Πανεπιστήμιο Πειραιώς
Τμήμα Ψηφιακών Συστημάτων
Π.Μ.Σ. " Ψηφιακές Επικοινωνίες & Δίκτυα "

Μεταπτυχιακή Διπλωματική Εργασία

"Παρακολούθηση αρχείων καταγραφής/ειδοποιήσεων ασφαλείας βασισμένη στην τεχνητή νοημοσύνη / μηχανική μάθηση για έξυπνη διαχείριση απειλών σε σύγχρονα δίκτυα"

Καμπάνης Αλέξανδρος Ιάσονας, MWE2004

02/2023

Υπό την επίβλεψη του Κωνσταντίνου Τσαγκάρη

Πίνακας Περιεχομένων

1. Εισαγωγή

Στην σημερινή εποχή τα πληροφοριακά συστήματα και τα συστήματα επικοινωνιών βρίσκονται παντού και παίζουν καθοριστικό ρόλο στην καθημερινότητα και πολλές από τις δραστηριότητες ανθρώπων και οργανισμών. Παραδείγματα είναι τα κοινωνικά δίκτυα και το e-banking όπου και στις δύο περιπτώσεις η διαθεσιμότητα και η ασφάλεια παίζουν σημαντικό ρόλο και έχουν αντίκτυπο που μπορεί να μας επηρεάσει αρνητικά. Στην πρώτη περίπτωση η ασφάλεια είναι επιβεβλημένη ανάγκη διότι χρησιμοποιώντας τα κοινωνικά δίκτυα για να επικοινωνούμε και να μοιραζόμαστε εμπειρίες μπορεί να καταλήξουν δεδομένα προσωπικού χαρακτήρα που δεν θα θέλαμε να γίνουν γνωστά σε τρίτους με αποτέλεσμα να εκτεθούμε ή ακόμα χειρότερα να πέσουμε θύμα εκβιασμού ή απάτης. Στην δεύτερη περίπτωση και επειδή οι ηλεκτρονικές συναλλαγές πλέον αποτελούν τον κανόνα η πιθανότητα να γίνει κάποιος στόχος εγκληματιών αυξάνεται. Οι εγκληματίες μπορεί απλά να είναι απατεώνες που αναλώνονται σε τεχνικές κοινωνικής μηχανικής με πιο γνωστό παράδειγμα το phishing ή ομάδες υψηλά καταρτισμένων τεχνολογικά ανθρώπων με πολλούς πόρους και οργάνωση ή/και ανθρώπους με τους οποίους συνεργάζονται σε κάποιον οργανισμό που είναι στόχος. Τέτοιες ομάδες μπορούν να στοχεύσουν είτε άτομα, είτε μαζικά οργανισμούς όπως τράπεζες ή καταστήματα με πολλούς πελάτες με στόχο να αποκτήσουν πρόσβαση σε δεδομένα πολλών ανθρώπων με μία καλά οργανωμένη και πολύπλοκη επίθεση. Τέτοια περιστατικά ναι μεν είναι πιο σπάνια από το πρώτο είδος που αναφέραμε αλλά μπορούν να έχουν μεγάλο αντίκτυπο όταν συμβούν σε αντίθεση με τα πρώτα όπου ο αντίκτυπος είναι περιορισμένος. Και στα δύο σενάρια οι στόχοι είναι να βρεθούν τρόποι να αναγνωρίσουμε σε πρώτο χρόνο ότι δεχόμαστε επίθεση, να έχουμε εργαλεία να σταματήσουμε την επίθεση, να έχουμε εργαλεία να μετριάσουμε τις επιπτώσεις της επίθεσης και να διατηρήσουμε στοιχεία για να ερευνήσουμε την πιθανή επίθεση. Τα εργαλεία αυτά δεν είναι αναγκαστικά τεχνικής φύσης αλλά πολλές φορές είναι μέτρα διαχειριστικά όπως τακτικοί έλεγχοι και εφαρμογή πολιτικών που στόχο έχουν να εκπαιδεύσουν χρήστες σε σωστή συμπεριφορά ή να την επιβάλουν μέσω προκαθορισμένων ρυθμίσεων σε σταθμούς εργασίας.

Για αυτό τον λόγο οι πόροι που αφιερώνονται στην έρευνα για την ασφάλεια από οργανισμούς και πανεπιστήμια όλο και αυξάνονται. Γενικά η ασφάλεια έχει ως στόχο να διασφαλίσει πρώτον το Confidentiality, δηλαδή το ότι ευαίσθητα δεδομένα παραμένουν κρυφά. Δεύτερον το availability το οποίο στόχο έχει μία κρίσιμη υπηρεσία παραμένει διαθέσιμη κατελάχιστο για το χρόνο που έχει προγραμματιστεί ως αρκετός. Τρίτον το integrity, δηλαδή το ότι τα δεδομένα που μεταφέρονται η αποθηκεύονται δεν έχουν αλλοιωθεί και ακόμα να μπορούμε να γνωρίζουμε το ότι αν συνέβη κάτι τέτοιο είμαστε σε θέση το λιγότερο να το εντοπισουμε. Φυσικά αυτός ο ορισμός της ασφάλειας είναι πολύ γενικός και στην πράξη τα προβλήματα που παρουσιάζονται είναι πιο

συγκεκριμένα και δεν κατατάσσονται αποκλειστικά σε μία κατηγορία. Έτσι εδώ και δεκαετίες έχουν αναπτυχθεί εργαλεία που στόