιο εύαλωτες στην απομόνωση. Σε ένα τυχαίο δέντρο προσανατολισμένο στα δεδομένα, οι εγγραφές κόβονται αναδρομικά μέχρι να απομονωθούν όλες οι εγγραφές. Αυτό το τυχαίο διαμέρισμα κάνει την εγγραφή outlier να έχει μικρότερο μήκος διαδρομής, επειδή οι εγγραφές με διακριτές τιμές χαρακτηριστικών είναι πιθανότερο να διαχωριστούν σε πρώιμα διαμερίσματα. Το IForest αποτελείται από μερικά iTrees (δέντρο απομόνωσης). Κάθε iTree είναι ένα δυαδικό δέντρο. Τα βήματα υλοποίησης έχουν ως εξής:

- (1) Επιλέξτε τυχαία έναν σταθερό αριθμό σημείων δειγμάτων από δεδομένα εκπαίδευσης ως υποδείγματα και τοποθετήστε τα στον ριζικό κόμβο (root node) του δέντρου.
- (2) Προσδιορίστε τυχαία ένα χαρακτηριστικό και δημιουργήστε τυχαία ένα σημείο κοπής p στα τρέχοντα δεδομένα του κόμβου, το σημείο κοπής δημιουργείται μεταξύ της μέγιστης και της ελάχιστης τιμής του καθορισμένου χαρακτηριστικού στα τρέχοντα δεδομένα του κόμβου.
- (3) Ένα υπερπλάνο (hyperplane) δημιουργείται από αυτό το σημείο κοπής και ο χώρος δεδομένων του τρέχοντος κόμβου χωρίζεται σε δύο δευτερεύουσες θέσεις: τα δεδομένα μικρότερα από p στο καθορισμένο χαρακτηριστικό τοποθετούνται στο αριστερό θυγατρικό (child) του τρέχοντος κόμβου και τα δεδομένα μεγαλύτερα ή ίσα με p τοποθετούνται στο δεξιό θυγατρικό (child) του τρέχοντος κόμβου.
- (4) Εκτελέστε αναδρομικά τα βήματα 2 και 3, έως ότου ο θυγατρικός (child) κόμβος έχει μόνο μία εγγραφή ή το iTree φτάσει στο καθορισμένο ύψος.

Αφού λάβετε αυτά τα iTrees, η εκπαίδευση του iForest τερματίζεται και, στη συνέχεια, μπορούμε να αξιολογήσουμε τα δεδομένα δοκιμών (testing data) χρησιμοποιώντας το iForest που δημιουργήθηκε. Για μία δοκιμαστική εγγραφή (testing record), αφήστε το να διασχίσει κάθε iTree και, στη συνέχεια, υπολογίστε το ύψος των εγγραφών που τελικά πέφτουν σε κάθε δέντρο. Τότε μπορούμε να πάρουμε το μέσο ύψος της εγγραφής σε κάθε δέντρο. Εάν το μέσο ύψος είναι μικρότερο από το δεδομένο όριο, τότε η εγγραφή θεωρείται outlier.

Κεφάλαιο 3

Δομή προγράμματος

3.1 Δομή Καταγραφής κυκλοφορίας δικτύου	11
3.2 Δομή Προετοιμασίας Δεδομένων	13
3.3 Δομή Διαδικασίας Εκπαίδευσης	14
3.4 Δομή Ανίχνευσης επιθέσεων	15
3.5 Δημιουργία Εργαλείου	16
3.6 Πολυπλοκότητα	17

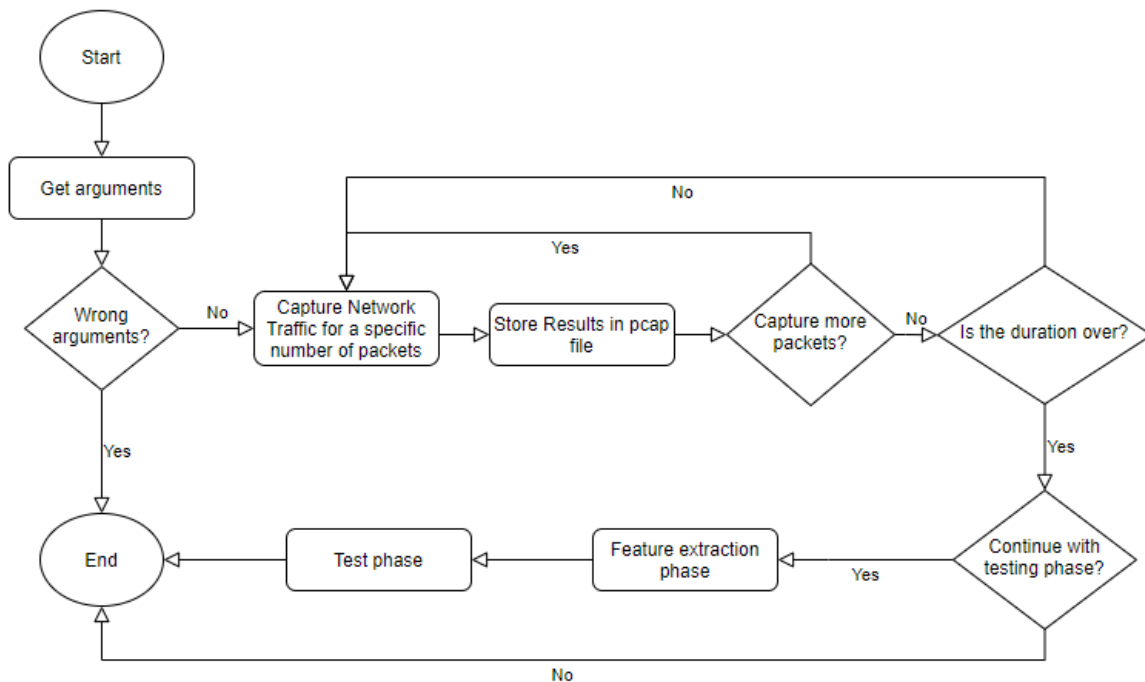
3.1 Δομή Καταγραφής κυκλοφορίας δικτύου

Αρχικά το πρόγραμμα παίρνει τις παραμέτρους, υπάρχουν 6 προαιρετικές παραμέτροι που είναι οι εξής:

- (1) `--i <true/false>`: Καθορίζει αν το πρόγραμμα θα λειτουργεί επ' άπειρον (infinitely). Η προκαθορισμένη τιμή είναι false.
- (2) `--t <number>`: Καθορίζει τον αριθμό των λεπτών που θα λειτουργεί το πρόγραμμα. Η προκαθορισμένη τιμή είναι καμία αφού απλά θα καταγράψει για μία μόνο φορά την κίνηση δικτύου όπως καθορίστηκε από τον αριθμό των πακέτων που της παραμέτρου `--p`.
- (3) `--s <true/false>`: Καθορίζει αν το πρόγραμμα θα αποθηκεύσει σε μόνο ένα αρχείο pcap τη καταγραμμένη κίνηση δικτύου. Η προκαθορισμένη τιμή είναι false.
- (4) `--p <number>`: Καθορίζει τον αριθμό των πακέτων που θα καταγραφτούν. Η προκαθορισμένη τιμή είναι 200.

(5) --c <true/false>: Καθορίζει αν το πρόγραμμα θα συνεχίσει με την χρήση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων, Τροφοδοσία των δεδομένων στον αλγόριθμο και Ανίχνευση των επιθέσεων). Η προκαθορισμένη τιμή είναι false.

(6) --d <true/false>: Καθορίζει αν το πρόγραμμα θα διαγράψει τα αρχεία που περιέχουν τα εξαγόμενα χαρακτηριστικά (extracted features) μετά την ολοκλήρωση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων, Τροφοδοσία των δεδομένων στον αλγόριθμο και Ανίχνευση των επιθέσεων). Η προκαθορισμένη τιμή είναι true. Σημείωση ότι για να έχει οποιοδήποτε αποτέλεσμα η συγκεκριμένη παράμετρος πρέπει να πρώτα να υπάρχει σαν παράμετρος η --c αφού για να υπάρξουν τα αρχεία με τα εξαγόμενα χαρακτηριστικά πρέπει να συνεχιστεί η λειτουργία του προγράμματος όπου θα συνεχίσει με την χρήση των υπόλοιπων λειτουργιών.



Εικόνα 3.1 Σχεδιάγραμμα δομής καταγραφής κυκλοφορίας δικτύου

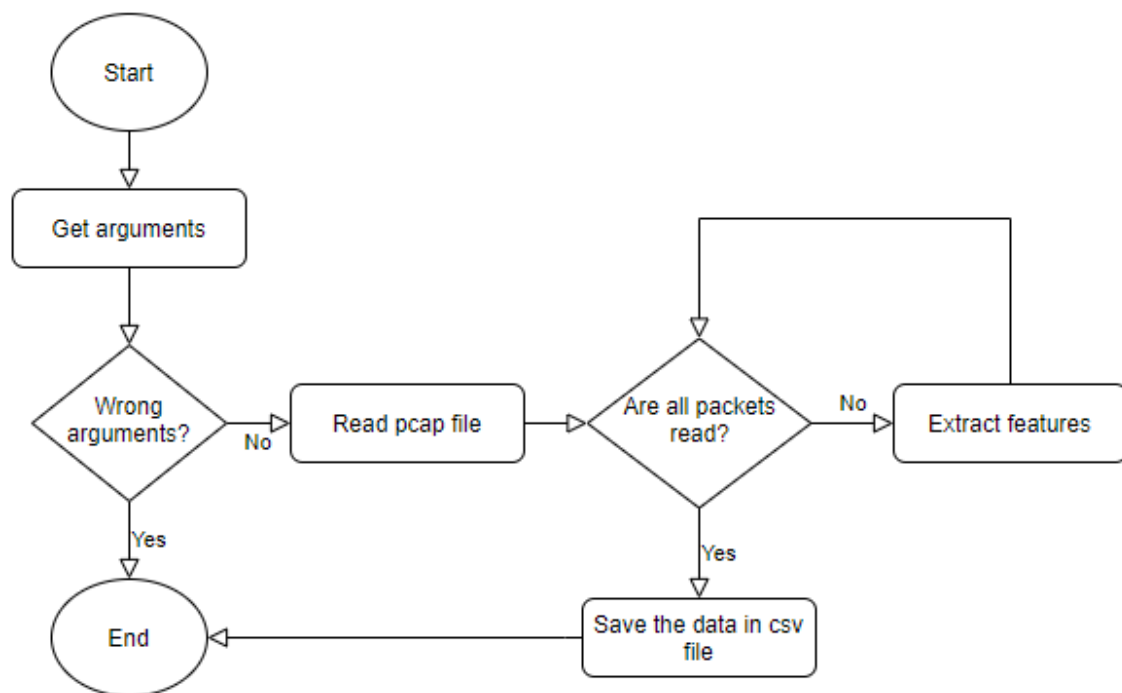
Χρησιμοποιώντας το εργαλείο sniff της βιβλιοθήκης scapy [16] το πρόγραμμα ξεκινάει την διαδικασία sniff όπου καταγράφει όλη την κυκλοφορία προς την προκαθορισμένη σύνδεση (connection) πού είναι και αυτή που όντως γίνεται η επικοινωνία από και προς την συσκευή. Όταν ολοκληρωθεί η διαδικασία αυτή για τον αριθμό των πακέτων που καθορίστηκε τότε δημιουργείται ένα νέο αρχείο pcap όπου και αποθηκεύεται η καταγεγραμμένη πληροφορία.

Αν επιλέχτηκε η συνέχεια του προγράμματος όπου θα προχωρήσει στις υπόλοιπες λειτουργίες τότε καλούνται τα απαραίτητα αρχεία για να επιτευχθεί αυτό.

3.2 Δομή Προετοιμασίας Δεδομένων

Αρχικά το πρόγραμμα παίρνει τις παραμέτρους, υπάρχει 1 υποχρεωτική παράμετρος που είναι η εξής:

(1) `--f <filename.pcap>`: Καθορίζει το αρχείο εισόδου pcap που θα χρησιμοποιηθεί. Χωρίς αυτήν την παράμετρο δεν αρχίζει η λειτουργία του προγράμματος.



Εικόνα 3.2 Σχεδιάγραμμα δομής προετοιμασίας δεδομένων

Χρησιμοποιώντας τη βιβλιοθήκη `scapy` [16] το πρόγραμμα ξεκινάει την διαδικασία διαβάσματος του αρχείου pcap όπου παίρνει ένα προς ένα τα πακέτα και βλέπει το τι περιέχει το κάθε πακέτο και αποθηκεύει 32 χαρακτηριστικά (features) για το κάθε ένα από αυτά.

Packet	MAC-Src1	MAC-Src2	MAC-Src3	MAC-Src4	MAC-Src5
MAC-Src6	MAC-Dst1	MAC-Dst2	MAC-Dst3	MAC-Dst4	MAC-Dst5
MAC-Dst5	LLC	IPv	IP-Src1	IP-Src2	IP-Src3
IP-Src4	IP-Dst1	IP-Dst2	IP-Dst3	IP-Dst4	IP-Tos
IP-Len	Protocol	Port-Src	Port-Dst	wirelen	caplen
sec	usec				

Πίνακας 3.1 Εξαγόμενα χαρακτηριστικά

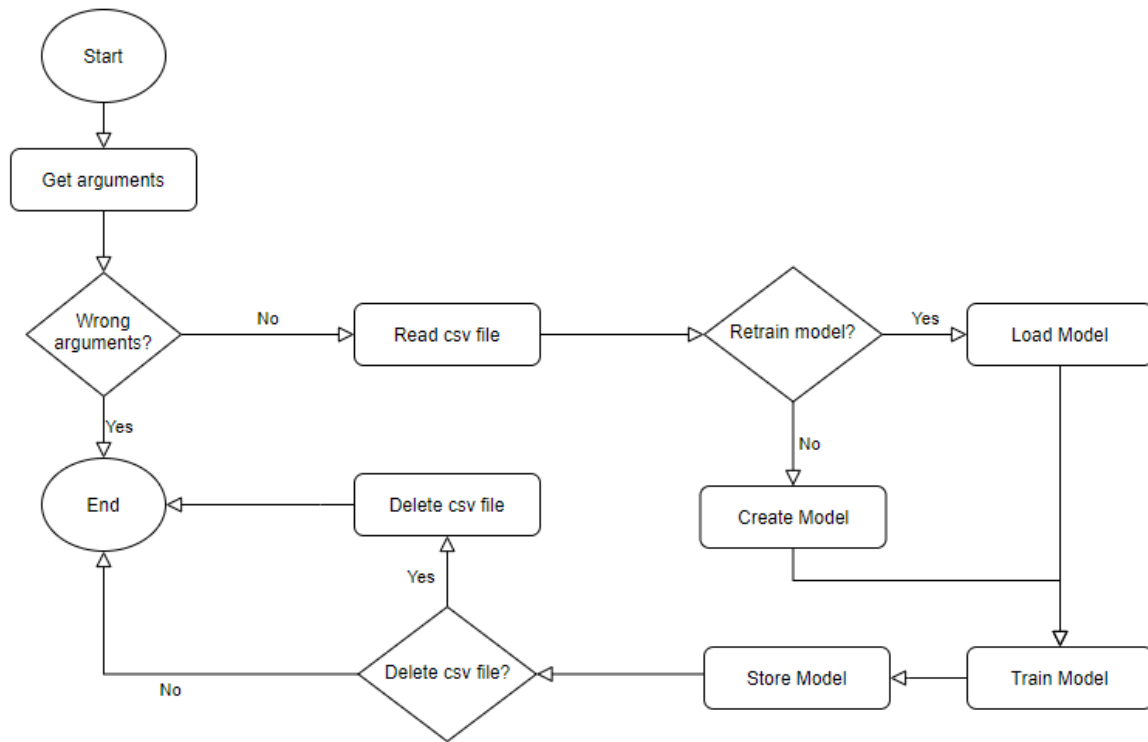
Μετά την ολοκλήρωση της εξαγωγής χαρακτηριστικών για όλα τα πακέτα που βρίσκονταν στο αρχείο pcap τα χαρακτηριστικά αποθηκεύονται σε ένα αρχείο csv για να χρησιμοποιηθούν αργότερα με την χρήση της βιβλιοθήκης pandas [17].

3.3 Δομή Διαδικασίας Εκπαίδευσης

Αρχικά το πρόγραμμα παίρνει τις παραμέτρους, υπάρχει 1 προαιρετική παράμετρος και 1 υποχρεωτική που είναι οι εξής:

(1) --r <true/false>: Καθορίζει αν ο αλγόριθμος θα προσθέσει τα νέα δεδομένα σε ένα ήδη εκπαιδευμένο μοντέλο του Isolation Forest (true) ή αν θα δημιουργήσει ένα νέο μοντέλο του Isolation Forest (false). Η προκαθορισμένη τιμή είναι false.

(2) --f <filename.csv>: Καθορίζει το αρχείο εισόδου csv που θα χρησιμοποιηθεί. Χωρίς αυτήν την παράμετρο δεν αρχίζει η λειτουργία του προγράμματος.



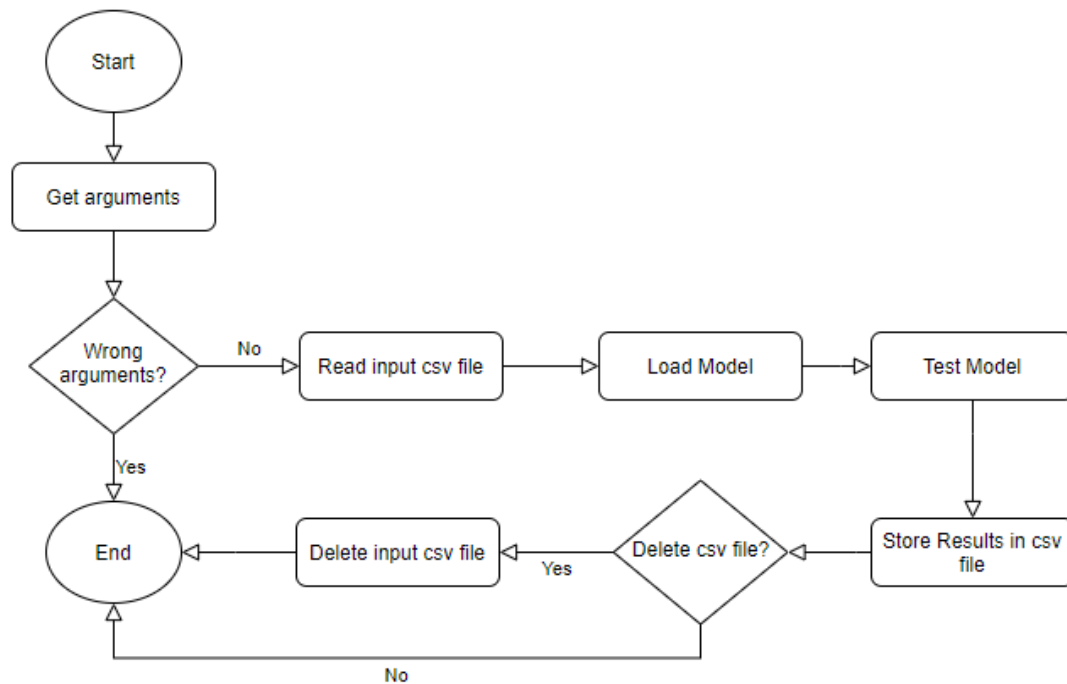
Εικόνα 3.3 Σχεδιάγραμμα δομής διαδικασίας εκπαίδευσης

Χρησιμοποιώντας τη βιβλιοθήκη pandas [17] το πρόγραμμα ξεκινάει την διαδικασία διαβάσματος του αρχείου csv όπου παίρνει τα δεδομένα και τα τροφοδοτεί στον αλγόριθμο Isolation Forest της βιβλιοθήκης sklearn [18] όπου και γίνεται η εκπαίδευση του μοντέλου. Όταν ολοκληρωθεί η διαδικασία αυτή τα μοντέλα αποθηκεύεται στον σύστημα με την χρήση της βιβλιοθήκης joblib [19].

3.4 Δομή Ανίχνευσης επιθέσεων

Αρχικά το πρόγραμμα παίρνει τις παραμέτρους, υπάρχει 1 υποχρεωτική παράμετρος που είναι η εξής:

- (1) `--f <filename.csv>`: Καθορίζει το αρχείο εισόδου csv που θα χρησιμοποιηθεί. Χωρίς αυτήν την παράμετρο δεν αρχίζει η λειτουργία του προγράμματος.



Εικόνα 3.4 Σχεδιάγραμμα δομής ανίχνευσης επιθέσεων

Χρησιμοποιώντας τη βιβλιοθήκη pandas [17] το πρόγραμμα ξεκινάει την διαδικασία διαβάσματος του αρχείου csv όπου παίρνει τα δεδομένα τα οποία θα χρησιμοποιηθούν αργότερα. Επίσης φορτώνει το μοντέλο (το οποίο είναι ήδη εκπαιδευμένο) του αλγορίθμου Isolation Forest που είναι αποθηκευμένο στο σύστημα με την χρήση της βιβλιοθήκης joblib [19]. Αφού φορτώσει και τα δύο αυτά απαραίτητα συστατικά, τροφοδοτεί τα δεδομένα στον αλγόριθμο Isolation Forest της βιβλιοθήκης sklearn [18] όπου και γίνεται η αναγνώριση ανωμαλιών. Για κάθε ένα από τα πακέτα υπολογίζει το anomaly score και για όσα πακέτα το σκορ είναι κάτω από το όριο (τον μέσο όρο) τότε τα κατηγοριοποιεί ως κακόβουλα (malicious) ενώ τα υπόλοιπα τα κατηγοριοποιεί ως άκακα (benign). Τα αποτελέσματα αυτά αποθηκεύονται σε ένα αρχείο csv, με την χρήση της βιβλιοθήκης pandas [17], όπου περιέχει τον αριθμό του πακέτου και το αποτέλεσμα αυτής της κατηγοριοποίησης για το κάθε πακέτο.

3.5 Δημιουργία Εργαλείου

Χρησιμοποιώντας τη βιβλιοθήκη PyQt5 [20] όλες οι παραπάνω λειτουργίες ενώνονται με ένα γραφικό περιβάλλον χρήστη, Graphical User Interface (GUI). Με την χρήση αυτού του γραφικού περιβάλλοντος χρήστη απλοποιείται ο τρόπος λειτουργίας του εν λόγω εργαλείου. Όλες οι λειτουργίες που αναφέρθηκαν είναι δημιουργημένες με τη χρήση της python 3. Ο

κώδικας υλοποίησης του εργαλείου αυτού, καθώς και όλων των επιμέρους λειτουργιών (είναι ενσωματωμένες στο εργαλείο), βρίσκεται στο Παράρτημα Α.

3.6 Πολυπλοκότητα

Η χρονική πολυπλοκότητα τόσο του αλγορίθμου Isolation Forest όσο και τον υπόλοιπων προγραμμάτων και λειτουργιών που τον πλαισιώνουν είναι γραμμική (linear time complexity). Αυτό επιτυγχάνετε για το λόγο ότι δεν υπάρχει οποιαδήποτε αναδρομή ή επανάληψη στα ίδια δεδομένα ή και πακέτα που καλείται ο αλγόριθμος να επεξεργαστεί. Όλες οι διαδικασίες τρέχουν μόνο μια φορά για το κάθε δεδομένο και πακέτο. Έτσι η Βέλτιστη, η Χείριστη και η Μέση περίπτωση έχουν ακριβώς την ίδια χρονική πολυπλοκότητα. Όσο για την σταθερά της χρονικής πολυπλοκότητας, είναι σχετικά αρκετά μικρή αφού τα βήματα που χρειάζονται είναι πολύ μικρά και συγκεκριμένα.

Αυτό δεν επιτυγχάνεται μόνο στην χρονική αλλά και στην χωρική πολυπλοκότητα. Όπως εξηγήθηκε και πιο πάνω όλες οι διαδικασίες τρέχουν μόνο μια φορά για το κάθε δεδομένο και πακέτο, κάνοντάς τον αλγόριθμο πολύ γρήγορο και αποτελεσματικό. Για το κάθε πακέτο χρειάζεται να το χαρακτηρίσει με μόνο 32 μοναδικά χαρακτηριστικά, τα οποία είναι τύπου Integer και παίρνουν 8 bytes το κάθε ένα. Αυτό σημαίνει πως συνολικά το κάθε πακέτο χρειάζεται 256 bytes για να αποθηκευτεί για την διαδικασία της προετοιμασίας των δεδομένων (feature extraction).

Κεφάλαιο 4

Υλοποίηση Συστήματος

4.1 Καταγραφή κυκλοφορίας δικτύου	18
4.2 Προετοιμασία Δεδομένων	25
4.3 Διαδικασία Εκπαίδευσης	25
4.4 Ανίχνευση επιθέσεων	31

4.1 Καταγραφή κυκλοφορίας δικτύου

Σε αυτό το σημείο παρουσιάζεται το γραφικό περιβάλλον χρήστη μέσω του οποίου έχει την δυνατότητα να καταγράψει την κυκλοφορία του δικτύου καθώς και διάφορες άλλες επιλογές που έχουν επεξηγηθεί στο Κεφάλαιο 3.1.

Στην Εικόνα 4.1 παρουσιάζεται το προεπιλεγμένο γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου.

Στην Εικόνα 4.2 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού έτρεξε με τις προεπιλεγμένες επιλογές.

Στην Εικόνα 4.3 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Run Infinitely, όπου το πρόγραμμα θα λειτουργεί επ' άπειρον (infinitely).

Στην Εικόνα 4.4 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Duration of Packet Sniffer (in minutes), όπου καθόρισε τον αριθμό των λεπτών που θα λειτουργεί.

Στην Εικόνα 4.5 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Single pcap file for captured traffic, όπου το πρόγραμμα θα αποθηκεύσει σε μόνο ένα αρχείο pcap τη καταγραφόμενη κίνηση δικτύου. Σημειώνεται ότι για να γίνει αυτή η επιλογή τροποποιήσιμη πρέπει πρώτα να επιλεγεί η επιλογή Duration of Packet Sniffer (in minutes), όπου καθόρισε τον αριθμό των λεπτών που θα λειτουργεί.

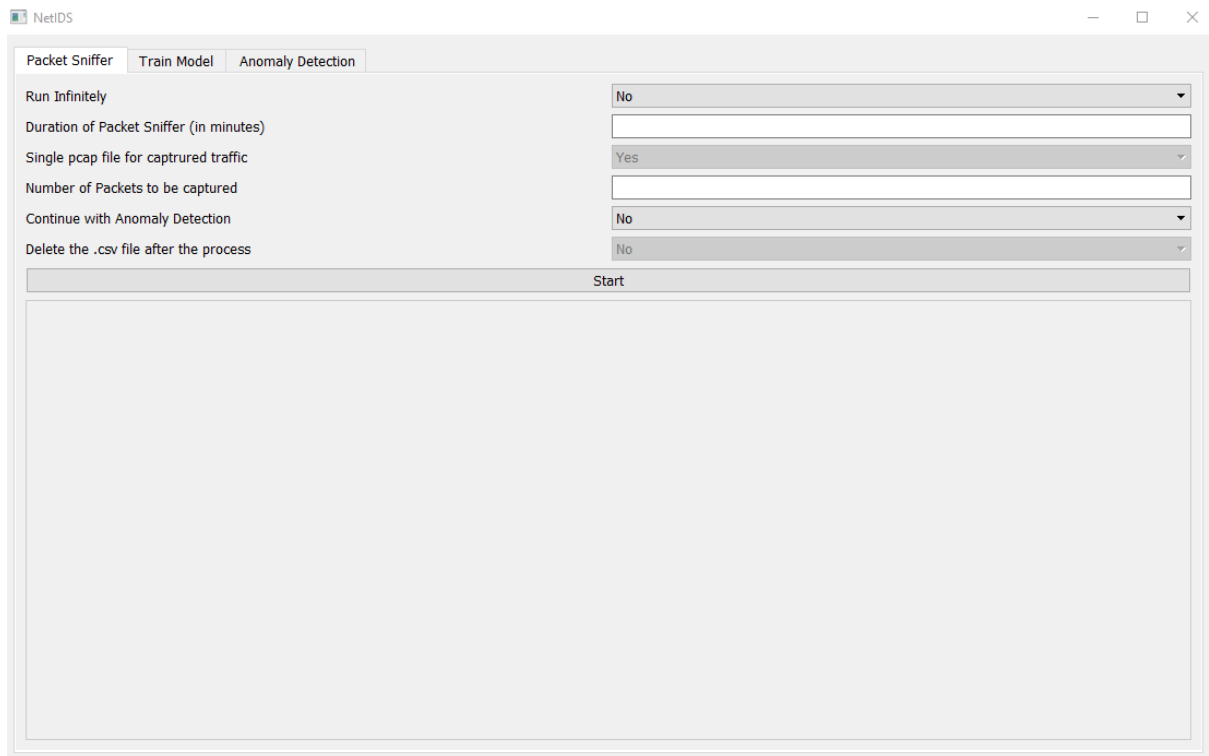
Στην Εικόνα 4.6 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Number of Packets to be captured, όπου καθόρισε τον αριθμό των πακέτων που θα καταγραφούν.

Στην Εικόνα 4.7 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Continue with Anomaly Detection, όπου καθόρισε πως το πρόγραμμα θα συνεχίσει με την χρήση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων, Τροφοδοσία των δεδομένων στον αλγόριθμο και Ανίχνευση των επιθέσεων).

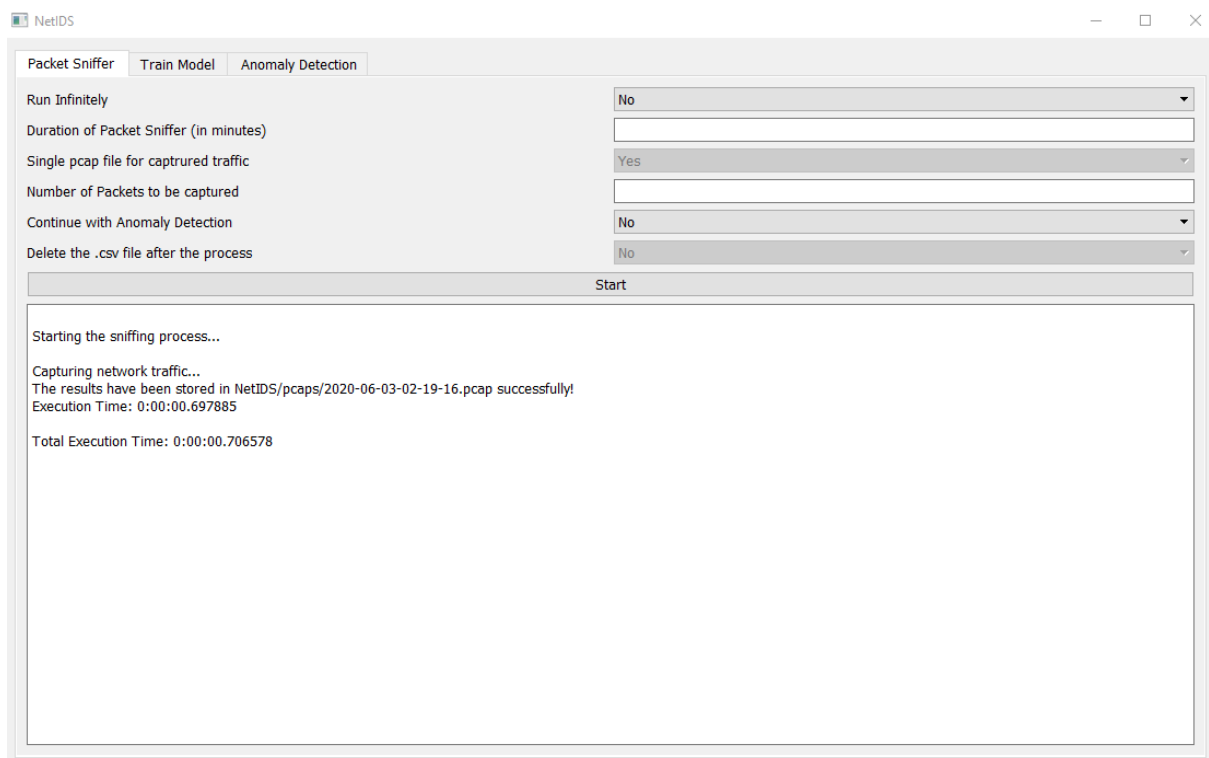
Στην Εικόνα 4.8 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Delete the .csv file after the process, όπου το πρόγραμμα θα διαγράψει τα αρχεία που περιέχουν τα εξαγόμενα χαρακτηριστικά (extracted features) μετά την ολοκλήρωση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων, Τροφοδοσία των δεδομένων στον αλγόριθμο και Ανίχνευση των επιθέσεων). Σημειώνεται ότι για να γίνει αυτή η επιλογή τροποποιήσιμη πρέπει πρώτα να επιλεγεί η επιλογή Continue with Anomaly Detection, όπου καθόρισε πως το πρόγραμμα θα συνεχίσει με την χρήση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων, Τροφοδοσία των δεδομένων στον αλγόριθμο και Ανίχνευση των επιθέσεων).

Στην Εικόνα 4.9 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Duration of Packet Sniffer (in minutes), όπου ο χρήστης είχε εισαγάγει λανθασμένη τιμή και το πρόγραμμα του εμφάνισε ένα μήνυμα σφάλματος.

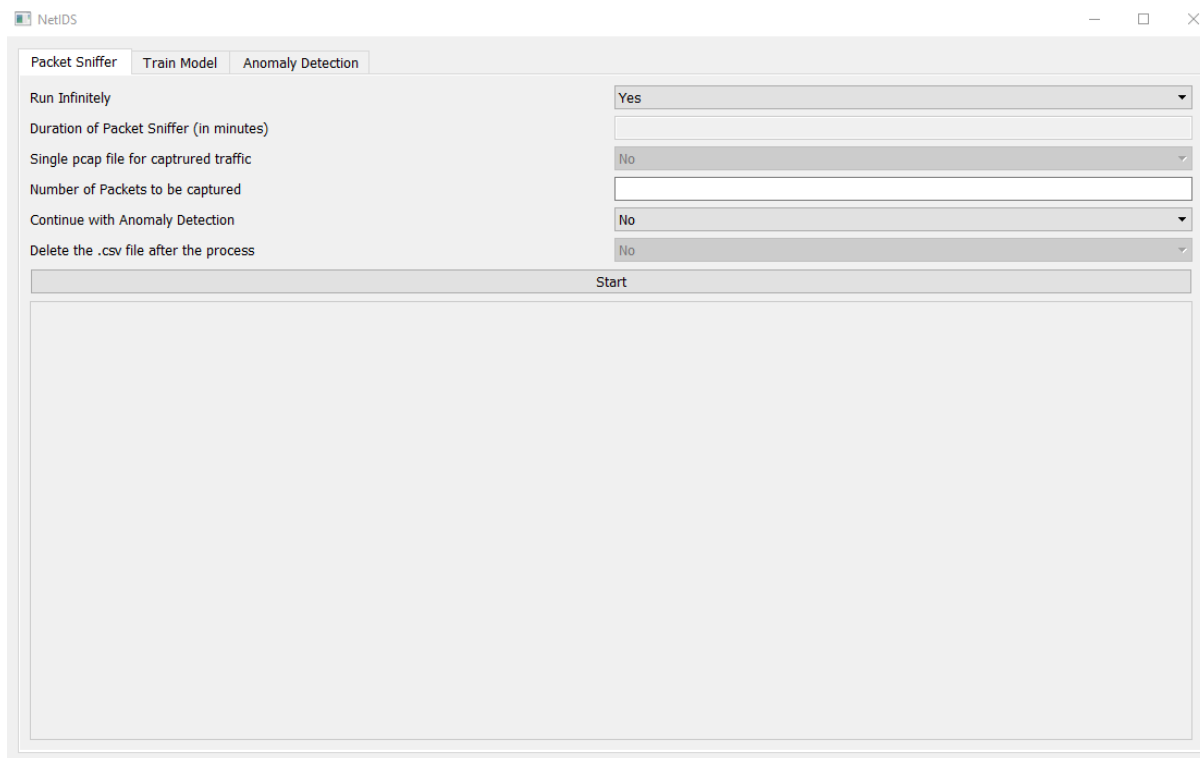
Στην Εικόνα 4.10 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της καταγραφής κυκλοφορίας δικτύου αφού επιλέχτηκε η επιλογή Number of Packets to be captured, όπου ο χρήστης είχε εισαγάγει λανθασμένη τιμή και το πρόγραμμα του εμφάνισε ένα μήνυμα σφάλματος.



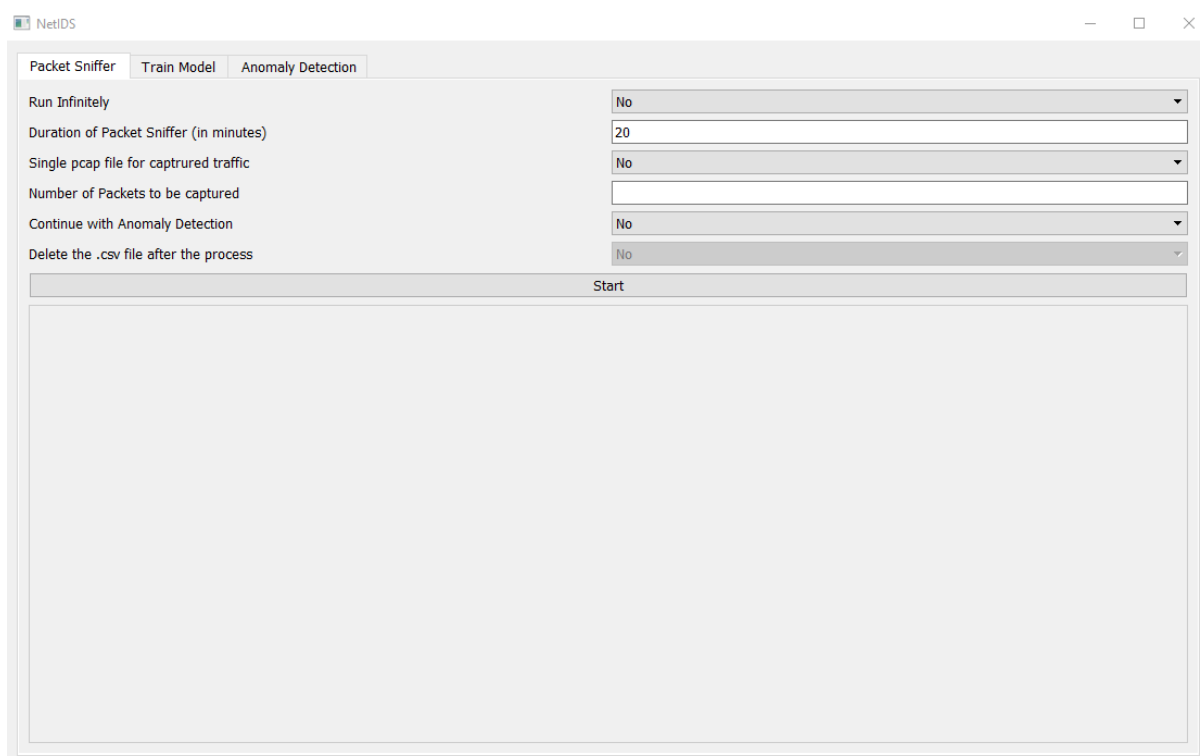
Εικόνα 4.1 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



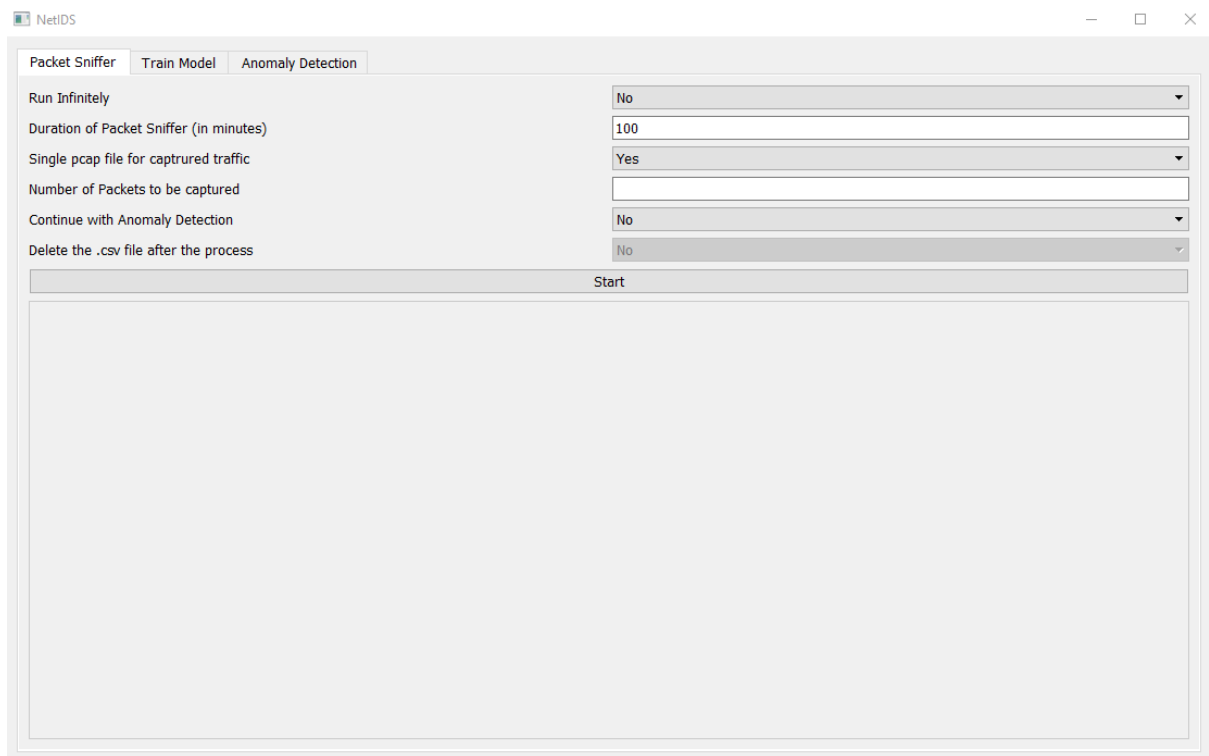
Εικόνα 4.2 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



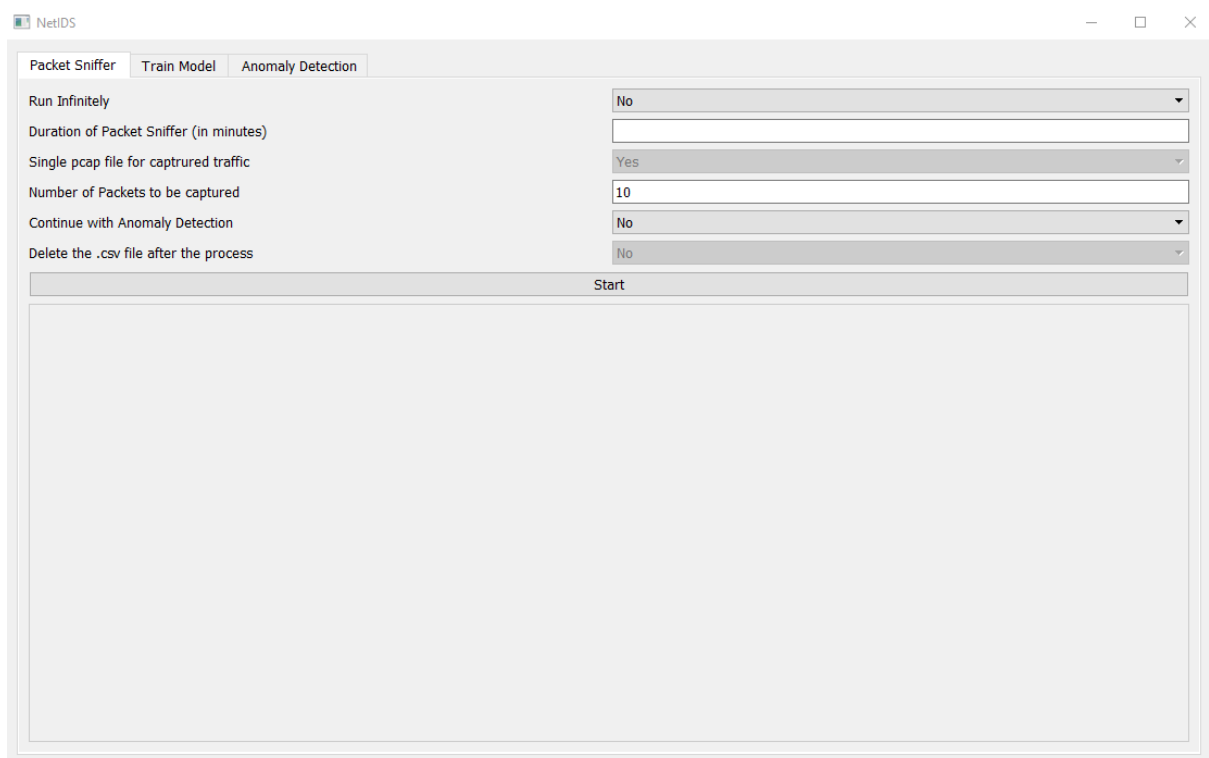
Εικόνα 4.3 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



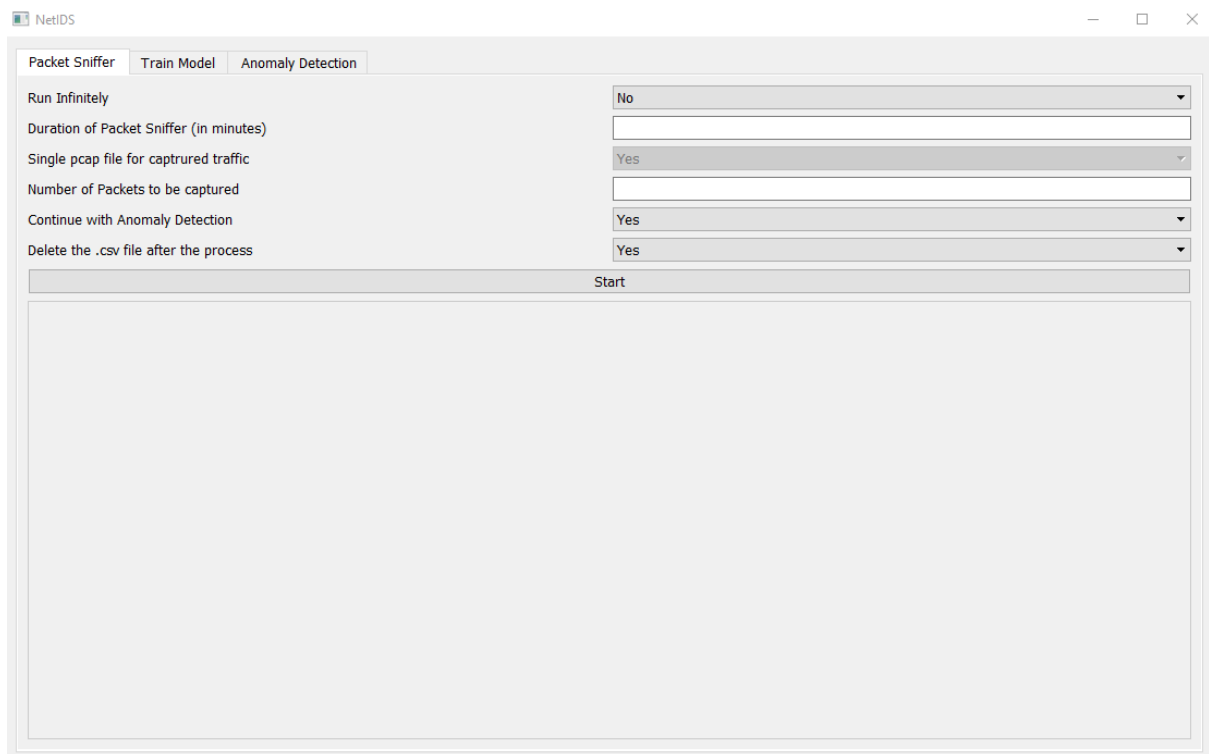
Εικόνα 4.4 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



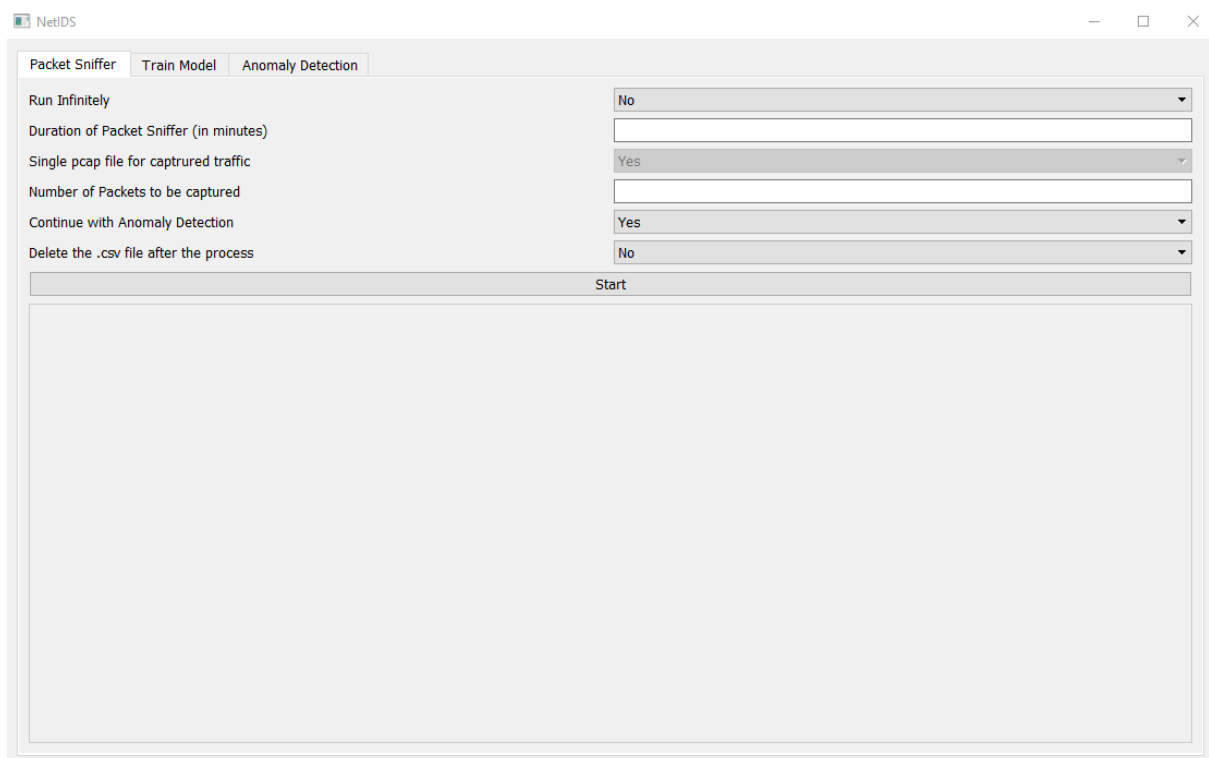
Εικόνα 4.5 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



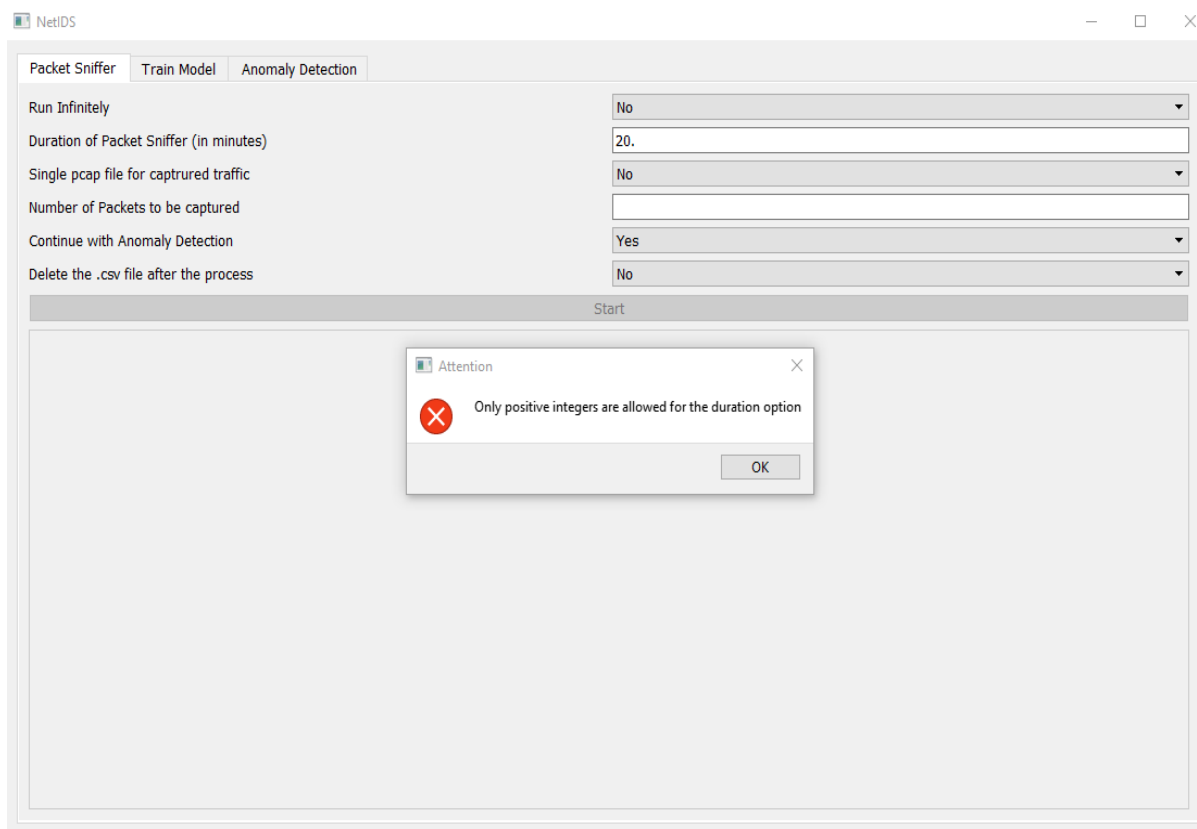
Εικόνα 4.6 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



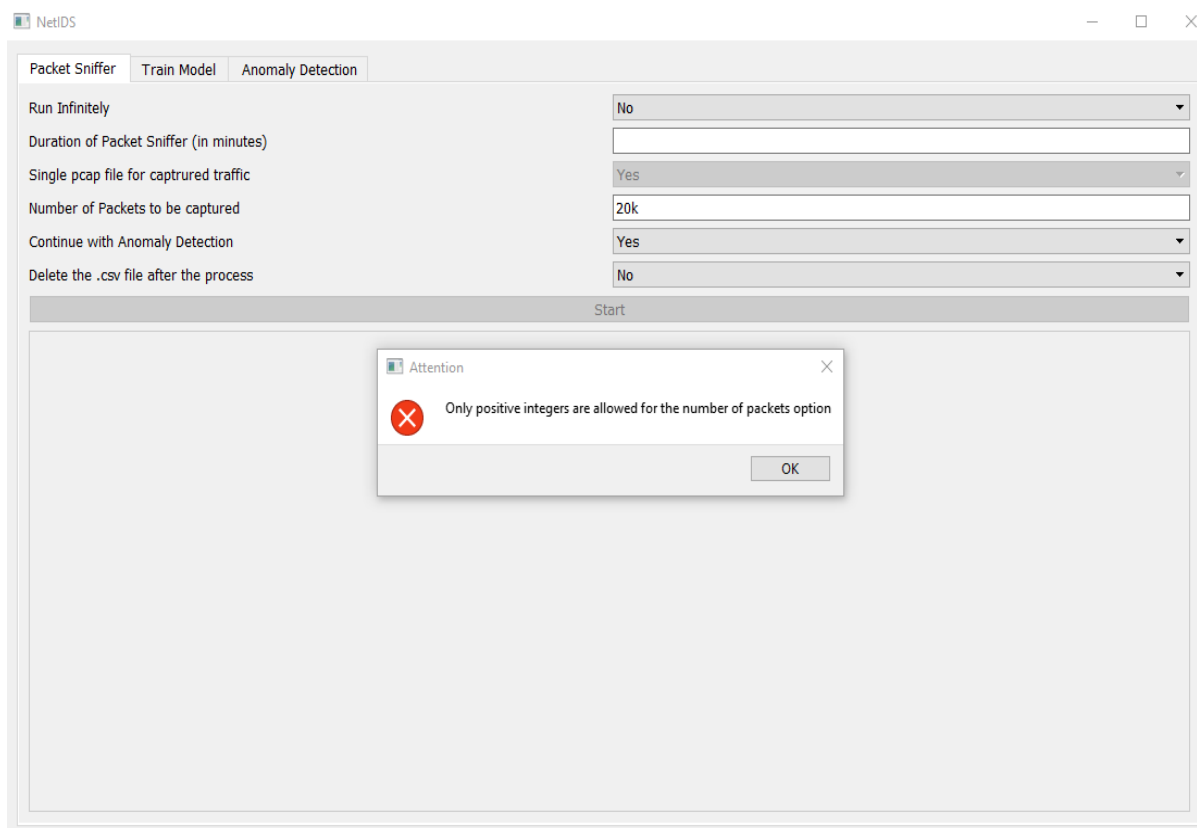
Εικόνα 4.7 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



Εικόνα 4.8 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



Εικόνα 4.9 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου



Εικόνα 4.10 Γραφικό περιβάλλον καταγραφής κυκλοφορίας δικτύου

4.2 Προετοιμασία Δεδομένων

Η διαδικασία προετοιμασίας των δεδομένων δεν προσφέρεται σαν αυτόνομη λειτουργία αφού ο απώτερος σκοπός είναι η ανίχνευσή των επιθέσεων σε ένα δίκτυο. Η λειτουργία αυτή είναι όμως άκρως απαραίτητη για όλες τις υπόλοιπες λειτουργίες και εκτελείται πάντοτε σε αρχεία τύπου pcap, είτε προορίζονται για εκπαίδευση του αλγορίθμου είτε για ανίχνευσή των επιθέσεων.

4.3 Διαδικασία Εκπαίδευσης

Σε αυτό το σημείο παρουσιάζεται το γραφικό περιβάλλον χρήστη μέσω του οποίου έχει την δυνατότητα να εκπαιδεύσει το μοντέλο του αλγορίθμου Isolation Forest καθώς και διάφορες άλλες επιλογές που έχουν επεξηγηθεί στο Κεφάλαιο 3.3.

Στην Εικόνα 4.11 παρουσιάζεται το προεπιλεγμένο γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest.

Στην Εικόνα 4.12 παρουσιάζεται το παράθυρο που εμφανίζεται για την επιλογή αρχείου εισόδου αφού έχει πατηθεί το κουμπί Browse File.

Στην Εικόνα 4.13 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού έγινε η επιλογή αρχείου εισόδου.

Στην Εικόνα 4.14 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού έτρεξε με τις προεπιλεγμένες επιλογές.

Στην Εικόνα 4.15 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού επιλέχτηκε η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

Στην Εικόνα 4.16 παρουσιάζεται το παράθυρο που εμφανίζεται για την επιλογή αρχείου εισόδου αφού έχει πατηθεί το κουμπί Browse File. Σημειώνεται ότι έχει ήδη επιλεχτεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

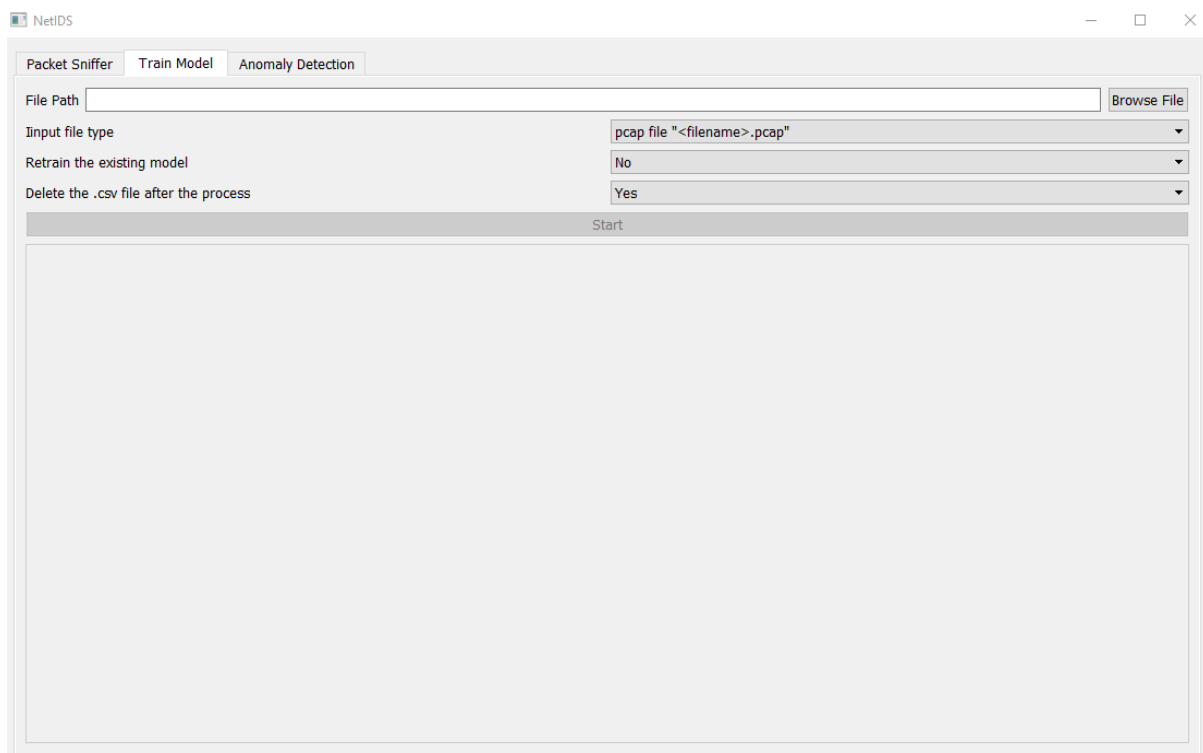
Στην Εικόνα 4.17 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού έγινε η επιλογή αρχείου εισόδου.

Σημειώνεται ότι έχει ήδη επιλεχτεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

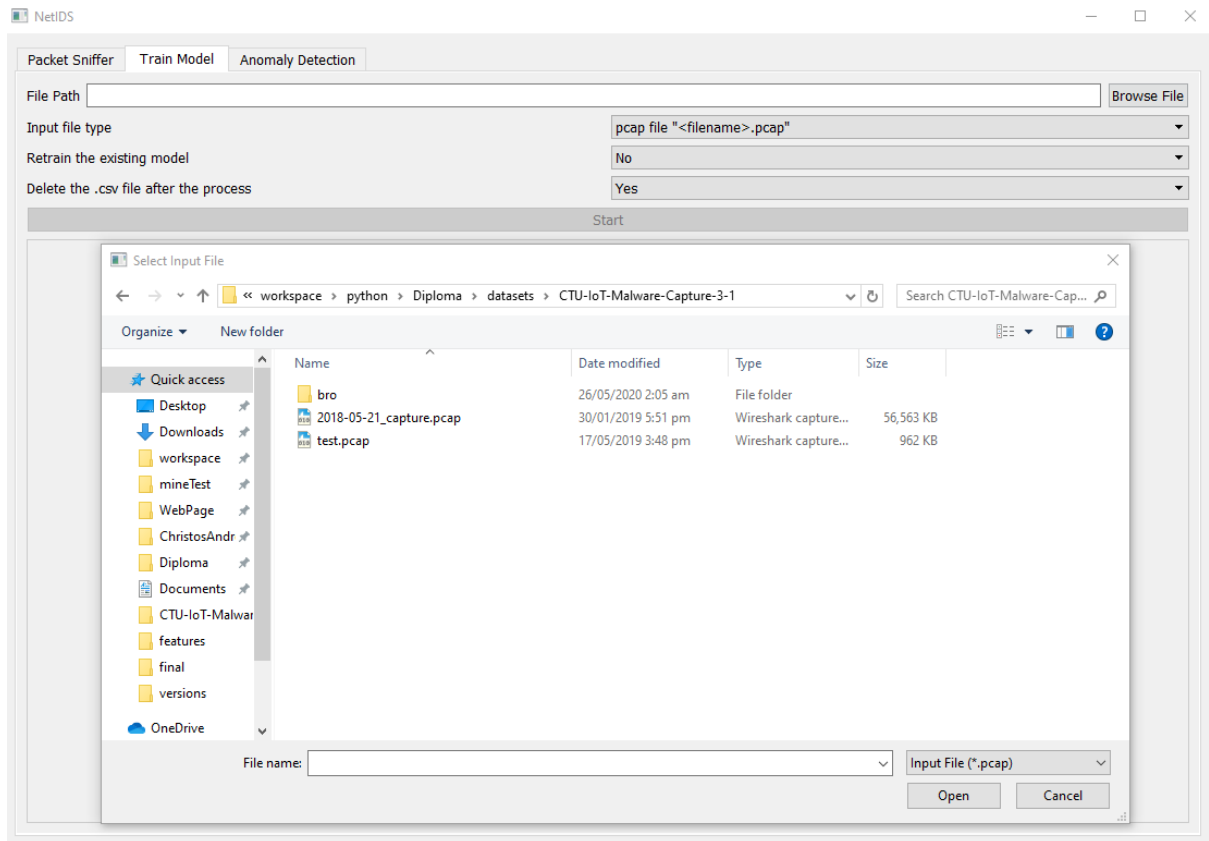
Στην Εικόνα 4.18 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού έτρεξε με τις προεπιλεγμένες επιλογές. Σημειώνεται ότι έχει ήδη επιλεχτεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

Στην Εικόνα 4.19 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού επιλέχτηκε η επιλογή Retrain the existing model, όπου καθόρισε το αν ο αλγόριθμος θα προσθέσει τα νέα δεδομένα σε ένα ήδη εκπαιδευμένο μοντέλο του Isolation Forest (Yes) ή αν θα δημιουργήσει ένα νέο μοντέλο του Isolation Forest (No).

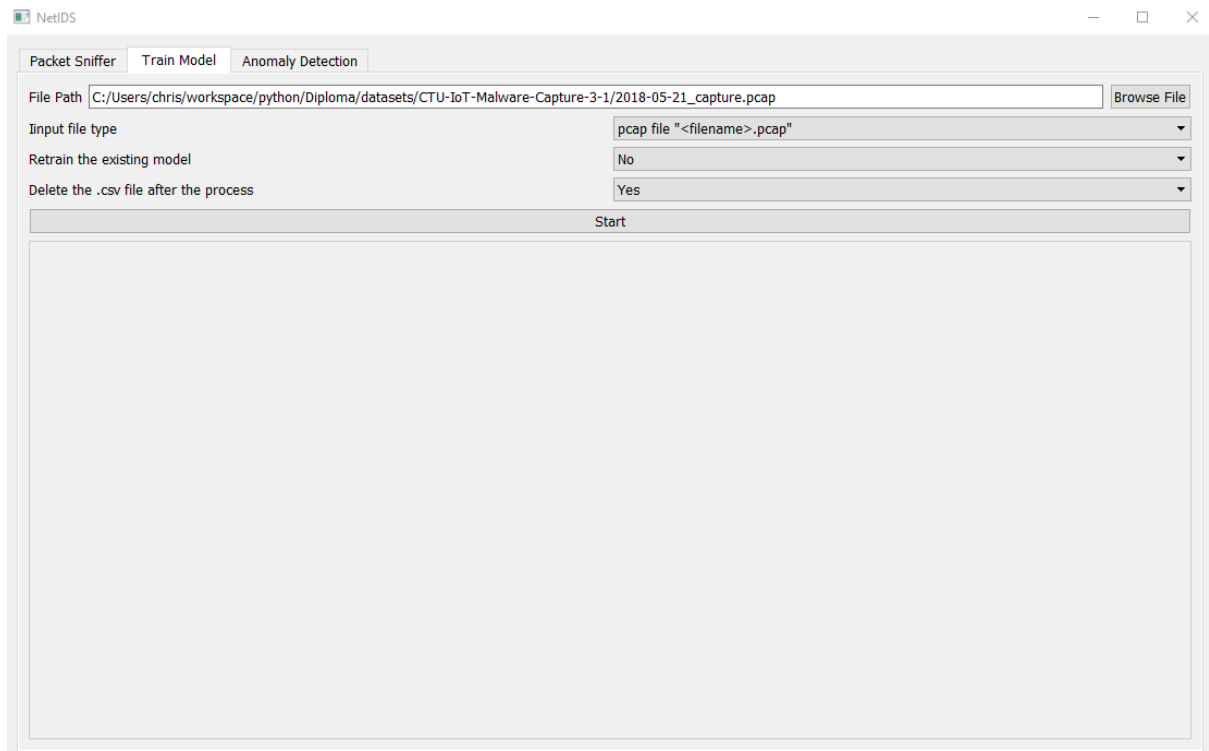
Στην Εικόνα 4.20 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της εκπαίδευσης του μοντέλου του αλγορίθμου Isolation Forest αφού επιλέχτηκε η επιλογή Delete the .csv file after the process, όπου το πρόγραμμα θα διαγράψει τα αρχεία που περιέχουν τα εξαγόμενα χαρακτηριστικά (extracted features) μετά την ολοκλήρωση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων και Τροφοδοσία των δεδομένων στον αλγόριθμο). Σημειώνεται ότι για να γίνει αυτή η επιλογή τροποποιήσιμη πρέπει πρώτα να επιλεχτεί η επιλογή Input file type, όπου πρέπει να καθοριστεί ο τύπος του αρχείου εισόδου σε pcap.



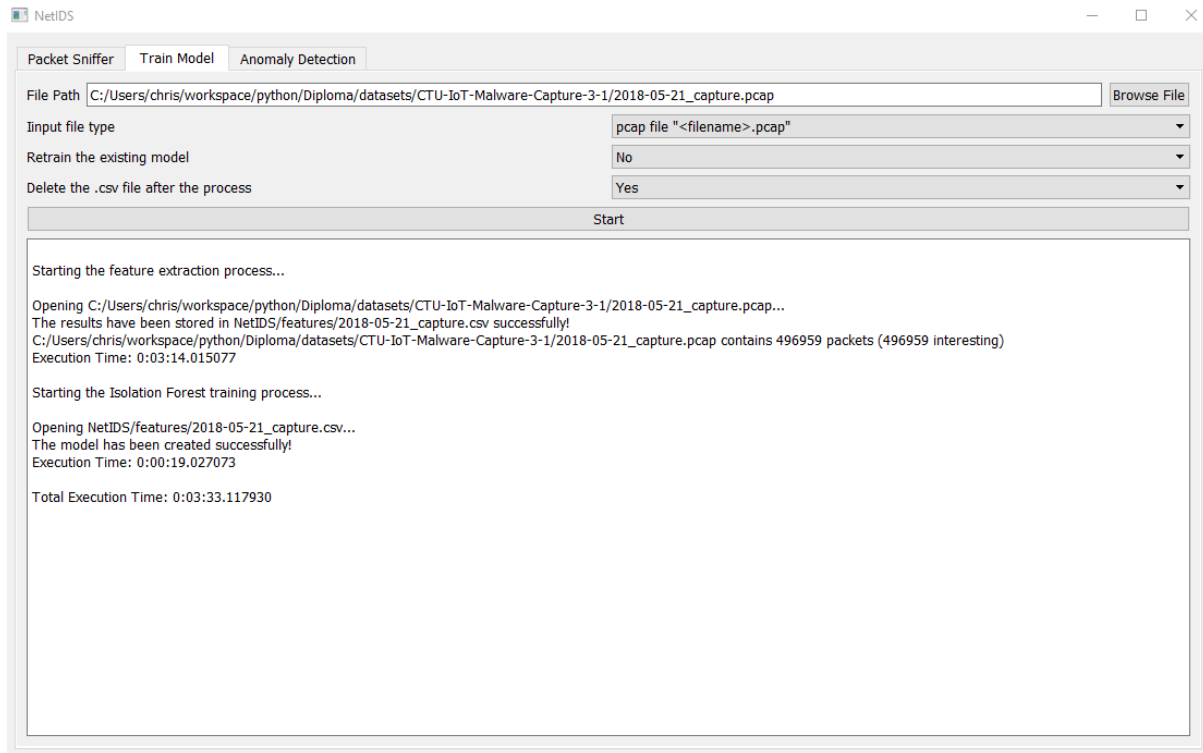
Εικόνα 4.11 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



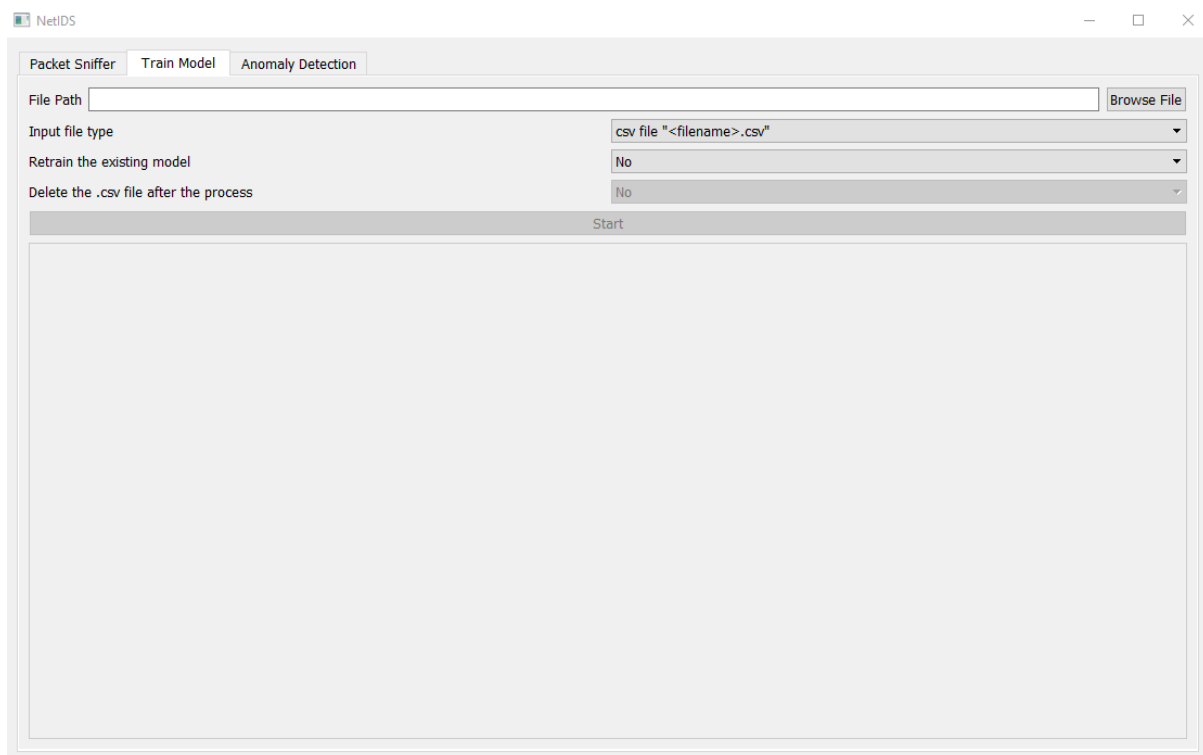
Εικόνα 4.12 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



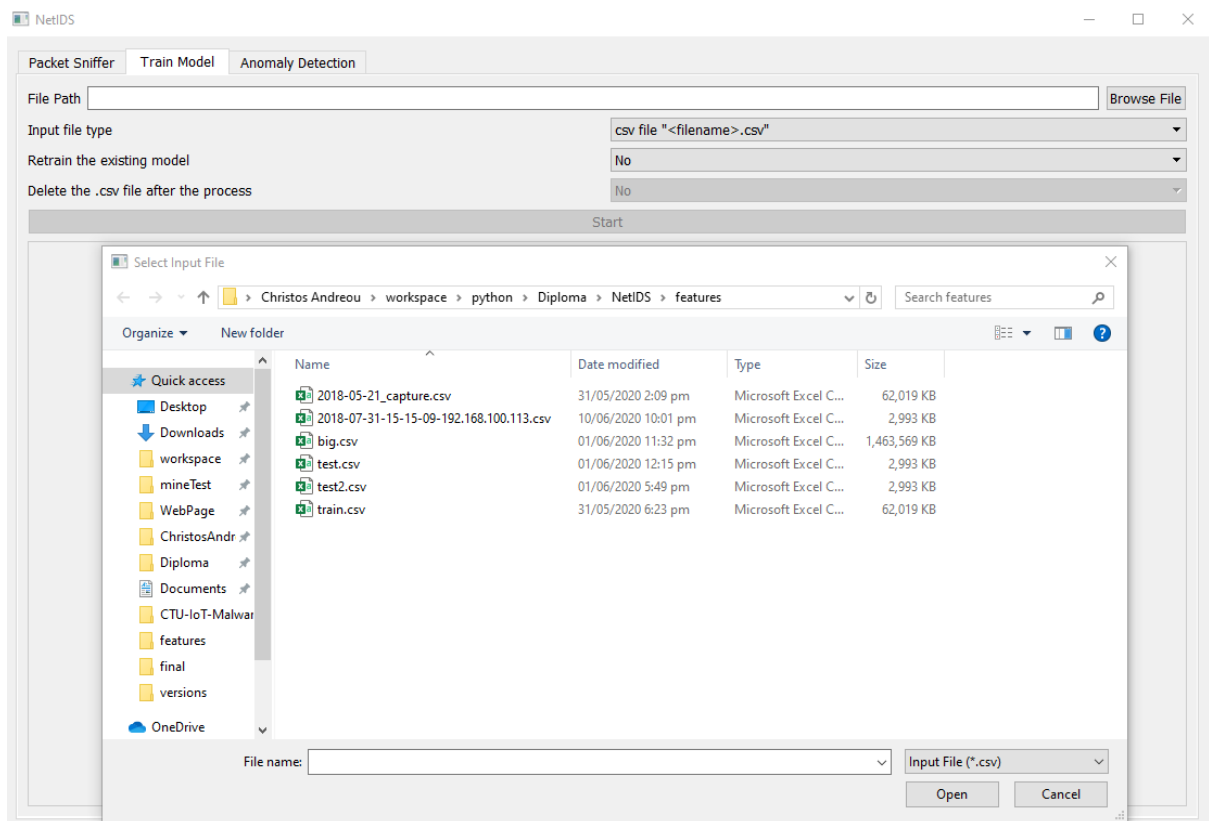
Εικόνα 4.13 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



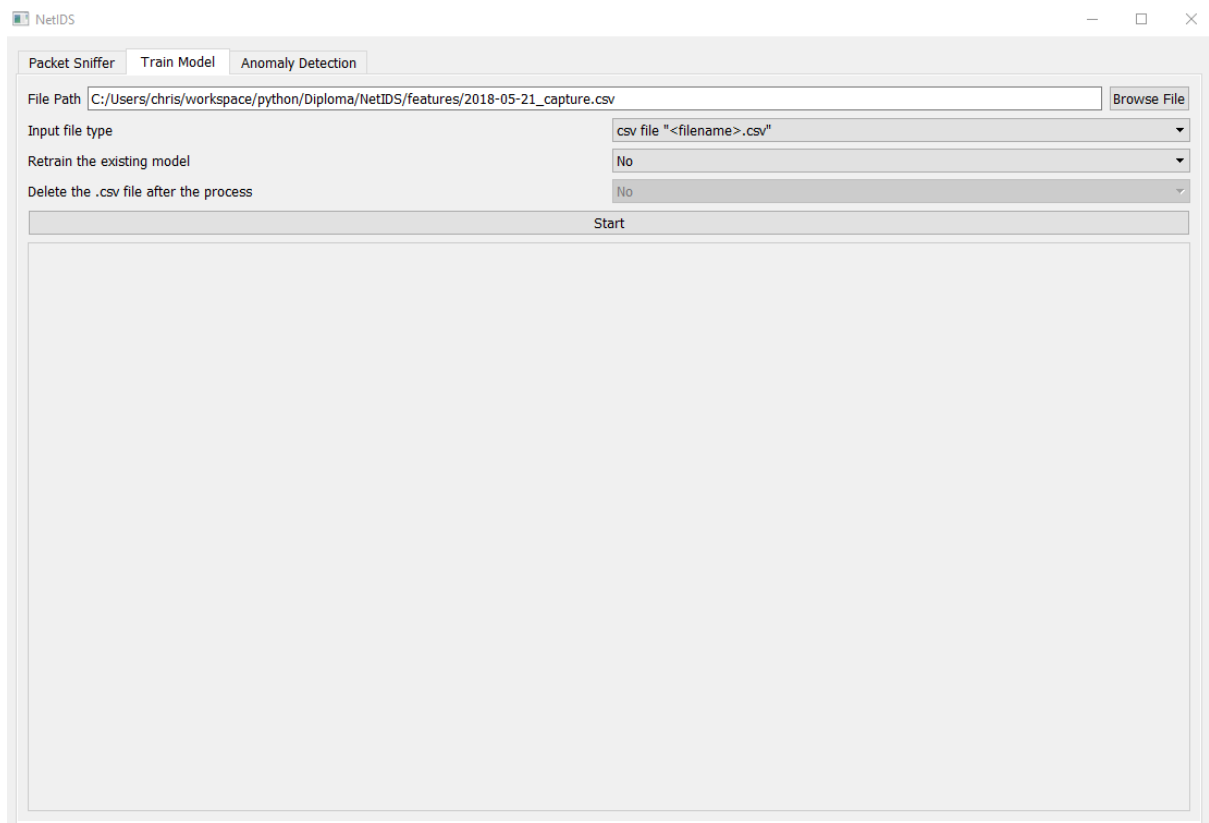
Εικόνα 4.14 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



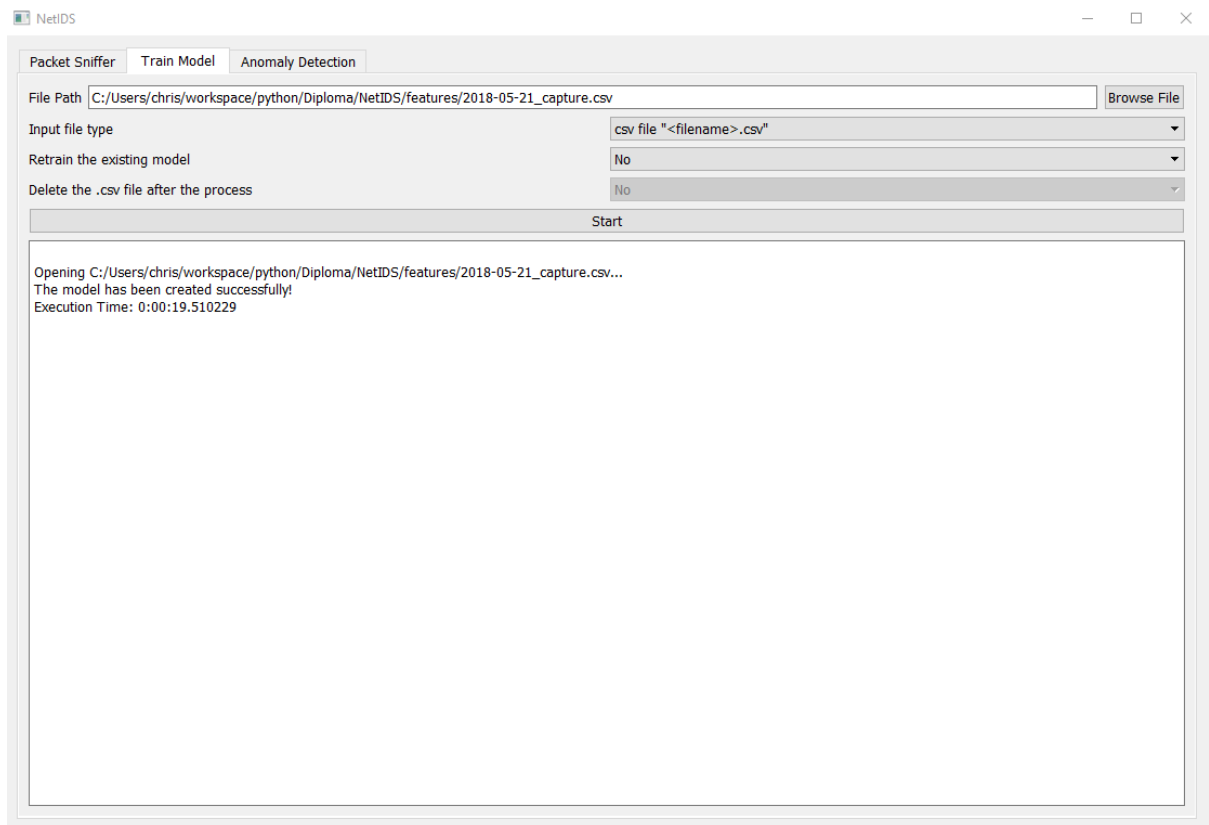
Εικόνα 4.15 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



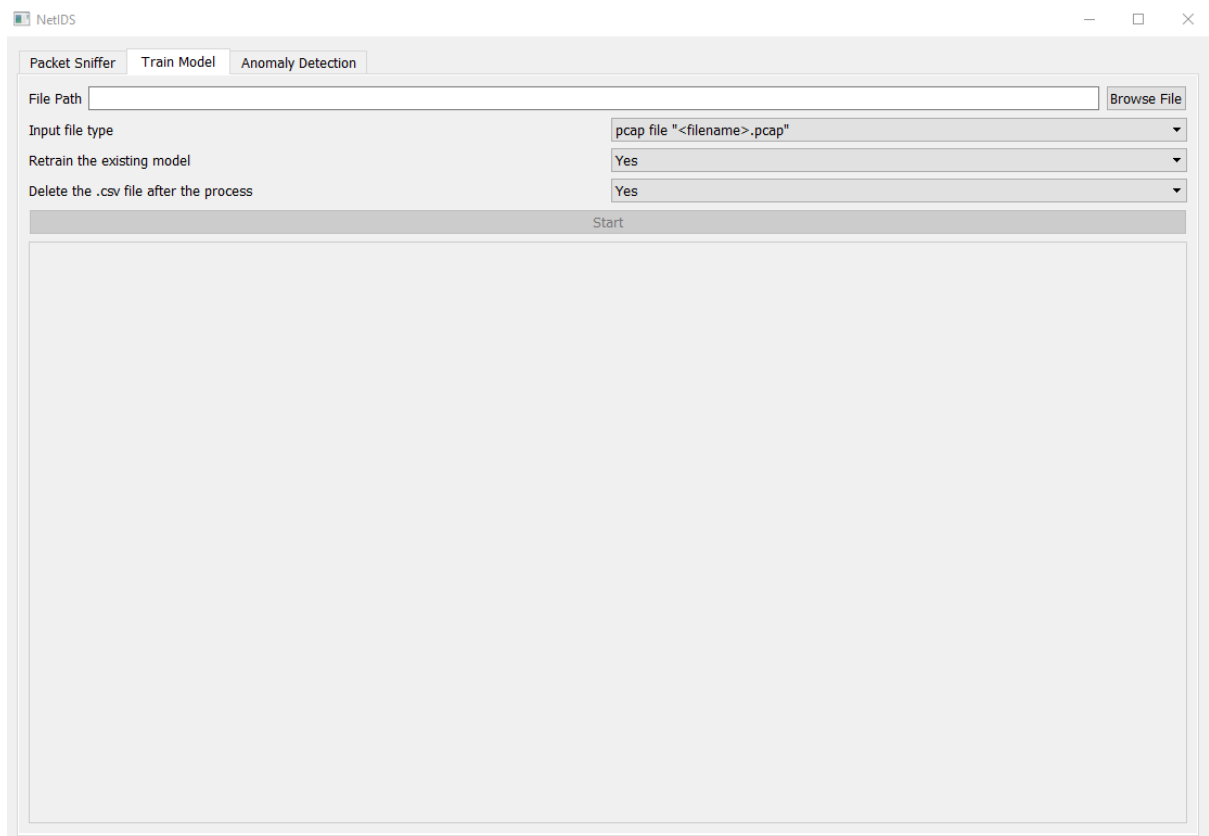
Εικόνα 4.16 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



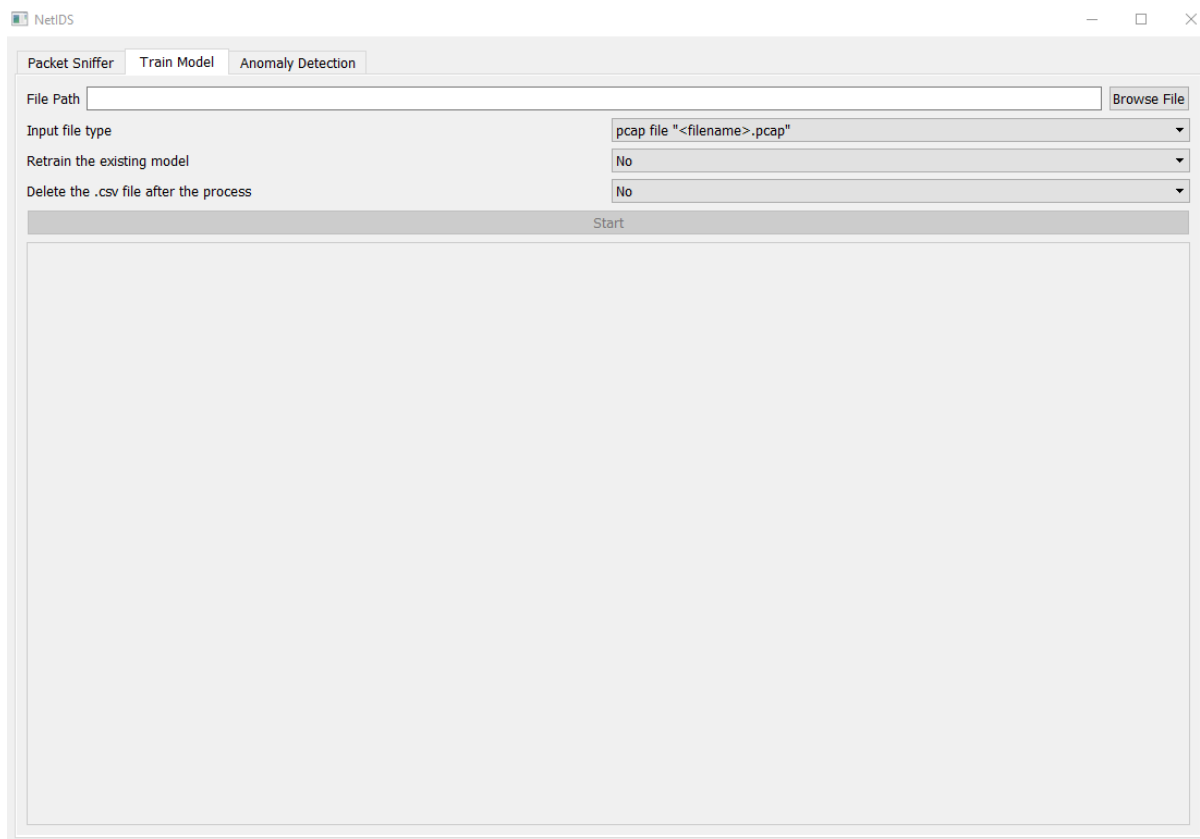
Εικόνα 4.17 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



Εικόνα 4.18 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



Εικόνα 4.19 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης



Εικόνα 4.20 Γραφικό περιβάλλον διαδικασίας εκπαίδευσης

4.4 Ανίχνευση επιθέσεων

Σε αυτό το σημείο παρουσιάζεται το γραφικό περιβάλλον χρήστη μέσω του οποίου έχει την δυνατότητα να χρησιμοποιήσει το ήδη εκπαιδευμένο μοντέλο του αλγορίθμου Isolation Forest για ανίχνευση επιθέσεων καθώς και άλλες επιλογές που έχουν επεξηγηθεί στο Κεφάλαιο 3.4.

Στην Εικόνα 4.21 παρουσιάζεται το προεπιλεγμένο γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων.

Στην Εικόνα 4.22 παρουσιάζεται το παράθυρο που εμφανίζεται για την επιλογή αρχείου εισόδου αφού έχει πατηθεί το κουμπί Browse File.

Στην Εικόνα 4.23 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού έγινε η επιλογή αρχείου εισόδου.

Στην Εικόνα 4.24 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού έτρεξε με τις προεπιλεγμένες επιλογές.

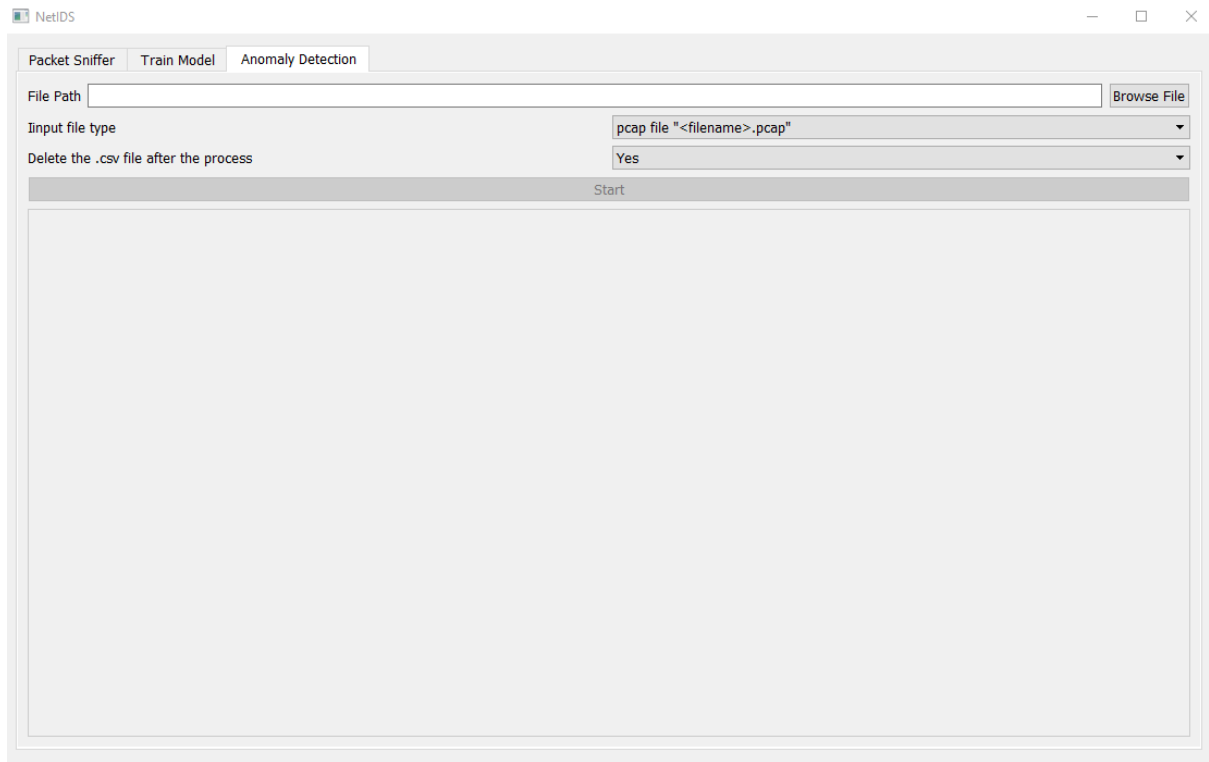
Στην Εικόνα 4.25 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού επιλέχτηκε η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

Στην Εικόνα 4.26 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού έχει πατηθεί το κουμπί Browse File. Σημειώνεται ότι έχει ήδη επιλεγεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

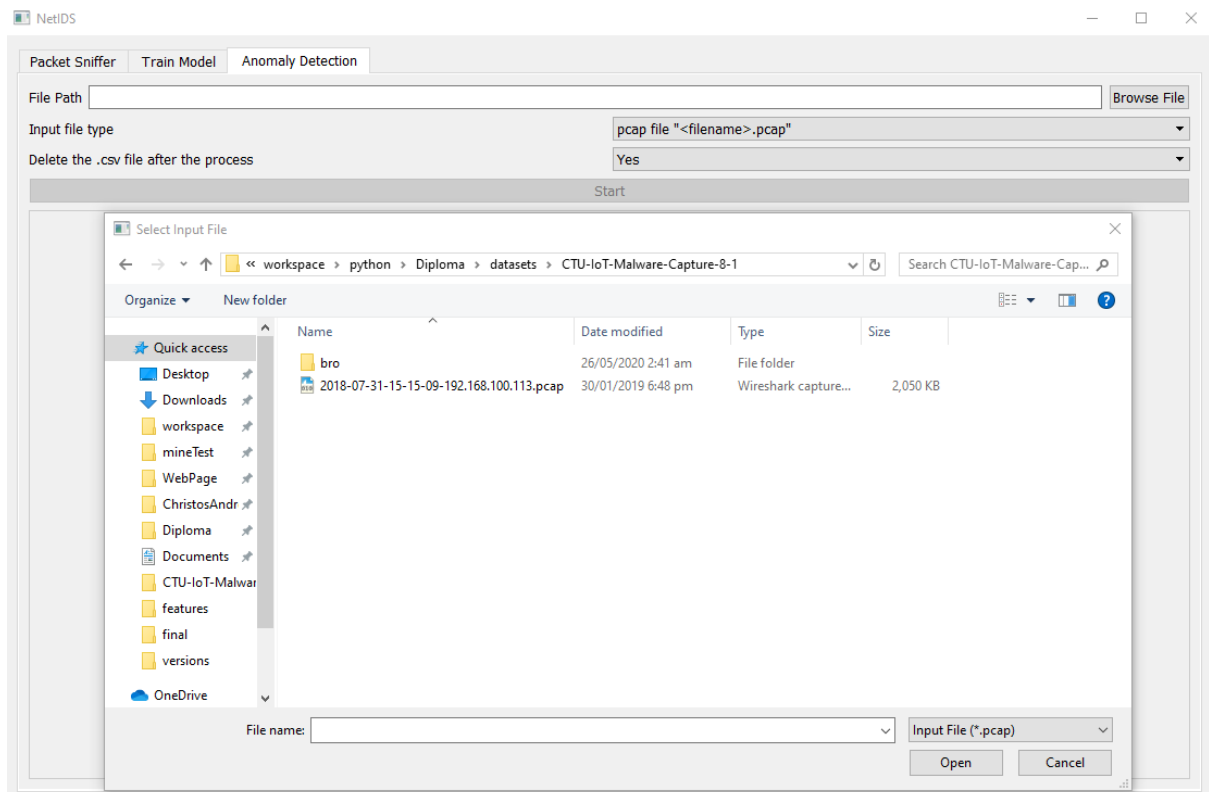
Στην Εικόνα 4.27 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού έγινε η επιλογή αρχείου εισόδου. Σημειώνεται ότι έχει ήδη επιλεγεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

Στην Εικόνα 4.28 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού έτρεξε με τις προεπιλεγμένες επιλογές. Σημειώνεται ότι έχει ήδη επιλεγεί η επιλογή Input file type, όπου καθόρισε τον τύπο του αρχείου εισόδου σε csv.

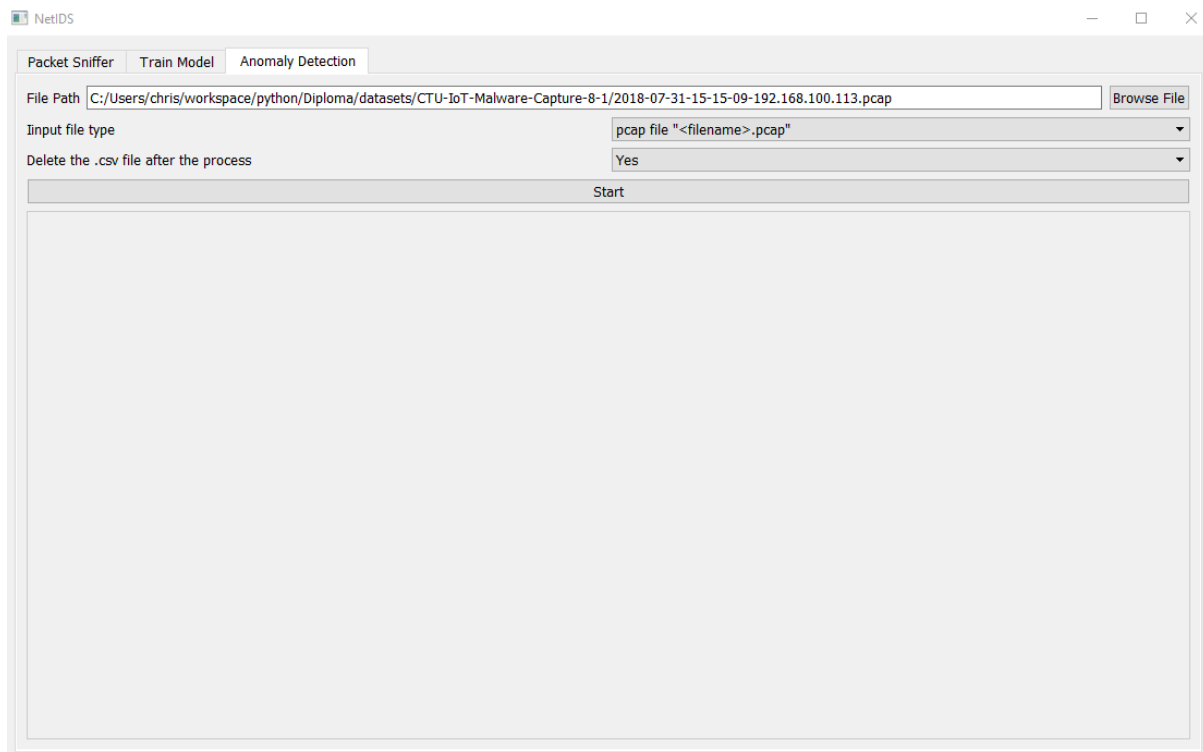
Στην Εικόνα 4.29 παρουσιάζεται το γραφικό περιβάλλον για την λειτουργία της ανίχνευσης επιθέσεων αφού επιλέχτηκε η επιλογή Delete the .csv file after the process, όπου το πρόγραμμα θα διαγράψει τα αρχεία που περιέχουν τα εξαγόμενα χαρακτηριστικά (extracted features) μετά την ολοκλήρωση των υπόλοιπων λειτουργιών (Προετοιμασία Δεδομένων και Τροφοδοσία των δεδομένων στον αλγόριθμο). Σημειώνεται ότι για να γίνει αυτή η επιλογή τροποποιήσιμη πρέπει πρώτα να επιλεγεί η επιλογή Input file type, όπου πρέπει να καθοριστεί ο τύπος του αρχείου εισόδου σε pcap.



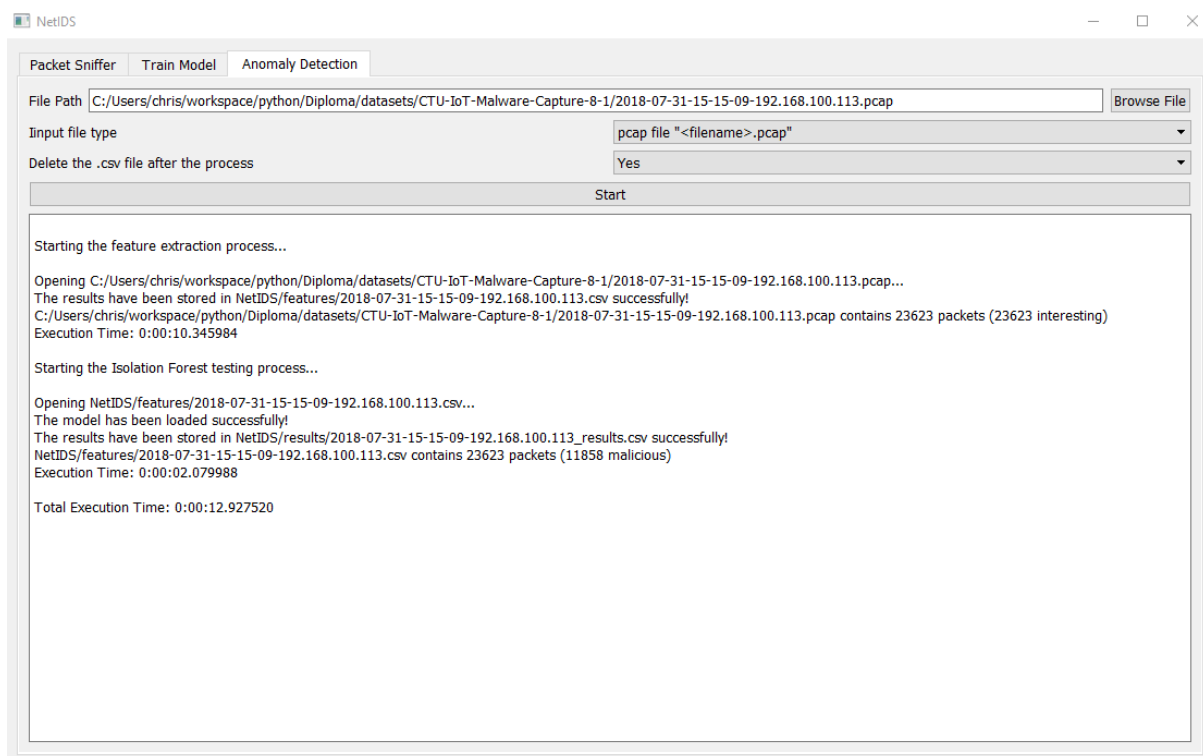
Εικόνα 4.21 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



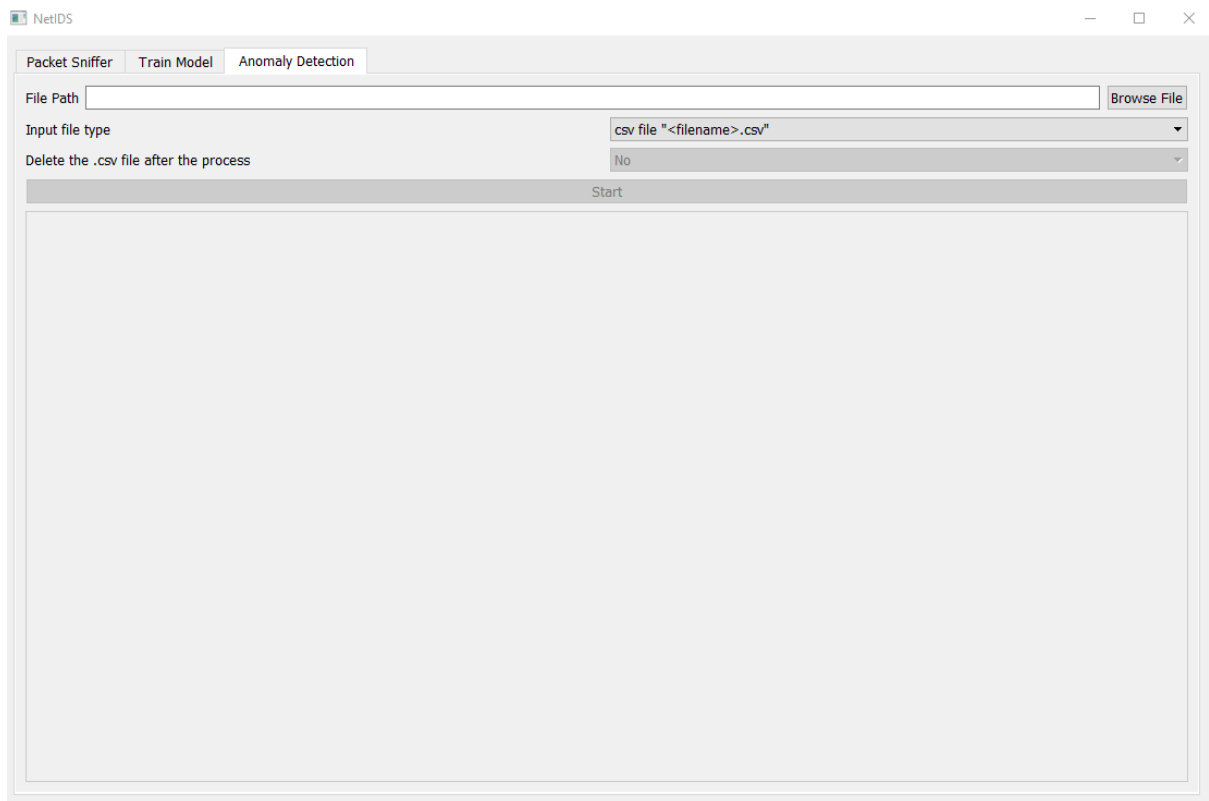
Εικόνα 4.22 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



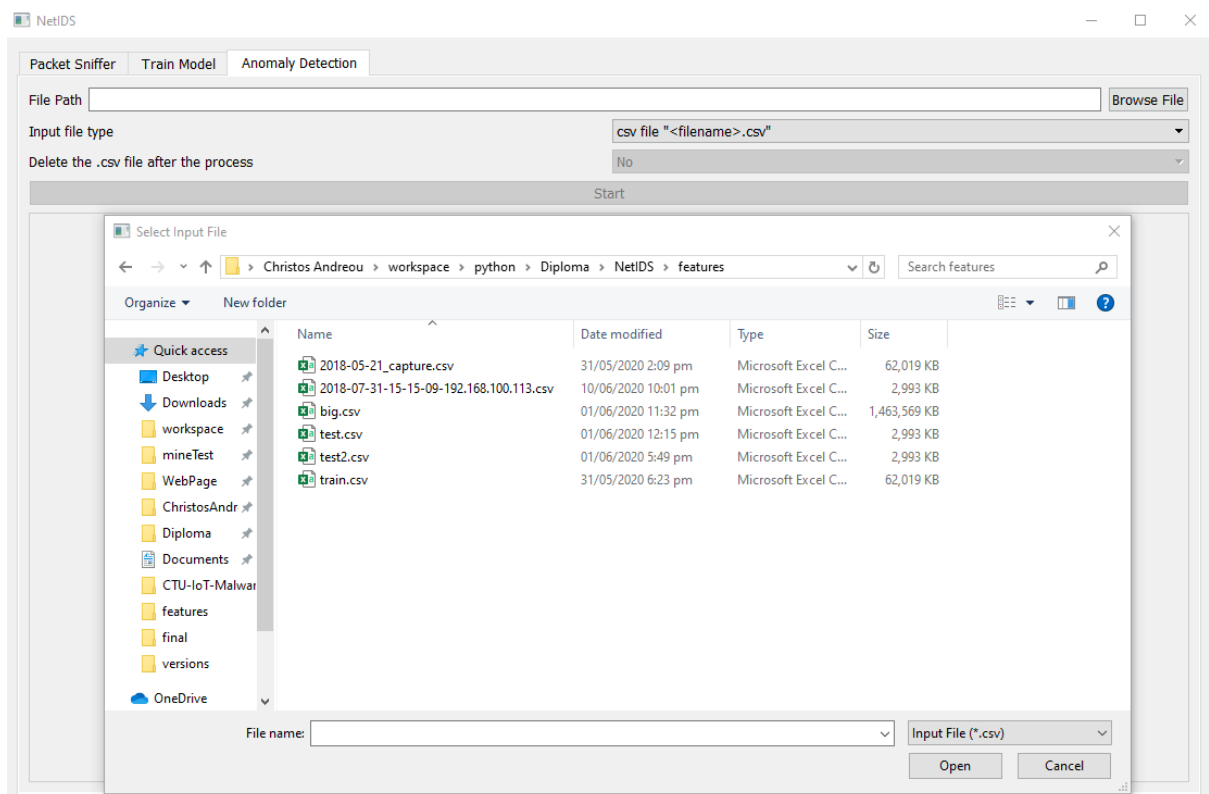
Εικόνα 4.23 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



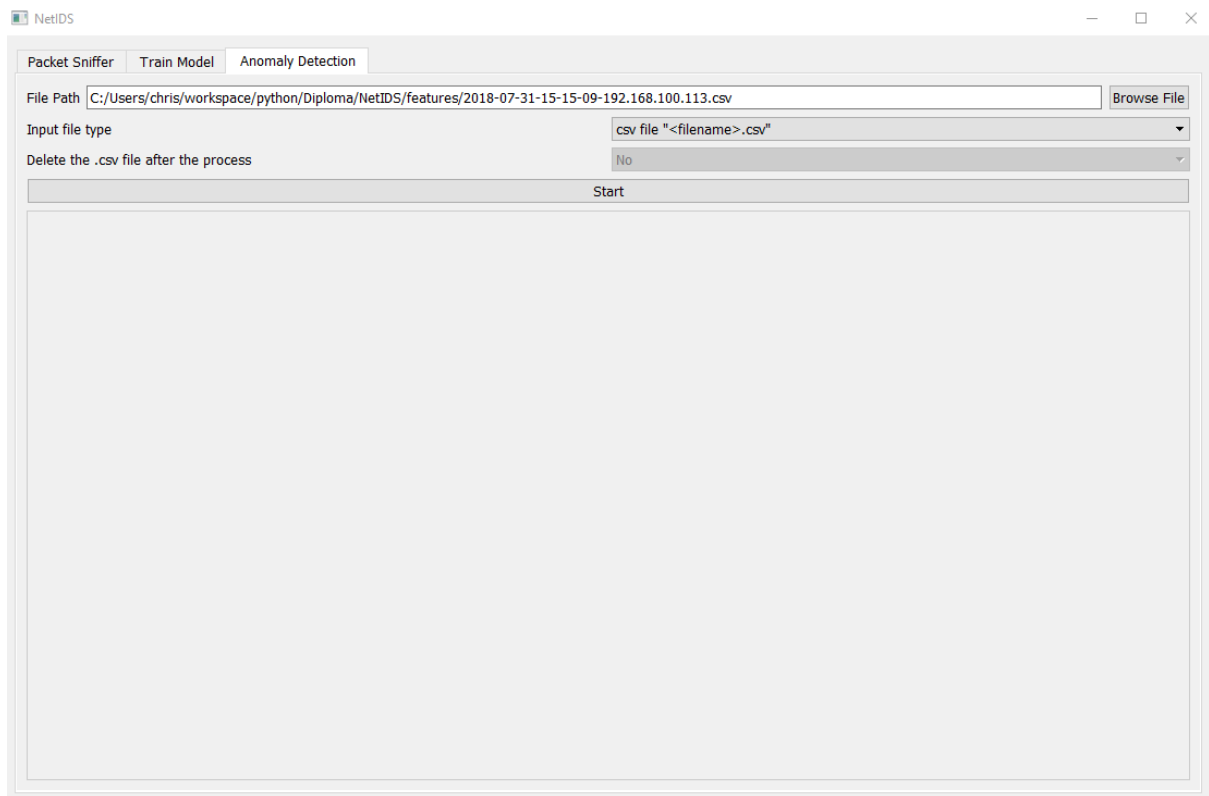
Εικόνα 4.24 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



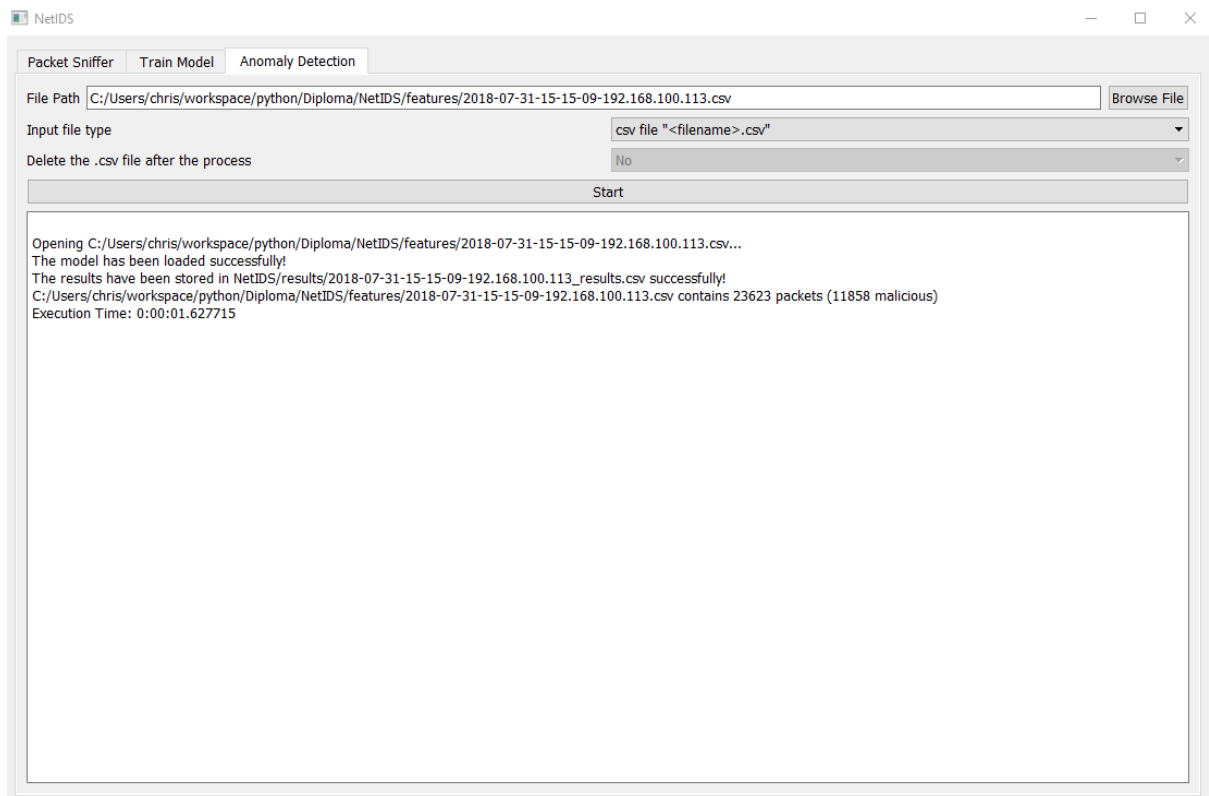
Εικόνα 4.25 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



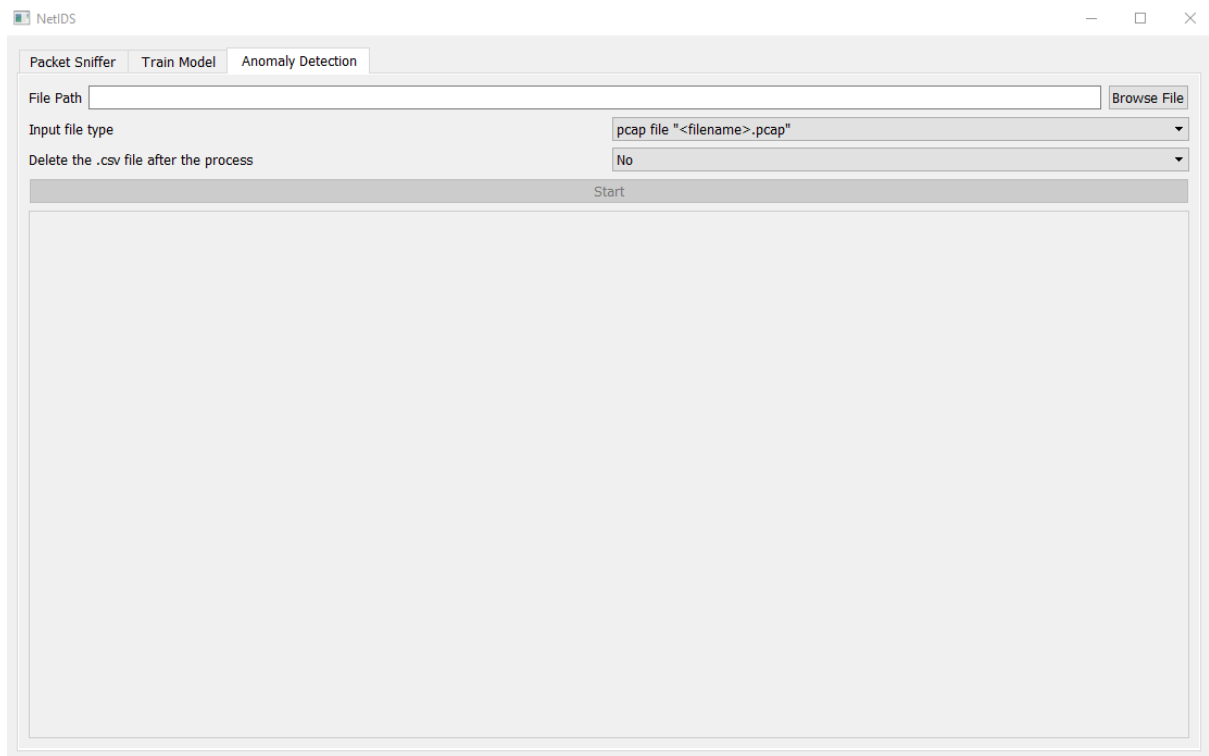
Εικόνα 4.26 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



Εικόνα 4.27 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



Εικόνα 4.28 Γραφικό περιβάλλον ανίχνευσης επιθέσεων



Εικόνα 4.29 Γραφικό περιβάλλον ανίχνευσης επιθέσεων

Κεφάλαιο 5

Αποτελέσματα

5.1 Μετρήσεις	38
5.2 Σύγκριση αποτελεσμάτων	39

5.1 Μετρήσεις

Τρέχοντας τον αλγόριθμο για ανίχνευση ανωμαλιών παράχθηκαν τα αποτελέσματα που φαίνονται στις πιο κάτω γραφικές παραστάσεις. Σε γενικές γραμμές ο αλγόριθμός πλησιάζει τα επιθυμητά αποτελέσματα αλλά δεν μπορεί να τα φτάσει στο σημείο που να μπορεί να διακρίνει με απόλυτη ακρίβεια και σιγουριά το κατά πόσο ένα πακέτο είναι κακόβουλο ή όχι.



Εικόνα 5.1 1^η Γραφική παράσταση



Εικόνα 5.2 2^η Γραφική παράσταση

5.2 Σύγκριση αποτελεσμάτων

Μελετώντας τόσο τις γραφικές παραστάσεις, Εικόνα 5.1 και Εικόνα 5.2, που παράχθηκαν όσο και την απόδοση του αλγορίθμου είναι ξεκάθαρο πως παρατηρείται μια αισθητή διαφορά στην ακρίβεια του αλγορίθμου. Πιο συγκεκριμένα η ακρίβεια του αλγορίθμου Isolation Forest είναι 69%. Αυτό σημαίνει ότι παρόλο που έχει περάσει το κατώφλι του 50% εξακολουθεί να μην δίνει ένα ποσοστό το οποίο να εμπνέει σιγουριά ότι το σύστημα αυτό είναι ικανό να αναγνωρίσει σχεδόν όλες τις επιθέσεις και τα είδη επιθέσεων που υπάρχουν.

Κεφάλαιο 6

Συμπεράσματα

6.1 Παρατηρήσεις	40
6.2 Συμπεράσματα	40
6.3 Μελλοντική Εργασία	41

6.1 Παρατηρήσεις

Παρά την αξιοσημείωτη περιγραφή του αλγορίθμου Isolation Forest για την γραμμική χρονική πολυπλοκότητα, την αποδοτικότητα και την χαμηλή υπολογιστική δύναμη που χρειάζεται για να τρέξει, ο αλγόριθμος δεν είναι αρκετά ικανός να επιλύσει τέτοιου είδους προβλήματα στην απλή του μορφή. Ωστόσο αξίζει να σημειωθεί πως όλα όσα περιγράφουν τον αλγόριθμο αυτό, είναι πραγματικά αληθές και ξεπερνούν κάθε προσδοκία.

6.2 Συμπεράσματα

Είναι προφανές πως το απλό μοντέλο του αλγορίθμου Isolation Forest δεν είναι αρκετά ικανό για την επίλυση τέτοιου είδους προβλημάτων με υψηλή ακρίβεια. Σύμφωνα και με τις προαναφερθείσες δοκιμές που έχουν γίνει σε αυτή την έρευνα όσο και σε άλλες έρευνες που είχαν γίνει στο παρελθόν η ανάγκη τροποποίησης του αλγορίθμου αυτού είναι εμφανής. Συγκεκριμένα ένα υβριδικό μοντέλο (hybrid model) του αλγορίθμου Isolation Forest είναι απαραίτητο για την επίλυση προβλημάτων ανίχνευσης επιθέσεων. Εντούτοις ένα τέτοιο μοντέλο (hybrid model) θα πρέπει να σχεδιαστεί με απώτερο σκοπό να παραμείνουν όλα τα θετικά και αξιοσημείωτα χαρακτηριστικά που περιγράφουν το απλό μοντέλο του αλγορίθμου Isolation Forest έτσι ώστε να μπορεί το μοντέλο αυτό, το υβριδικό μοντέλο (hybrid model), να τρέχει σε συσκευές IoT. Στην παρούσα φάση τα μοντέλα που έχουν προταθεί για τροποποίηση του απλού μοντέλου Isolation Forest μπορεί από τη μια πλευρά να

επιτυγχάνουν υψηλότερη ακρίβεια με μειονέκτημα όμως ότι δεν είναι πλέον τόσο αποδοτικά και ούτε έχουν τόσο χαμηλή υπολογιστική ανάγκη. Καθιστώντας τα έτσι μη συμβατές επιλογές για συσκευές IoT όπου η χαμηλή υπολογιστική ανάγκη είναι ένα από τα κύρια προαπαιτούμενα.

6.3 Μελλοντική Εργασία

Σίγουρα η έρευνα αυτή θα μπορούσε να αποτελέσει τα θεμέλια για συνέχιση και επέκταση του εργαλείου που προτάθηκε.

Μία πτυχή του εργαλείου αυτού που θα μπορούσε να αναπτυχθεί σε μεγάλο βαθμό που όμως θα απαιτούσε απίστευτη γνώση και εξειδίκευση στο κομμάτι των αλγορίθμων και πολυπλοκότητας είναι η δημιουργία ενός υβριδικού μοντέλου (hybrid model) ή ενός τροποποιημένου μοντέλου του αλγορίθμου Isolation Forest που να είναι σχεδιασμένο για τα μέτρα μίας συσκευής IoT.

Ακόμα μία πτυχή του εργαλείου που θα μπορούσε να αναπτυχθεί είναι η χρήση παράλληλης επεξεργασίας για μείωση του χρόνου διεκπεραίωσης των διαδικασιών.

Βιβλιογραφία

- [1] Dimitar Karev, Christopher McCubbin, and Ruslan Vaulin. 2017. Cyber Threat Hunting Through the Use of an Isolation Forest. In Proceedings of the 18th International Conference on Computer Systems and Technologies (CompSysTech'17). Association for Computing Machinery, New York, NY, USA, 163–170. DOI:<https://doi.org/10.1145/3134302.3134319>.
- [2] F. T. Liu, K. M. Ting and Z. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17.
- [3] Ren, H. (2019). Building an Effective Intrusion Detection System by Using Hybrid Data Optimization Based on Machine Learning Algorithms Security and Communication Networks, 2019, 7130868.
- [4] Sun, Li & Versteeg, Steven & Boztas, Serdar & Rao, Asha. (2016). Detecting Anomalous User Behavior Using an Extended Isolation Forest Algorithm: An Enterprise Case Study.
- [5] Marteau, Pierre-Francois & SOHEILY-KHAH, Saeid & Béchet, Nicolas. (2017). Hybrid Isolation Forest - Application to Intrusion Detection
- [6] Mocnej, J. (2018). Network Traffic Characteristics of the IoT Application Use Cases.
- [7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. ACM Comput. Surv., 41(3):15:1–15:58, July 2009.
- [8] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PLOS ONE, 11(4):1–31, 04 2016.
- [9] Mennatallah Amer and Markus Goldstein. Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer. In Simon Fischer and Ingo Mierswa,

editors, Proceedings of the 3rd RapidMiner Community Meeting and Conference (RCOMM 2012), pages 1–12. Shaker Verlag GmbH, 8 2012.

- [10] Chesner Desir, Simon Bernard, Caroline Petitjean, and Laurent Heutte. One class random forests. *Pattern Recogn.*, 46(12):3490–3506, December 2013.
- [11] Michael Kemmler, Erik Rodner, Esther-Sabrina Wacker, and Joachim Denzler. One-class classification with gaussian processes. *Pattern Recognition*, 46(12):3507 – 3518, 2013.
- [12] Wenke Lee and Dong Xiang. Information-theoretic measures for anomaly detection. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy, SP '01*, pages 130–, Washington, DC, USA, 2001. IEEE Computer Society.
- [13] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: A survey. *Data Min. Knowl. Discov.*, 29(3):626–688, May 2015.
- [14] Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708 – 713, 2015.
- [15] Yanhui Shen, Huawen Liu, Yanxia Wang, Zhongyu Chen, and Guanghua Sun. A Novel Isolation-Based Outlier Detection Method, pages 446–456. Springer International Publishing, Cham, 2016.
- [16] Biondi, P. (n.d.). Scapy. Retrieved from <https://scapy.net/>
- [17] pandas. (n.d.). Retrieved from <https://pandas.pydata.org/>
- [18] scikit-learn (n.d.). Retrieved from <https://scikit-learn.org/stable/>
- [19] Joblib: running Python functions as pipeline jobs. (n.d.). Retrieved from <https://joblib.readthedocs.io/en/latest/>
- [20] PyQt5. (n.d.). Retrieved from <https://pypi.org/project/PyQt5/>

Παράρτημα Α

```
# -*- coding: utf-8 -*-
#####
# Author:      Christos Andreou      #
# Project Name: NetIDS                #
# Date:        June 2020             #
# University:   University of Cyprus #
#####

from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QMessageBox
import os
import sys
from scapy.all import *
import pandas as pd
import numpy as np
from sklearn.ensemble import IsolationForest
import joblib
from scapy.utils import RawPcapReader
from scapy.layers.l2 import Ether, ARP
from scapy.layers.inet import IP, TCP, UDP, TCPErrors, UDPErrors
import time
import datetime

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(1114, 670)
        font = QtGui.QFont()
        font.setPointSize(10)
        Form.setFont(font)
        self.verticalLayout_4 = QtWidgets.QVBoxLayout(Form)
        self.verticalLayout_4.setObjectName("verticalLayout_4")
        self.tabWidget = QtWidgets.QTabWidget(Form)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.tabWidget.setFont(font)
        self.tabWidget.setAutoFillBackground(True)
        self.tabWidget.setTabsClosable(False)
        self.tabWidget.setObjectName("tabWidget")
        self.Tool1 = QtWidgets.QWidget()
        self.Tool1.setAutoFillBackground(True)
        self.Tool1.setObjectName("Tool1")
        self.verticalLayout = QtWidgets.QVBoxLayout(self.Tool1)
        self.verticalLayout.setObjectName("verticalLayout")
        self.hl_1_Tool1 = QtWidgets.QHBoxLayout()
        self.hl_1_Tool1.setObjectName("hl_1_Tool1")
        self.lblInfinitely_Tool1 = QtWidgets.QLabel(self.Tool1)
        font = QtGui.QFont()
        font.setPointSize(10)
        self.lblInfinitely_Tool1.setFont(font)
        self.lblInfinitely_Tool1.setObjectName("lblInfinitely_Tool1")
        self.hl_1_Tool1.addWidget(self.lblInfinitely_Tool1)
```

```

self.comboBoxInfinitely_Tool1 = QtWidgets.QComboBox(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxInfinitely_Tool1.setFont(font)
self.comboBoxInfinitely_Tool1.setEditable(False)
self.comboBoxInfinitely_Tool1.setObjectName("comboBoxInfinitely_Tool1")
self.comboBoxInfinitely_Tool1.addItem("")
self.comboBoxInfinitely_Tool1.addItem("")
self.hl_1_Tool1.addWidget(self.comboBoxInfinitely_Tool1)
self.verticalLayout.addLayout(self.hl_1_Tool1)
self.hl_2_Tool1 = QtWidgets.QHBoxLayout()
self.hl_2_Tool1.setObjectName("hl_2_Tool1")
self.lblDuration_Tool1 = QtWidgets.QLabel(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.lblDuration_Tool1.setFont(font)
self.lblDuration_Tool1.setObjectName("lblDuration_Tool1")
self.hl_2_Tool1.addWidget(self.lblDuration_Tool1)
self.txtDuration_Tool1 = QtWidgets.QLineEdit(self.Tool1)
sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets.QSizePolicy.Fixed)
sizePolicy.setHorizontalStretch(0)
sizePolicy.setVerticalStretch(0)
sizePolicy.setHeightForWidth(self.txtDuration_Tool1.sizePolicy().hasHeightForWidth())
self.txtDuration_Tool1.setSizePolicy(sizePolicy)
font = QtGui.QFont()
font.setPointSize(10)
self.txtDuration_Tool1.setFont(font)
self.txtDuration_Tool1.setObjectName("txtDuration_Tool1")
self.hl_2_Tool1.addWidget(self.txtDuration_Tool1)
self.verticalLayout.addLayout(self.hl_2_Tool1)
self.hl_3_Tool1 = QtWidgets.QHBoxLayout()
self.hl_3_Tool1.setObjectName("hl_3_Tool1")
self.lblSinglePcap_Tool1 = QtWidgets.QLabel(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.lblSinglePcap_Tool1.setFont(font)
self.lblSinglePcap_Tool1.setObjectName("lblSinglePcap_Tool1")
self.hl_3_Tool1.addWidget(self.lblSinglePcap_Tool1)
self.comboBoxSinglePcap_Tool1 = QtWidgets.QComboBox(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxSinglePcap_Tool1.setFont(font)
self.comboBoxSinglePcap_Tool1.setObjectName("comboBoxSinglePcap_Tool1")
self.comboBoxSinglePcap_Tool1.addItem("")
self.comboBoxSinglePcap_Tool1.addItem("")
self.hl_3_Tool1.addWidget(self.comboBoxSinglePcap_Tool1)
self.verticalLayout.addLayout(self.hl_3_Tool1)
self.hl_4_Tool1 = QtWidgets.QHBoxLayout()
self.hl_4_Tool1.setObjectName("hl_4_Tool1")
self.lblPacketNum_Tool1 = QtWidgets.QLabel(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.lblPacketNum_Tool1.setFont(font)
self.lblPacketNum_Tool1.setObjectName("lblPacketNum_Tool1")
self.hl_4_Tool1.addWidget(self.lblPacketNum_Tool1)
self.txtPacketNum_Tool1 = QtWidgets.QLineEdit(self.Tool1)
sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets.QSizePolicy.Fixed)
sizePolicy.setHorizontalStretch(0)
sizePolicy.setVerticalStretch(0)
sizePolicy.setHeightForWidth(self.txtPacketNum_Tool1.sizePolicy().hasHeightForWidth())
self.txtPacketNum_Tool1.setSizePolicy(sizePolicy)
font = QtGui.QFont()
font.setPointSize(10)
self.txtPacketNum_Tool1.setFont(font)

```

```

self.txtPacketNum_Tool1.setObjectName("txtPacketNum_Tool1")
self.hl_4_Tool1.addWidget(self.txtPacketNum_Tool1)
self.verticalLayout.addLayout(self.hl_4_Tool1)
self.hl_5_Tool1 = QtWidgets.QHBoxLayout()
self.hl_5_Tool1.setObjectName("hl_5_Tool1")
self.lblContinue_Tool1 = QtWidgets.QLabel(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.lblContinue_Tool1.setFont(font)
self.lblContinue_Tool1.setObjectName("lblContinue_Tool1")
self.hl_5_Tool1.addWidget(self.lblContinue_Tool1)
self.comboBoxContinue_Tool1 = QtWidgets.QComboBox(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxContinue_Tool1.setFont(font)
self.comboBoxContinue_Tool1.setObjectName("comboBoxContinue_Tool1")
self.comboBoxContinue_Tool1.addItem("")
self.comboBoxContinue_Tool1.addItem("")
self.hl_5_Tool1.addWidget(self.comboBoxContinue_Tool1)
self.verticalLayout.addLayout(self.hl_5_Tool1)
self.hl_6_Tool1 = QtWidgets.QHBoxLayout()
self.hl_6_Tool1.setObjectName("hl_6_Tool1")
self.lblDelete_Tool1 = QtWidgets.QLabel(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.lblDelete_Tool1.setFont(font)
self.lblDelete_Tool1.setObjectName("lblDelete_Tool1")
self.hl_6_Tool1.addWidget(self.lblDelete_Tool1)
self.comboBoxDelete_Tool1 = QtWidgets.QComboBox(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxDelete_Tool1.setFont(font)
self.comboBoxDelete_Tool1.setObjectName("comboBoxDelete_Tool1")
self.comboBoxDelete_Tool1.addItem("")
self.comboBoxDelete_Tool1.addItem("")
self.hl_6_Tool1.addWidget(self.comboBoxDelete_Tool1)
self.verticalLayout.addLayout(self.hl_6_Tool1)
self.btnStart_Tool1 = QtWidgets.QPushButton(self.Tool1)
font = QtGui.QFont()
font.setPointSize(10)
self.btnStart_Tool1.setFont(font)
self.btnStart_Tool1.setObjectName("btnStart_Tool1")
self.verticalLayout.addWidget(self.btnStart_Tool1)
self.textBrowser_Tool1 = QtWidgets.QTextBrowser(self.Tool1)
self.textBrowser_Tool1.setObjectName("textBrowser_Tool1")
self.verticalLayout.addWidget(self.textBrowser_Tool1)
self.tabWidget.addTab(self.Tool1, "")
self.Tool2 = QtWidgets.QWidget()
self.Tool2.setAutoFillBackground(True)
self.Tool2.setObjectName("Tool2")
self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.Tool2)
self.verticalLayout_3.setObjectName("verticalLayout_3")
self.hl_1_Tool2 = QtWidgets.QHBoxLayout()
self.hl_1_Tool2.setObjectName("hl_1_Tool2")
self.lblPath_Tool2 = QtWidgets.QLabel(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.lblPath_Tool2.setFont(font)
self.lblPath_Tool2.setObjectName("lblPath_Tool2")
self.hl_1_Tool2.addWidget(self.lblPath_Tool2)
self.txtPath_Tool2 = QtWidgets.QLineEdit(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.txtPath_Tool2.setFont(font)
self.txtPath_Tool2.setObjectName("txtPath_Tool2")

```

```

self.hl_1_Tool2.addWidget(self.txtPath_Tool2)
self.btnBrowse_Tool2 = QtWidgets.QPushButton(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.btnBrowse_Tool2.setFont(font)
self.btnBrowse_Tool2.setObjectName("btnBrowse_Tool2")
self.hl_1_Tool2.addWidget(self.btnBrowse_Tool2)
self.verticalLayout_3.addLayout(self.hl_1_Tool2)
self.hl_2_Tool2 = QtWidgets.QHBoxLayout()
self.hl_2_Tool2.setObjectName("hl_2_Tool2")
self.lblFileType_Tool2 = QtWidgets.QLabel(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.lblFileType_Tool2.setFont(font)
self.lblFileType_Tool2.setObjectName("lblFileType_Tool2")
self.hl_2_Tool2.addWidget(self.lblFileType_Tool2)
self.comboBoxFileType_Tool2 = QtWidgets.QComboBox(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxFileType_Tool2.setFont(font)
self.comboBoxFileType_Tool2.setEditable(False)
self.comboBoxFileType_Tool2.setObjectName("comboBoxFileType_Tool2")
self.comboBoxFileType_Tool2.addItem("")
self.comboBoxFileType_Tool2.addItem("")
self.hl_2_Tool2.addWidget(self.comboBoxFileType_Tool2)
self.verticalLayout_3.addLayout(self.hl_2_Tool2)
self.hl_3_Tool2 = QtWidgets.QHBoxLayout()
self.hl_3_Tool2.setObjectName("hl_3_Tool2")
self.lblRetrain_Tool2 = QtWidgets.QLabel(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.lblRetrain_Tool2.setFont(font)
self.lblRetrain_Tool2.setObjectName("lblRetrain_Tool2")
self.hl_3_Tool2.addWidget(self.lblRetrain_Tool2)
self.comboBoxRetrain_Tool2 = QtWidgets.QComboBox(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxRetrain_Tool2.setFont(font)
self.comboBoxRetrain_Tool2.setObjectName("comboBoxRetrain_Tool2")
self.comboBoxRetrain_Tool2.addItem("")
self.comboBoxRetrain_Tool2.addItem("")
self.hl_3_Tool2.addWidget(self.comboBoxRetrain_Tool2)
self.verticalLayout_3.addLayout(self.hl_3_Tool2)
self.hl_4_Tool2 = QtWidgets.QHBoxLayout()
self.hl_4_Tool2.setObjectName("hl_4_Tool2")
self.lblDelete_Tool2 = QtWidgets.QLabel(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.lblDelete_Tool2.setFont(font)
self.lblDelete_Tool2.setObjectName("lblDelete_Tool2")
self.hl_4_Tool2.addWidget(self.lblDelete_Tool2)
self.comboBoxDelete_Tool2 = QtWidgets.QComboBox(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxDelete_Tool2.setFont(font)
self.comboBoxDelete_Tool2.setObjectName("comboBoxDelete_Tool2")
self.comboBoxDelete_Tool2.addItem("")
self.comboBoxDelete_Tool2.addItem("")
self.hl_4_Tool2.addWidget(self.comboBoxDelete_Tool2)
self.verticalLayout_3.addLayout(self.hl_4_Tool2)
self.btnStart_Tool2 = QtWidgets.QPushButton(self.Tool2)
font = QtGui.QFont()
font.setPointSize(10)
self.btnStart_Tool2.setFont(font)
self.btnStart_Tool2.setObjectName("btnStart_Tool2")

```

```

self.verticalLayout_3.addWidget(self.btnStart_Tool2)
self.textBrowser_Tool2 = QtWidgets.QTextBrowser(self.Tool2)
self.textBrowser_Tool2.setObjectName("textBrowser_Tool2")
self.verticalLayout_3.addWidget(self.textBrowser_Tool2)
self.tabWidget.addTab(self.Tool2, "")
self.Tool3 = QtWidgets.QWidget()
self.Tool3.setAutoFillBackground(True)
self.Tool3.setObjectName("Tool3")
self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.Tool3)
self.verticalLayout_2.setObjectName("verticalLayout_2")
self.hl_1_Tool3 = QtWidgets.QHBoxLayout()
self.hl_1_Tool3.setObjectName("hl_1_Tool3")
self.lblPath_Tool3 = QtWidgets.QLabel(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.lblPath_Tool3.setFont(font)
self.lblPath_Tool3.setObjectName("lblPath_Tool3")
self.hl_1_Tool3.addWidget(self.lblPath_Tool3)
self.txtPath_Tool3 = QtWidgets.QLineEdit(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.txtPath_Tool3.setFont(font)
self.txtPath_Tool3.setObjectName("txtPath_Tool3")
self.hl_1_Tool3.addWidget(self.txtPath_Tool3)
self.btnBrowse_Tool3 = QtWidgets.QPushButton(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.btnBrowse_Tool3.setFont(font)
self.btnBrowse_Tool3.setObjectName("btnBrowse_Tool3")
self.hl_1_Tool3.addWidget(self.btnBrowse_Tool3)
self.verticalLayout_2.addLayout(self.hl_1_Tool3)
self.hl_2_Tool3 = QtWidgets.QHBoxLayout()
self.hl_2_Tool3.setObjectName("hl_2_Tool3")
self.lblFileType_Tool3 = QtWidgets.QLabel(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.lblFileType_Tool3.setFont(font)
self.lblFileType_Tool3.setObjectName("lblFileType_Tool3")
self.hl_2_Tool3.addWidget(self.lblFileType_Tool3)
self.comboBoxFileType_Tool3 = QtWidgets.QComboBox(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxFileType_Tool3.setFont(font)
self.comboBoxFileType_Tool3.setEditable(False)
self.comboBoxFileType_Tool3.setObjectName("comboBoxFileType_Tool3")
self.comboBoxFileType_Tool3.addItem("")
self.comboBoxFileType_Tool3.addItem("")
self.hl_2_Tool3.addWidget(self.comboBoxFileType_Tool3)
self.verticalLayout_2.addLayout(self.hl_2_Tool3)
self.hl_3_Tool3 = QtWidgets.QHBoxLayout()
self.hl_3_Tool3.setObjectName("hl_3_Tool3")
self.lblDelete_Tool3 = QtWidgets.QLabel(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.lblDelete_Tool3.setFont(font)
self.lblDelete_Tool3.setObjectName("lblDelete_Tool3")
self.hl_3_Tool3.addWidget(self.lblDelete_Tool3)
self.comboBoxDelete_Tool3 = QtWidgets.QComboBox(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.comboBoxDelete_Tool3.setFont(font)
self.comboBoxDelete_Tool3.setObjectName("comboBoxDelete_Tool3")
self.comboBoxDelete_Tool3.addItem("")
self.comboBoxDelete_Tool3.addItem("")
self.hl_3_Tool3.addWidget(self.comboBoxDelete_Tool3)

```

```

self.verticalLayout_2.addLayout(self.hl_3_Tool3)
self.btnStart_Tool3 = QtWidgets.QPushButton(self.Tool3)
font = QtGui.QFont()
font.setPointSize(10)
self.btnStart_Tool3.setFont(font)
self.btnStart_Tool3.setObjectName("btnStart_Tool3")
self.verticalLayout_2.addWidget(self.btnStart_Tool3)
self.textBrowser_Tool3 = QtWidgets.QTextBrowser(self.Tool3)
# self.textBrowser_Tool3.ensureCursorVisible()
self.textBrowser_Tool3.setEnabled(True)
self.textBrowser_Tool3.setAutoFillBackground(False)
self.textBrowser_Tool3.setObjectName("textBrowser_Tool3")
self.verticalLayout_2.addWidget(self.textBrowser_Tool3)
self.tabWidget.addTab(self.Tool3, "")
self.verticalLayout_4.addWidget(self.tabWidget)

self.retranslateUi(Form)
self.tabWidget.setCurrentIndex(0)
QtCore.QMetaObject.connectSlotsByName(Form)

```

```

#####
#####

```

```

# Code for IDS

```

```

#####
#####

```

```

self.rootFolder = "NetIDS"

self.pcapsFolder = self.rootFolder + "/pcaps"

self.featuresFolder = self.rootFolder + "/features"

self.modelFolder = self.rootFolder + "/model"

self.modelFile = self.modelFolder + "/iForestModel.pkl"

self.resultsFolder = self.rootFolder + "/results"

self.redirectFolder = self.rootFolder + "/redirect"

self.redirectFile = self.redirectFolder + "/output.txt"

```

```

#####
#####

```

```

#####
#####

```

```

# Tool 1

```

```

#####
#####

```

```

# disables the text browser by default until the button Start is clicked
self.textBrowser_Tool1.setEnabled(False)

# makes the text browser read only mode
self.textBrowser_Tool1.setReadOnly(True)

# the output for Tool1 that is going to be written in the text browser
self.outputTool1 = ""

```

```

# disables the comboBoxSinglePcap_Tool1 by default
self.comboBoxSinglePcap_Tool1.setCurrentIndex(1)
self.comboBoxSinglePcap_Tool1.setEnabled(False)

# disables the comboBoxDelete_Tool1 by default
self.comboBoxDelete_Tool1.setCurrentIndex(1)
self.comboBoxDelete_Tool1.setEnabled(False)

# connecting the combo box Infinitely with the appropriate function
self.comboBoxInfinitely_Tool1.currentIndexChanged.connect(self.clickedInfinitely_Tool1)

# connecting the combo box Duration with the appropriate function
self.txtDuration_Tool1.textChanged.connect(self.durationChanged_Tool1)

# connecting the Packet number textbox with the appropriate function
self.txtPacketNum_Tool1.textChanged.connect(self.packetNumChanged_Tool1)

# connecting the combo box Continue with the appropriate function
self.comboBoxContinue_Tool1.currentIndexChanged.connect(self.clickedContinue_Tool1)

# connecting the Start button with the appropriate function
self.btnStart_Tool1.clicked.connect(self.start_Tool1)

#####

#####

#####

# Tool 2

#####

#####

# disables the text browser by default until the button Start is clicked
self.textBrowser_Tool2.setEnabled(False)

# makes the text browser read only mode
self.textBrowser_Tool2.setReadOnly(True)

# the output for Tool2 that is going to be written in the text browser
self.outputTool2 = ""

# disables the start button by default until the file is selected
self.btnStart_Tool2.setEnabled(False)

# connecting the Browse button with the appropriate function
self.btnBrowse_Tool2.clicked.connect(self.browse_Tool2)

# connecting the Path textbox with the appropriate function
self.txtPath_Tool2.textChanged.connect(self.pathChanged_Tool2)

# connecting the combo box File Type with the appropriate function
self.comboBoxFileType_Tool2.currentIndexChanged.connect(self.clickedFileType_Tool2)

# connecting the Start button with the appropriate function
self.btnStart_Tool2.clicked.connect(self.start_Tool2)

#####

#####

#####

```

```
#####
# Tool 3

#####

# disables the text browser by default until the button Start is clicked
self.textBrowser_Tool3.setEnabled(False)

# makes the text browser read only mode
self.textBrowser_Tool3.setReadOnly(True)

# the output for Tool3 that is going to be written in the text browser
self.outputTool3 = ""

# disables the start button by default until the file is selected
self.btnStart_Tool3.setEnabled(False)

# connecting the Browse button with the appropriate function
self.btnBrowse_Tool3.clicked.connect(self.browse_Tool3)

# connecting the Path text box with the appropriate function
self.txtPath_Tool3.textChanged.connect(self.pathChanged_Tool3)

# connecting the combo box File Type with the appropriate function
self.comboBoxFileType_Tool3.currentIndexChanged.connect(self.clickedFileType_Tool3)

# connecting the Start button with the appropriate function
self.btnStart_Tool3.clicked.connect(self.start_Tool3)

#####

def retranslateUi(self, Form):
    _translate = QtCore.QCoreApplication.translate
    Form.setWindowTitle(_translate("Form", "NetIDS"))
    self.lblInfinitely_Tool1.setText(_translate("Form", "Run Infinitely"))
    self.comboBoxInfinitely_Tool1.setCurrentText(_translate("Form", "No"))
    self.comboBoxInfinitely_Tool1.setItemText(0, _translate("Form", "No"))
    self.comboBoxInfinitely_Tool1.setItemText(1, _translate("Form", "Yes"))
    self.lblDuration_Tool1.setText(_translate("Form", "Duration of Packet Sniffer (in minutes)"))
    self.lblSinglePcap_Tool1.setText(_translate("Form", "Single pcap file for captured traffic"))
    self.comboBoxSinglePcap_Tool1.setItemText(0, _translate("Form", "No"))
    self.comboBoxSinglePcap_Tool1.setItemText(1, _translate("Form", "Yes"))
    self.lblPacketNum_Tool1.setText(_translate("Form", "Number of Packets to be captured"))
    self.lblContinue_Tool1.setText(_translate("Form", "Continue with Anomaly Detection"))
    self.comboBoxContinue_Tool1.setItemText(0, _translate("Form", "No"))
    self.comboBoxContinue_Tool1.setItemText(1, _translate("Form", "Yes"))
    self.lblDelete_Tool1.setText(_translate("Form", "Delete the .csv file after the process"))
    self.comboBoxDelete_Tool1.setItemText(0, _translate("Form", "Yes"))
    self.comboBoxDelete_Tool1.setItemText(1, _translate("Form", "No"))
    self.btnStart_Tool1.setText(_translate("Form", "Start"))
    self.tabWidget.setTabText(self.tabWidget.indexOf(self.Tool1), _translate("Form", "Packet Sniffer"))
    self.lblPath_Tool2.setText(_translate("Form", "File Path"))
    self.btnBrowse_Tool2.setText(_translate("Form", "Browse File"))
    self.lblFileType_Tool2.setText(_translate("Form", "Input file type"))
    self.comboBoxFileType_Tool2.setCurrentText(_translate("Form", "pcap file \ "<filename>.pcap\""))
    self.comboBoxFileType_Tool2.setItemText(0, _translate("Form", "pcap file \ "<filename>.pcap\""))
    self.comboBoxFileType_Tool2.setItemText(1, _translate("Form", "csv file \ "<filename>.csv\""))
    self.lblRetrain_Tool2.setText(_translate("Form", "Retrain the existing model"))
    self.comboBoxRetrain_Tool2.setItemText(0, _translate("Form", "No"))
    self.comboBoxRetrain_Tool2.setItemText(1, _translate("Form", "Yes"))
    self.lblDelete_Tool2.setText(_translate("Form", "Delete the .csv file after the process"))
    self.comboBoxDelete_Tool2.setItemText(0, _translate("Form", "Yes"))
```

```

self.comboBoxDelete_Tool2.setItemText(1, _translate("Form", "No"))
self.btnStart_Tool2.setText(_translate("Form", "Start"))
self.tabWidget.setTabText(self.tabWidget.indexOf(self.Tool2), _translate("Form", "Train Model"))
self.lblPath_Tool3.setText(_translate("Form", "File Path"))
self.btnBrowse_Tool3.setText(_translate("Form", "Browse File"))
self.lblFileType_Tool3.setText(_translate("Form", "Input file type"))
self.comboBoxFileType_Tool3.setCurrentText(_translate("Form", "pcap file \ "<filename>.pcap\""))
self.comboBoxFileType_Tool3.setItemText(0, _translate("Form", "pcap file \ "<filename>.pcap\""))
self.comboBoxFileType_Tool3.setItemText(1, _translate("Form", "csv file \ "<filename>.csv\""))
self.lblDelete_Tool3.setText(_translate("Form", "Delete the .csv file after the process"))
self.comboBoxDelete_Tool3.setItemText(0, _translate("Form", "Yes"))
self.comboBoxDelete_Tool3.setItemText(1, _translate("Form", "No"))
self.btnStart_Tool3.setText(_translate("Form", "Start"))
self.tabWidget.setTabText(self.tabWidget.indexOf(self.Tool3), _translate("Form", "Anomaly
Detection"))

#####
#####
# Code for IDS

#####
#####

def create_folder(self, createFold):

    if not os.path.isdir(createFold):

        os.mkdir(createFold)

def print_redirect(self, tool, text):

    if (tool == 1):

        self.outputTool1 += text + "\n"

        self.textBrowser_Tool1.setPlainText(self.outputTool1)

self.textBrowser_Tool1.verticalScrollBar().setValue(self.textBrowser_Tool1.verticalScrollBar().maximum(
))

    elif (tool == 2):

        self.outputTool2 += text + "\n"

        self.textBrowser_Tool2.setPlainText(self.outputTool2)

self.textBrowser_Tool2.verticalScrollBar().setValue(self.textBrowser_Tool2.verticalScrollBar().maximum(
))

    elif (tool == 3):

        self.outputTool3 += text + "\n"

        self.textBrowser_Tool3.setPlainText(self.outputTool3)

self.textBrowser_Tool3.verticalScrollBar().setValue(self.textBrowser_Tool3.verticalScrollBar().maximum(
))

    QtCore.QCoreApplication.processEvents()

    print(text)

```

```

def valid_FExtraction(self, tool, fileName):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    self.print_redirect(tool, '\nOpening {}...'format(fileName))

    colsCSV = ["Packet", "MAC-Src1", "MAC-Src2", "MAC-Src3", "MAC-Src4", "MAC-Src5", "MAC-Src6",
               "MAC-Dst1", "MAC-Dst2", "MAC-Dst3", "MAC-Dst4", "MAC-Dst5", "MAC-Dst6", "LLC", "IPv",
               "IP-Src1", "IP-Src2", "IP-Src3", "IP-Src4", "IP-Dst1", "IP-Dst2", "IP-Dst3", "IP-Dst4", "IP-Tos",
               "IP-Len", "Protocol", "Port-Src", "Port-Dst",
               "wirelen", "caplen", "sec", "usec"]

    tmp1 = fileName.split("/")

    tmp2 = tmp1[len(tmp1) - 1]

    tmp3 = tmp2.replace(".pcap", "")

    resFile = self.featuresFolder + "/" + tmp3 + ".csv"

    packetsCSV = []

    resultingOutliers = pd.DataFrame(packetsCSV, columns=colsCSV)

    resultingOutliers.to_csv(resFile, index=False, mode='w', header=True)

    count = 0

    interesting_packet_count = 0

    for (pkt_data, pkt_metadata,) in RawPcapReader(fileName):

        count += 1

        col = []

        col.append(str(count)) # for Packet

        ether_pkt = Ether(pkt_data)

        txtMACsrc = str(ether_pkt.src)

        tMACsrc = txtMACsrc.split(":")

        # col.append(str(ether_pkt.src))

        col.append(str(int(tMACsrc[0], 16))) # for MAC-Src1
        col.append(str(int(tMACsrc[1], 16))) # for MAC-Src2
        col.append(str(int(tMACsrc[2], 16))) # for MAC-Src3
        col.append(str(int(tMACsrc[3], 16))) # for MAC-Src4
        col.append(str(int(tMACsrc[4], 16))) # for MAC-Src5
        col.append(str(int(tMACsrc[5], 16))) # for MAC-Src6

        txtMACdst = str(ether_pkt.dst)

        tMACdst = txtMACdst.split(":")

        # col.append(str(ether_pkt.dst))

        col.append(str(int(tMACdst[0], 16))) # for MAC-Dst1
        col.append(str(int(tMACdst[1], 16))) # for MAC-Dst2
        col.append(str(int(tMACdst[2], 16))) # for MAC-Dst3

```

```

col.append(str(int(tMACdst[3], 16))) # for MAC-Dst4
col.append(str(int(tMACdst[4], 16))) # for MAC-Dst5
col.append(str(int(tMACdst[5], 16))) # for MAC-Dst6

# LLC frames will have 'len' instead of 'type'.
if 'type' not in ether_pkt.fields:

    col.append("1") # for LLC

    col.append(str(ether_pkt.len)) # for IPv

else:

    col.append("0") # for LLC

    col.append(str(ether_pkt.type)) # for IPv

if (IP in ether_pkt):

    ip_pkt = ether_pkt[IP]

    txtIPsrc = str(ip_pkt.src)

    tIPsrc = txtIPsrc.split(".")

    txtIPdst = str(ip_pkt.dst)

    tIPdst = txtIPdst.split(".")

    col.append(tIPsrc[0]) # for IP-Src1
    col.append(tIPsrc[1]) # for IP-Src2
    col.append(tIPsrc[2]) # for IP-Src3
    col.append(tIPsrc[3]) # for IP-Src4

    col.append(tIPdst[0]) # for IP-Dst1
    col.append(tIPdst[1]) # for IP-Dst2
    col.append(tIPdst[2]) # for IP-Dst3
    col.append(tIPdst[3]) # for IP-Dst4

    col.append(str(ip_pkt.tos)) # for IP-Tos

    col.append(str(ip_pkt.len)) # for IP-Len

    col.append(str(ip_pkt.proto)) # for protocol

    if (TCP in ip_pkt):

        tcp_pkt = ip_pkt[TCP]

        col.append(str(tcp_pkt.sport)) # for Port-Src

        col.append(str(tcp_pkt.dport)) # for Port-Dst

    elif (TCPErr in ip_pkt):

        tcp_error_pkt = ip_pkt[TCPErr]

        col.append(str(tcp_error_pkt.sport)) # for Port-Src

        col.append(str(tcp_error_pkt.dport)) # for Port-Dst

    elif (UDP in ip_pkt):

        udp_pkt = ip_pkt[UDP]

```

```

col.append(str(udp_pkt.sport)) # for Port-Src

col.append(str(udp_pkt.dport)) # for Port-Dst

elif (UDPError in ip_pkt):

    udp_error_pkt = ip_pkt[UDPError]

    col.append(str(udp_error_pkt.sport)) # for Port-Src

    col.append(str(udp_error_pkt.dport)) # for Port-Dst

else:

    col.append("-1") # for Port-Src

    col.append("-1") # for Port-Dst

else:

    if (ARP in ether_pkt):

        arp_pkt = ether_pkt[ARP]

        txtIPsrc = str(arp_pkt.psrc)

        tIPsrc = txtIPsrc.split(".")

        txtIPdst = str(arp_pkt.pdst)

        tIPdst = txtIPdst.split(".")

        col.append(tIPsrc[0]) # for IP-Src1
        col.append(tIPsrc[1]) # for IP-Src2
        col.append(tIPsrc[2]) # for IP-Src3
        col.append(tIPsrc[3]) # for IP-Src4

        col.append(tIPdst[0]) # for IP-Dst1
        col.append(tIPdst[1]) # for IP-Dst2
        col.append(tIPdst[2]) # for IP-Dst3
        col.append(tIPdst[3]) # for IP-Dst4

        col.append("-1") # for IP-Tos

        col.append("-1") # for IP-Len

        col.append(str(arp_pkt.ptype)) # for protocol

        col.append("-1") # for Port-Src

        col.append("-1") # for Port-Dst

    else:

        col.append("-1") # for IP-Src1
        col.append("-1") # for IP-Src2
        col.append("-1") # for IP-Src3
        col.append("-1") # for IP-Src4

        col.append("-1") # for IP-Dst1
        col.append("-1") # for IP-Dst2
        col.append("-1") # for IP-Dst3
        col.append("-1") # for IP-Dst4

        col.append("-1") # for IP-Tos

```

```

        col.append("-1") # for IP-Len

        col.append("-1") # for protocol

        col.append("-1") # for Port-Src

        col.append("-1") # for Port-Dst

    col.append(str(pkt_metadata.wirelen)) # for wirelen

    col.append(str(pkt_metadata.caplen)) # for caplen

    col.append(str(pkt_metadata.sec)) # for sec

    col.append(str(pkt_metadata.usec)) # for usec

    interesting_packet_count += 1

    packetsCSV.append(col)

    if ((count % 20000) == 0):

        QtCore.QCoreApplication.processEvents()

    if ((count % 10000000) == 0):

        self.print_redirect(tool, str(count))

        resultingOutliers = pd.DataFrame(packetsCSV, columns=colsCSV)

        resultingOutliers.to_csv(resFile, index=False, mode='a', header=False)

        packetsCSV = []

    QtCore.QCoreApplication.processEvents()

    resultingOutliers = pd.DataFrame(data=packetsCSV, columns=colsCSV)

    resultingOutliers.to_csv(resFile, index=False, mode='a', header=False)

    self.print_redirect(tool, "The results have been stored in {} successfully!".format(resFile))

    self.print_redirect(tool, '{} contains {} packets ({} interesting)'.format(fileName, count,
interesting_packet_count))

    t1 = time.perf_counter() - t0

    self.print_redirect(tool, "Execution Time: " + str(datetime.timedelta(seconds=t1)))

    return True

def start_FExtraction(self, tool, file_name):

    QtCore.QCoreApplication.processEvents()

    self.create_folder(self.rootFolder)

    self.create_folder(self.featuresFolder)

    if (file_name is not None):

        if (len(file_name) < 6): # 5 chars for .pcap + 1 char more for an actual file with a name

            self.print_redirect(tool, "\n" + "{} is not valid. Only .pcap files are allowed.".format(file_name))

```

```

        return False

    if (file_name.endswith(".pcap") is False):

        self.print_redirect(tool, "\n" + "{}" is not valid. Only .pcap files are allowed.'.format(file_name))

        return False

    else:

        self.print_redirect(tool, "\nThe file parameter is mandatory.')

        return False

    if not os.path.isfile(file_name):

        self.print_redirect(tool, "\n" + "{}" does not exist'.format(file_name))

        return False

    if (self.valid_FExtraction(tool, file_name) is False):

        return False

    return True

def readDataSet(self, fileName):

    data = pd.read_csv(str(fileName), sep=',').values

    data = pd.DataFrame(data)

    data = data.astype({0: 'uint32'}) # Packet

    data = data.astype({1: 'uint8'}) # MAC-Src1
    data = data.astype({2: 'uint8'}) # MAC-Src2
    data = data.astype({3: 'uint8'}) # MAC-Src3
    data = data.astype({4: 'uint8'}) # MAC-Src4
    data = data.astype({5: 'uint8'}) # MAC-Src5
    data = data.astype({6: 'uint8'}) # MAC-Src6

    data = data.astype({7: 'uint8'}) # MAC-Dst1
    data = data.astype({8: 'uint8'}) # MAC-Dst2
    data = data.astype({9: 'uint8'}) # MAC-Dst3
    data = data.astype({10: 'uint8'}) # MAC-Dst4
    data = data.astype({11: 'uint8'}) # MAC-Dst5
    data = data.astype({12: 'uint8'}) # MAC-Dst6

    data = data.astype({13: 'uint8'}) # LLC

    data = data.astype({14: 'int16'}) # IPv

    data = data.astype({15: 'uint8'}) # IP-Src1
    data = data.astype({16: 'uint8'}) # IP-Src2
    data = data.astype({17: 'uint8'}) # IP-Src3
    data = data.astype({18: 'uint8'}) # IP-Src4

    data = data.astype({19: 'uint8'}) # IP-Dst1
    data = data.astype({20: 'uint8'}) # IP-Dst2
    data = data.astype({21: 'uint8'}) # IP-Dst3
    data = data.astype({22: 'uint8'}) # IP-Dst4

    data = data.astype({23: 'int16'}) # IP-Tos

```

```

data = data.astype({24: 'int16'}) # IP-Len

data = data.astype({25: 'int16'}) # Protocol

data = data.astype({26: 'int32'}) # Port-Src

data = data.astype({27: 'int32'}) # Port-Dst

data = data.astype({28: 'uint16'}) # wirelen

data = data.astype({29: 'uint16'}) # caplen

# data = data.astype({30: 'uint32'}) # sec

# data = data.astype({31: 'uint32'}) # usec

data = data.values

data = np.nan_to_num(data)

QtCore.QCoreApplication.processEvents()

return data

def valid_Train(self, tool, fileName, retOption):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    trainFile = fileName

    self.print_redirect(tool, '\nOpening {}'.format(trainFile))

    trainData = self.readDataSet(trainFile)

    # clf = Pipeline([("scale", StandardScaler()), ("clf", IsolationForest(contamination="auto",
random_state=0))])

    clf = IsolationForest(contamination="auto", random_state=0, warm_start=True)

    iForestModelFile = self.modelFile # the name of the model that is going to be stored

    if (retOption is True):

        # Start section: load model

        if not os.path.isfile(iForestModelFile):

            self.print_redirect(tool,
'\nThe stored (trained) model for Isolation Forest "{}" does not exist'.format(iForestModelFile))

            return False

        # Load from file
        clf = joblib.load(iForestModelFile)

        modelParams = clf.get_params()

        model_n_estimators = modelParams['n_estimators']

        model_n_estimators += 100 # 100 is the default value for n_estimators, so just add the default each
time

        clf.set_params(n_estimators=model_n_estimators)

```

```

        self.print_redirect(tool, "The model has been loaded successfully!")

# End section: load model

clf.fit(trainData)

# Start section: store model

# Save to file in the current working directory
joblib.dump(clf, iForestModelFile)

# End section: store model

self.print_redirect(tool, "The model has been created successfully!")

t1 = time.perf_counter() - t0

self.print_redirect(tool, "Execution Time: " + str(datetime.timedelta(seconds=t1)))

return True

def start_Train(self, tool, retargu, file_name):

    QtCore.QCoreApplication.processEvents()

    self.create_folder(self.rootFolder)

    self.create_folder(self.modelFolder)

    retoption = False

    if (retargu is not None):

        retarg = retargu.lower()

        if (retarg == "true"):

            retoption = True

        elif (retarg == "false"):

            retoption = False

        else:

            self.print_redirect(tool, "\n{}" no such option exists. You can only use true or
false'.format(retargu))

            return False

    if (file_name is not None):

        if (len(file_name) < 5): # 4 chars for .csv + 1 char more for an actual file with a name

            self.print_redirect(tool, "\n{}" is not valid. Only .csv files are allowed.'.format(file_name))

            return False

        if (file_name.endswith(".csv") is False):

            self.print_redirect(tool, "\n{}" is not valid. Only .csv files are allowed.'.format(file_name))

            return False

```

```

else:

    self.print_redirect(tool, '\nThe file parameter is mandatory.')

    return False

if not os.path.isfile(file_name):

    self.print_redirect(tool, '\n"{}" does not exist'.format(file_name))

    return False

if (self.valid_Train(tool, file_name, retoption) is False):

    return False

return True

def valid_PcapToTrain(self, tool, fileName, delOption, retOption):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    self.print_redirect(tool, "\nStarting the feature extraction process...")

    if (self.start_FExtraction(tool, fileName) is False):

        return False

    tmp1 = fileName.split("/")

    tmp2 = tmp1[len(tmp1) - 1]

    tmp3 = tmp2.replace(".pcap", "")

    resFile = self.featuresFolder + "/" + tmp3 + ".csv"

    self.print_redirect(tool, "\nStarting the Isolation Forest training process...")

    retrainModel = 'false'

    if (retOption is True):

        retrainModel = 'true'

    if (self.start_Train(tool, retrainModel, resFile) is False):

        return False

    if (delOption is True):

        if os.path.exists(resFile):

            os.remove(resFile)

    t1 = time.perf_counter() - t0

    self.print_redirect(tool, "\nTotal Execution Time: " + str(datetime.timedelta(seconds=t1)))

    return True

def start_PcapToTrain(self, tool, delargu, retargu, file_name):

    QtCore.QCoreApplication.processEvents()

```

```

deloption = True

if (delargu is not None):
    delarg = delargu.lower()

    if (delarg == "true"):
        deloption = True

    elif (delarg == "false"):
        deloption = False

    else:
        self.print_redirect(tool, "\n{}" no such option exists. You can only use true or
false'.format(delargu))

    return False

retoption = False

if (retargu is not None):
    retarg = retargu.lower()

    if (retarg == "true"):
        retoption = True

    elif (retarg == "false"):
        retoption = False

    else:
        self.print_redirect(tool, "\n{}" no such option exists. You can only use true or
false'.format(retargu))

    return False

if (file_name is not None):
    if (len(file_name) < 6): # 5 chars for .pcap + 1 char more for an actual file with a name
        self.print_redirect(tool, "\n{}" is not valid. Only .pcap files are allowed.'.format(file_name))
        return False

    if (file_name.endswith(".pcap") is False):
        self.print_redirect(tool, "\n{}" is not valid. Only .pcap files are allowed.'.format(file_name))
        return False

    else:
        self.print_redirect(tool, "\nThe file parameter is mandatory.')

        return False

if not os.path.isfile(file_name):

```

```

        self.print_redirect(tool, '\n{"} does not exist'.format(file_name))

    return False

if (self.valid_PcapToTrain(tool, file_name, deloption, retoption) is False):

    return False

return True

def valid_Test(self, tool, fileName):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    testFile = fileName

    self.print_redirect(tool, '\nOpening {}...'.format(testFile))

    testData = self.readDataSet(testFile)

    # Start section: load model

    iForestModelFile = self.modelFile # the name of the model that is going to be loaded

    if not os.path.isfile(iForestModelFile):

        self.print_redirect(tool,
        '\nThe stored (trained) model for Isolation Forest "{}" does not exist'.format(iForestModelFile))

        return False

    # Load from file
    iForestModel = joblib.load(iForestModelFile)

    self.print_redirect(tool, "The model has been loaded successfully!")

    # End section: load model

    # Calculate the accuracy and predictions
    resPrediction = iForestModel.predict(testData)

    tmp1 = testFile.split("/")

    tmp2 = tmp1[len(tmp1) - 1]

    tmp3 = tmp2.replace(".csv", "")

    resFile = self.resultsFolder + "/" + tmp3 + "_results.csv"

    colsCSV = ["Packet Number", "Anomaly Score"]

    packetsCSV = []

    resultingOutliers = pd.DataFrame(packetsCSV, columns=colsCSV)

    resultingOutliers.to_csv(resFile, index=False, mode='w', header=True)

    count = 0

    malicious_count = 0

    for row in resPrediction:

```

```

count += 1

col = []

col.append(count)

rowResult = "Benign"

if (row == -1):
    malicious_count += 1

    rowResult = "Malicious"

col.append(rowResult)

packetsCSV.append(col)

if ((count % 20000) == 0):

    QtCore.QCoreApplication.processEvents()

    if ((count % 10000000) == 0):

        self.print_redirect(tool, str(count))

        resultingOutliers = pd.DataFrame(packetsCSV, columns=colsCSV)

        resultingOutliers.to_csv(resFile, index=False, mode='a', header=False)

        packetsCSV = []

    resultingOutliers = pd.DataFrame(packetsCSV, columns=colsCSV)

    resultingOutliers.to_csv(resFile, index=False, mode='a', header=False)

    self.print_redirect(tool, "The results have been stored in {} successfully!".format(resFile))

    self.print_redirect(tool, '{} contains {} packets ({} malicious)'.format(testFile, count, malicious_count))

t1 = time.perf_counter() - t0

self.print_redirect(tool, "Execution Time: " + str(datetime.timedelta(seconds=t1)))

return True

def start_Test(self, tool, file_name):

    QtCore.QCoreApplication.processEvents()

    self.create_folder(self.rootFolder)

    self.create_folder(self.resultsFolder)

    if (file_name is not None):

        if (len(file_name) < 5): # 4 chars for .csv + 1 char more for an actual file with a name

            self.print_redirect(tool, "{} is not valid. Only .csv files are allowed.".format(file_name))

            return False

        if (file_name.endswith(".csv") is False):

            self.print_redirect(tool, "{} is not valid. Only .csv files are allowed.".format(file_name))

```

```

        return False

    else:

        self.print_redirect(tool, '\nThe file parameter is mandatory.')

        return False

    if not os.path.isfile(file_name):

        self.print_redirect(tool, '\n"{}" does not exist'.format(file_name))

        return False

    if (self.valid_Test(tool, file_name) is False):

        return False

    return True

def valid_PcapToTest(self, tool, fileName, delOption):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    self.print_redirect(tool, "\nStarting the feature extraction process...")

    if (self.start_FExtraction(tool, fileName) is False):

        return False

    tmp1 = fileName.split("/")

    tmp2 = tmp1[len(tmp1) - 1]

    tmp3 = tmp2.replace(".pcap", "")

    resFile = self.featuresFolder + "/" + tmp3 + ".csv"

    self.print_redirect(tool, "\nStarting the Isolation Forest testing process...")

    if (self.start_Test(tool, resFile) is False):

        return False

    if (delOption is True):

        if os.path.exists(resFile):

            os.remove(resFile)

    t1 = time.perf_counter() - t0

    self.print_redirect(tool, "\nTotal Execution Time: " + str(datetime.timedelta(seconds=t1)))

    return True

def start_PcapToTest(self, tool, delargu, file_name):

    QtCore.QCoreApplication.processEvents()

    deloption = True

    if (delargu is not None):

```

```

delarg = delargu.lower()

if (delarg == "true"):
    deloption = True
elif (delarg == "false"):
    deloption = False
else:
    self.print_redirect(tool, "\n{}" no such option exists. You can only use true or
false'.format(delargu))

    return False

if (file_name is not None):
    if (len(file_name) < 6): # 5 chars for .pcap + 1 char more for an actual file with a name
        self.print_redirect(tool, "\n{}" is not valid. Only .pcap files are allowed.'.format(file_name))
        return False

    if (file_name.endswith(".pcap") is False):
        self.print_redirect(tool, "\n{}" is not valid. Only .pcap files are allowed.'.format(file_name))
        return False

    else:
        self.print_redirect(tool, "\nThe file parameter is mandatory.')
        return False

    if not os.path.isfile(file_name):
        self.print_redirect(tool, "\n{}" does not exist'.format(file_name))
        return False

    if (self.valid_PcapToTest(tool, file_name, deloption) is False):
        return False

    return True

def valid_Sniffer_internal(self, tool, numPackets, contOption, delOption, storeOption, oneFile):
    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    self.print_redirect(tool, "\nCapturing network traffic...')

    curTime = str(datetime.datetime.now())

    tmp1 = curTime.split(".")
    tmp2 = tmp1[0].replace(":", "-")
    tmp3 = tmp2.replace(" ", "-")

```

```

resFile = self.pcapsFolder + "/" + tmp3 + ".pcap"

if (storeOption is True):

    resFile = oneFile

# allInterfaces = show_interfaces()

# print(conf.iface)

textFilter = "ip host " + conf.iface.ip

sniffedPackets = 0

sniffPackets = 20000

remPackets = numPackets - sniffedPackets

while (remPackets > 0):

    QtCore.QCoreApplication.processEvents()

    if (remPackets < sniffPackets):

        sniffPackets = remPackets

    pkts = sniff(filter=textFilter, iface=conf.iface, count=sniffPackets)

    wrpcap(resFile, pkts, append=True)

    sniffedPackets += sniffPackets

    remPackets = numPackets - sniffedPackets

self.print_redirect(tool, "The results have been stored in {} successfully!".format(resFile))

t1 = time.perf_counter() - t0

self.print_redirect(tool, "Execution Time: " + str(datetime.timedelta(seconds=t1)))

if (storeOption is False and contOption is True):

    deleteFeatures = 'false'

    if (delOption is True):

        deleteFeatures = 'true'

    if (self.start_PcapToTest(tool, deleteFeatures, resFile) is False):

        return False

    return True

def valid_Sniffer(self, tool, tMinutes, numPackets, contOption, delOption, storeOption):

    QtCore.QCoreApplication.processEvents()

    t0 = time.perf_counter()

    self.print_redirect(tool, '\nStarting the sniffing process...')

    curTime = str(datetime.datetime.now())

```

```

tmp1 = curTime.split(".")
tmp2 = tmp1[0].replace(":", "-")
tmp3 = tmp2.replace(" ", "-")

onePCAPFile = self.pcapsFolder + "/" + tmp3 + ".pcap"

if (tMinutes == -1): # Just once

    if (self.valid_Sniffer_internal(tool, numPackets, contOption, delOption, storeOption, onePCAPFile) is
False):

        return False

    if (storeOption is True and contOption is True):

        deleteFeatures = 'false'

        if (delOption is True):

            deleteFeatures = 'true'

        if (self.start_PcapToTest(tool, deleteFeatures, onePCAPFile) is False):

            return False

elif (tMinutes == -200):

    starTime = datetime.datetime.now()

    while True:

        QtCore.QCoreApplication.processEvents()

        if (self.valid_Sniffer_internal(tool, numPackets, contOption, delOption, storeOption, onePCAPFile) is
False):

            return False

        runTime = datetime.datetime.now() - starTime

        self.print_redirect(tool, "Running for: " + str(runTime))

else:

    endTime = datetime.datetime.now() + datetime.timedelta(minutes=tMinutes)

    while True:

        QtCore.QCoreApplication.processEvents()

        if datetime.datetime.now() >= endTime:

            break

        if (self.valid_Sniffer_internal(tool, numPackets, contOption, delOption, storeOption, onePCAPFile) is
False):

            return False

        tmpTime = datetime.datetime.now()

        if (endTime >= tmpTime):

```

```

        remTime = endTime - tmpTime

        self.print_redirect(tool, "Remaining: " + str(remTime))

    else:

        self.print_redirect(tool, "No Time Remaining")

    if (storeOption is True and contOption is True):

        deleteFeatures = 'false'

        if (delOption is True):

            deleteFeatures = 'true'

        if (self.start_PcapToTest(tool, deleteFeatures, onePCAPFile) is False):

            return False

    t1 = time.perf_counter() - t0

    self.print_redirect(tool, "\nTotal Execution Time: " + str(datetime.timedelta(seconds=t1)))

    return True

def start_Sniffer(self, tool, infargu, timeargu, storeargu, packargu, contargu, delargu):

    QtCore.QCoreApplication.processEvents()

    self.create_folder(self.rootFolder)

    self.create_folder(self.pcapsFolder)

    # infargu = args.i

    infoption = False

    if (infargu is not None):

        infarg = infargu.lower()

        if (infarg == "true"):

            infoption = True

        elif (infarg == "false"):

            infoption = False

        else:

            self.print_redirect(tool, "\n" + "{}" no such option exists. You can only use true or
false'.format(infargu))

            return False

    timeMinutes = -1

    # timeargu = args.t

    if (timeargu is not None):

        timearg = timeargu.isdecimal()

```

```

if (timearg): # it means true

    timeoption = int(timeargu)

    if (timeoption > 0):

        timeMinutes = timeoption

    else:

        self.print_redirect(tool,
            '\n' + "{} no negative numbers are allowed. You can only use a positive integer'.format(timeargu))

        return False

    else:

        self.print_redirect(tool, '\n' + "{} no such option exists. You can only use a positive
integer'.format(timeargu))

        return False

# storeargu = args.s

storeoption = False

if (storeargu is not None):

    storearg = storeargu.lower()

    if (storearg == "true"):

        storeoption = True

    elif (storearg == "false"):

        storeoption = False

    else:

        self.print_redirect(tool, '\n' + "{} no such option exists. You can only use true or
false'.format(storeargu))

        return False

packetNum = 200

# packargu = args.p

if (packargu is not None):

    packarg = packargu.isdecimal()

    if (packarg): # it means true

        packoption = int(packargu)

        if (packoption > 0):

            packetNum = packoption

        else:

            self.print_redirect(tool,
                '\n' + "{} no negative numbers are allowed. You can only use a positive integer'.format(packargu))

```

```

        return False

    else:

        self.print_redirect(tool, "\n" + "{}" no such option exists. You can only use a positive
integer'.format(packargu))

        return False

# contargu = args.c

contoption = False

if (contargu is not None):

    contarg = contargu.lower()

    if (contarg == "true"):

        contoption = True

    elif (contarg == "false"):

        contoption = False

    else:

        self.print_redirect(tool, "\n" + "{}" no such option exists. You can only use true or
false'.format(contargu))

        return False

# delargu = args.d

deloption = True

if (delargu is not None):

    delarg = delargu.lower()

    if (delarg == "true"):

        deloption = True

    elif (delarg == "false"):

        deloption = False

    else:

        self.print_redirect(tool, "\n" + "{}" no such option exists. You can only use true or
false'.format(delargu))

        return False

if (infoption is True):

    timeMinutes = -200

    if (storeoption is True):

        self.print_redirect(tool, "\nRunning infinitely and storing the captured traffic in a single pcap file is
forbidden')

```

```

        return False

    if (self.valid_Sniffer(tool, timeMinutes, packetNum, contoption, deloption, storeoption) is False):

        return False

    return True

#####

#####

#####
# Tool 1
#####

#####

# this function disables and enables all the available options based on the Infinitely option
def clickedInfinitely_Tool1(self):

    infinityOption = self.comboBoxInfinitely_Tool1.currentText()

    if (infinityOption == "Yes"):

        self.txtDuration_Tool1.setEnabled(False)

        self.comboBoxSinglePcap_Tool1.setCurrentIndex(0)

        self.comboBoxSinglePcap_Tool1.setEnabled(False)

        self.txtDuration_Tool1.setText("")

    elif (infinityOption == "No"):

        self.txtDuration_Tool1.setEnabled(True)

        self.comboBoxSinglePcap_Tool1.setCurrentIndex(1)

        self.comboBoxSinglePcap_Tool1.setEnabled(True)

        self.durationChanged_Tool1()

    else:

        self.showMessage("Something went wrong with the File Type Options")

# this function disables and enables all the available options based on the Duration option
def durationChanged_Tool1(self):

    duration = self.txtDuration_Tool1.text()

    if ((duration is not None) and (len(duration) > 0)):

        if ((duration.isdecimal()) and (int(duration) > 0)):

            self.comboBoxSinglePcap_Tool1.setEnabled(True)

            self.comboBoxSinglePcap_Tool1.setCurrentIndex(0)

            self.btnStart_Tool1.setEnabled(True)

        else:

```

```

        self.btnStart_Tool1.setEnabled(False)

        self.showMessage("Only positive integers are allowed for the duration option")

        self.txtDuration_Tool1.setText("")

    else:

        self.comboBoxSinglePcap_Tool1.setEnabled(False)

        self.comboBoxSinglePcap_Tool1.setCurrentIndex(1)

        self.btnStart_Tool1.setEnabled(True)

# this function disables and enables all the available options based on the Nunmber of Packets option
def packetNumChanged_Tool1(self):

    packetNum = self.txtPacketNum_Tool1.text()

    if ((packetNum is not None) and (len(packetNum) > 0)):

        if ((packetNum.isdecimal()) and (int(packetNum) > 0)):

            self.btnStart_Tool1.setEnabled(True)

        else:

            self.btnStart_Tool1.setEnabled(False)

            self.showMessage("Only positive integers are allowed for the number of packets option")

            self.txtPacketNum_Tool1.setText("")

    else:

        self.btnStart_Tool1.setEnabled(True)

# this function disables and enables all the available options based on the Continue option
def clickedContinue_Tool1(self):

    continueOption = self.comboBoxContinue_Tool1.currentText()

    if (continueOption == "Yes"):

        self.comboBoxDelete_Tool1.setEnabled(True)

        self.comboBoxDelete_Tool1.setCurrentIndex(0)

    elif (continueOption == "No"):

        self.comboBoxDelete_Tool1.setCurrentIndex(1)

        self.comboBoxDelete_Tool1.setEnabled(False)

    else:

        self.showMessage("Something went wrong with the File Type Options")

# this function calls the IDS functions based on the options provided
def start_Tool1(self):

    enabledInfinitely = False

    enabledDuration = False

```

```

enabledSinglePcap = False

enabledPacketNum = False

enabledContinue = False

enabledDelete = False

if (self.comboBoxInfinitely_Tool1.isEnabledTo(self.Tool1)):

    enabledInfinitely = True

if (self.txtDuration_Tool1.isEnabledTo(self.Tool1)):

    enabledDuration = True

if (self.comboBoxSinglePcap_Tool1.isEnabledTo(self.Tool1)):

    enabledSinglePcap = True

if (self.txtPacketNum_Tool1.isEnabledTo(self.Tool1)):

    enabledPacketNum = True

if (self.comboBoxContinue_Tool1.isEnabledTo(self.Tool1)):

    enabledContinue = True

if (self.comboBoxDelete_Tool1.isEnabledTo(self.Tool1)):

    enabledDelete = True

self.comboBoxInfinitely_Tool1.setEnabled(False)

self.txtDuration_Tool1.setEnabled(False)

self.comboBoxSinglePcap_Tool1.setEnabled(False)

self.txtPacketNum_Tool1.setEnabled(False)

self.comboBoxContinue_Tool1.setEnabled(False)

self.comboBoxDelete_Tool1.setEnabled(False)

self.btnStart_Tool1.setEnabled(False)

self.textBrowser_Tool1.setEnabled(True)

infinityArg = self.comboBoxInfinitely_Tool1.currentText()

durationArg = self.txtDuration_Tool1.text()

singlePcapArg = self.comboBoxSinglePcap_Tool1.currentText()

packetNumArg = self.txtPacketNum_Tool1.text()

continueOptionArg = self.comboBoxContinue_Tool1.currentText()

delOptionArg = self.comboBoxDelete_Tool1.currentText()

infinityBool = "false"

durArg = None

singlePcapBool = "false"

```

```

packArg = None

continueOptionBool = "false"

delOptionBool = "true"

if (infinityArg == "Yes"):
    infinityBool = "true"

if (durationArg == ""):
    durArg = None
else:
    durArg = durationArg

if (signlePcapArg == "Yes"):
    signlePcapBool = "true"

if (packetNumArg == ""):
    packArg = None
else:
    packArg = packetNumArg

if (continueOptionArg == "Yes"):
    continueOptionBool = "true"

if (delOptionArg == "No"):
    delOptionBool = "false"

tool = 1

# call the functions here

QtCore.QCoreApplication.processEvents()

# function call

self.start_Sniffer(tool, infinityBool, durArg, signlePcapBool, packArg, continueOptionBool,
delOptionBool)

QtCore.QCoreApplication.processEvents()

if (enabledInfinitely):
    self.comboBoxInfinitely_Tool1.setEnabled(True)

if (enabledDuration):
    self.txtDuration_Tool1.setEnabled(True)

if (enabledSinglePcap):
    self.comboBoxSinglePcap_Tool1.setEnabled(True)

if (enabledPacketNum):

```

```

        self.txtPacketNum_Tool1.setEnabled(True)

    if (enabledContinue):

        self.comboBoxContinue_Tool1.setEnabled(True)

    if (enabledDelete):

        self.comboBoxDelete_Tool1.setEnabled(True)

    self.btnStart_Tool1.setEnabled(True)

    # self.textBrowser_Tool1.setEnabled(False)

#####

#####

#####

#####
    # Tool 2

#####

#####

# this function browses the folders to open a file
def browse_Tool2(self):

    # empties the path textbox
    self.txtPath_Tool2.setText("")

    # disables the text browser by default until the button Start is clicked
    self.textBrowser_Tool2.setEnabled(False)

    # disables the start button by default until the file is selected
    self.btnStart_Tool2.setEnabled(False)

    fileType = self.comboBoxFileType_Tool2.currentText()

    fileType1 = fileType.split(".")

    inputFileType = "Input File (*.pcap)"

    if (fileType1[1].startswith("pcap")):

        inputFileType = "Input File (*.pcap)"

    elif (fileType1[1].startswith("csv")):

        inputFileType = "Input File (*.csv)"

    else:

        self.showMessage("Something went wrong with the File Type Options")

    fileName, _ = QtWidgets.QFileDialog.getOpenFileName(None, "Select Input File", "", inputFileType)

    if fileName:

        self.txtPath_Tool2.setText(fileName)

        fn = fileName.split("/")

```

```

fn1 = fn[len(fn) - 1].split(".")

fn2 = fn1[len(fn1) - 1]

if len(fn1) < 2 or (len(fn1) > 1 and not (("pcap" in fn2) or ("csv" in fn2))):

    self.showMessage("The file you have provided is not acceptable")

    return

elif not os.path.exists(fileName):

    self.showMessage("The file you have provided does not exist")

    return

self.btnStart_Tool2.setEnabled(True)

# this function disables and enables all the available options based on the validity of the given file
def pathChanged_Tool2(self):

    # disables the text browser by default until the button Start is clicked
    self.textBrowser_Tool2.setEnabled(False)

    # disables the start button by default until the file is selected
    self.btnStart_Tool2.setEnabled(False)

    fileName = self.txtPath_Tool2.text()

    fileType = self.comboBoxFileType_Tool2.currentText()

    fileType1 = fileType.split(".")

    if (fileType1[1].startswith("pcap")):

        if ((fileName is not None) and (len(fileName) > 0)):

            if ((fileName.endswith(".pcap")) and (len(fileName) > len(".pcap"))):

                self.btnStart_Tool2.setEnabled(True)

    elif (fileType1[1].startswith("csv")):

        if ((fileName is not None) and (len(fileName) > 0)):

            if ((fileName.endswith(".csv")) and (len(fileName) > len(".csv"))):

                self.btnStart_Tool2.setEnabled(True)

    else:

        self.showMessage("Something went wrong with the File Type Options")

# this function disables and enables all the available options based on the File Type option
def clickedFileType_Tool2(self):

    fileName = self.txtPath_Tool2.text()

    fileType = self.comboBoxFileType_Tool2.currentText()

    fileType1 = fileType.split(".")

    if (fileType1[1].startswith("pcap")):

        if ((fileName is not None) and (len(fileName) > 0)):

```

```

if ((fileName.endswith(".pcap")) and (len(fileName) > len(".pcap"))):

    self.btnStart_Tool2.setEnabled(True)

else:

    # disables the text browser by default until the button Start is clicked
    self.textBrowser_Tool2.setEnabled(False)

    # disables the start button by default until the file is selected
    self.btnStart_Tool2.setEnabled(False)

self.comboBoxDelete_Tool2.setEnabled(True)

self.comboBoxDelete_Tool2.setCurrentIndex(0)

elif (fileType1[1].startswith("csv")):

    if ((fileName is not None) and (len(fileName) > 0)):

        if ((fileName.endswith(".csv")) and (len(fileName) > len(".csv"))):

            self.btnStart_Tool2.setEnabled(True)

        else:

            # disables the text browser by default until the button Start is clicked
            self.textBrowser_Tool2.setEnabled(False)

            # disables the start button by default until the file is selected
            self.btnStart_Tool2.setEnabled(False)

        self.comboBoxDelete_Tool2.setCurrentIndex(1)

        self.comboBoxDelete_Tool2.setEnabled(False)

    else:

        self.showMessage("Something went wrong with the File Type Options")

# this function, calls the IDS functions based on the options provided
def start_Tool2(self):

    enabledPath = False

    enabledBrowse = False

    enabledFileType = False

    enabledRetrain = False

    enabledDelete = False

    if (self.txtPath_Tool2.isEnabledTo(self.Tool2)):

        enabledPath = True

    if (self.btnBrowse_Tool2.isEnabledTo(self.Tool2)):

        enabledBrowse = True

    if (self.comboBoxFileType_Tool2.isEnabledTo(self.Tool2)):

        enabledFileType = True

```

```

if (self.comboBoxRetrain_Tool2.isEnabledTo(self.Tool2)):
    enabledRetrain = True

if (self.comboBoxDelete_Tool2.isEnabledTo(self.Tool2)):
    enabledDelete = True

self.txtPath_Tool2.setEnabled(False)
self.btnBrowse_Tool2.setEnabled(False)
self.comboBoxFileType_Tool2.setEnabled(False)
self.comboBoxRetrain_Tool2.setEnabled(False)
self.comboBoxDelete_Tool2.setEnabled(False)
self.btnStart_Tool2.setEnabled(False)
self.textBrowser_Tool2.setEnabled(True)

fileNameArg = self.txtPath_Tool2.text()
fileTypeArg = self.comboBoxFileType_Tool2.currentText()
retrainOptionArg = self.comboBoxRetrain_Tool2.currentText()
delOptionArg = self.comboBoxDelete_Tool2.currentText()

retrainOptionBool = "false"
delOptionBool = "true"

if (retrainOptionArg == "Yes"):
    retrainOptionBool = "true"

if (delOptionArg == "No"):
    delOptionBool = "false"

tool = 2

# call the functions here

QtCore.QCoreApplication.processEvents()

# function call
if (fileTypeArg == 'pcap file "<filename>.pcap"'):
    if ((fileNameArg is not None) and (len(fileNameArg) > 0)):
        if ((fileNameArg.endswith(".pcap")) and (len(fileNameArg) > len(".pcap"))):
            self.start_PcapToTrain(tool, delOptionBool, retrainOptionBool, fileNameArg)
elif (fileTypeArg == 'csv file "<filename>.csv"'):
    if ((fileNameArg is not None) and (len(fileNameArg) > 0)):
        if ((fileNameArg.endswith(".csv")) and (len(fileNameArg) > len(".csv"))):
            self.start_Train(tool, retrainOptionBool, fileNameArg)

```

```

QtCore.QCoreApplication.processEvents()

if (enabledPath):

    self.txtPath_Tool2.setEnabled(True)

if (enabledBrowse):

    self.btnBrowse_Tool2.setEnabled(True)

if (enabledFileType):

    self.comboBoxFileType_Tool2.setEnabled(True)

if (enabledRetrain):

    self.comboBoxRetrain_Tool2.setEnabled(True)

if (enabledDelete):

    self.comboBoxDelete_Tool2.setEnabled(True)

self.btnStart_Tool2.setEnabled(True)

#####

#####

#####

# Tool 3

#####

#####

# this function browses the folders to open a file
def browse_Tool3(self):

    # empties the path textbox
    self.txtPath_Tool3.setText("")

    # disables the text browser by default until the button Start is clicked
    self.textBrowser_Tool3.setEnabled(False)

    # disables the start button by default until the file is selected
    self.btnStart_Tool3.setEnabled(False)

    fileType = self.comboBoxFileType_Tool3.currentText()

    fileType1 = fileType.split(".")

    inputFileType = "Input File (*.pcap)"

    if (fileType1[1].startswith("pcap")):

        inputFileType = "Input File (*.pcap)"

    elif (fileType1[1].startswith("csv")):

        inputFileType = "Input File (*.csv)"

    else:

        self.showMessage("Something went wrong with the File Type Options")

```

```

fileName, _ = QtWidgets.QFileDialog.getOpenFileName(None, "Select Input File", "", inputFileType)

if fileName:

    self.txtPath_Tool3.setText(fileName)

    fn = fileName.split("/")

    fn1 = fn[len(fn) - 1].split(".")

    fn2 = fn1[len(fn1) - 1]

    if len(fn1) < 2 or (len(fn1) > 1 and not (("pcap" in fn2) or ("csv" in fn2))):

        self.showMessage("The file you have provided is not acceptable")

        return

    elif not os.path.exists(fileName):

        self.showMessage("The file you have provided does not exist")

        return

    self.btnStart_Tool3.setEnabled(True)

# this function disables and enables all the available options based on the validity of the given file
def pathChanged_Tool3(self):

    # disables the text browser by default until the button Start is clicked
    self.textBrowser_Tool3.setEnabled(False)

    # disables the start button by default until the file is selected
    self.btnStart_Tool3.setEnabled(False)

    fileName = self.txtPath_Tool3.text()

    fileType = self.comboBoxFileType_Tool3.currentText()

    fileType1 = fileType.split(".")

    if (fileType1[1].startswith("pcap")):

        if ((fileName is not None) and (len(fileName) > 0)):

            if ((fileName.endswith(".pcap")) and (len(fileName) > len(".pcap"))):

                self.btnStart_Tool3.setEnabled(True)

    elif (fileType1[1].startswith("csv")):

        if ((fileName is not None) and (len(fileName) > 0)):

            if ((fileName.endswith(".csv")) and (len(fileName) > len(".csv"))):

                self.btnStart_Tool3.setEnabled(True)

    else:

        self.showMessage("Something went wrong with the File Type Options")

# this function disables and enables all the available options based on the File Type option
def clickedFileType_Tool3(self):

    fileName = self.txtPath_Tool3.text()

```

```

fileType = self.comboBoxFileType_Tool3.currentText()

fileType1 = fileType.split(".")

if (fileType1[1].startswith("pcap")):

    if ((fileName is not None) and (len(fileName) > 0)):

        if ((fileName.endswith(".pcap")) and (len(fileName) > len(".pcap"))):

            self.btnStart_Tool3.setEnabled(True)

        else:

            # disables the text browser by default until the button Start is clicked
            self.textBrowser_Tool3.setEnabled(False)

            # disables the start button by default until the file is selected
            self.btnStart_Tool3.setEnabled(False)

    self.comboBoxDelete_Tool3.setEnabled(True)

    self.comboBoxDelete_Tool3.setCurrentIndex(0)

elif (fileType1[1].startswith("csv")):

    if ((fileName is not None) and (len(fileName) > 0)):

        if ((fileName.endswith(".csv")) and (len(fileName) > len(".csv"))):

            self.btnStart_Tool3.setEnabled(True)

        else:

            # disables the text browser by default until the button Start is clicked
            self.textBrowser_Tool3.setEnabled(False)

            # disables the start button by default until the file is selected
            self.btnStart_Tool3.setEnabled(False)

    self.comboBoxDelete_Tool3.setCurrentIndex(1)

    self.comboBoxDelete_Tool3.setEnabled(False)

else:

    self.showMessage("Something went wrong with the File Type Options")

# this function, calls the IDS functions based on the options provided
def start_Tool3(self):

    enabledPath = False

    enabledBrowse = False

    enabledFileType = False

    enabledDelete = False

    if (self.txtPath_Tool3.isEnabledTo(self.Tool3)):

        enabledPath = True

    if (self.btnBrowse_Tool3.isEnabledTo(self.Tool3)):

```

```

        enabledBrowse = True

    if (self.comboBoxFileType_Tool3.isEnabledTo(self.Tool3)):

        enabledFileType = True

    if (self.comboBoxDelete_Tool3.isEnabledTo(self.Tool3)):

        enabledDelete = True

    self.txtPath_Tool3.setEnabled(False)

    self.btnBrowse_Tool3.setEnabled(False)

    self.comboBoxFileType_Tool3.setEnabled(False)

    self.comboBoxDelete_Tool3.setEnabled(False)

    self.btnStart_Tool3.setEnabled(False)

    self.textBrowser_Tool3.setEnabled(True)

    fileNameArg = self.txtPath_Tool3.text()

    fileTypeArg = self.comboBoxFileType_Tool3.currentText()

    delOptionArg = self.comboBoxDelete_Tool3.currentText()

    delOptionBool = "true"

    if (delOptionArg == "No"):

        delOptionBool = "false"

    tool = 3

    # call the functions here

    QtCore.QCoreApplication.processEvents()

    # function call
    if (fileTypeArg == 'pcap file "<filename>.pcap"'):

        if ((fileNameArg is not None) and (len(fileNameArg) > 0)):

            if ((fileNameArg.endswith(".pcap")) and (len(fileNameArg) > len(".pcap"))):

                self.start_PcapToTest(tool, delOptionBool, fileNameArg)

    elif (fileTypeArg == 'csv file "<filename>.csv"'):

        if ((fileNameArg is not None) and (len(fileNameArg) > 0)):

            if ((fileNameArg.endswith(".csv")) and (len(fileNameArg) > len(".csv"))):

                self.start_Test(tool, fileNameArg)

    QtCore.QCoreApplication.processEvents()

    if (enabledPath):

        self.txtPath_Tool3.setEnabled(True)

    if (enabledBrowse):

        self.btnBrowse_Tool3.setEnabled(True)

```

```

if (enabledFileType):
    self.comboBoxFileType_Tool3.setEnabled(True)

if (enabledDelete):
    self.comboBoxDelete_Tool3.setEnabled(True)

self.btnStart_Tool3.setEnabled(True)

#####

# show a pop up message
def showMessage(self, m):

    msg = QMessageBox()

    msg.setWindowTitle("Attention")

    msg.setText(m)

    msg.setIcon(QMessageBox.Critical)

    x = msg.exec_()

def main():

    app = QtWidgets.QApplication(sys.argv)

    window = QtWidgets.QWidget()

    ui = Ui_Form()

    ui.setupUi(window)

    window.show()

    sys.exit(app.exec_())

if __name__ == "__main__":

    main()

```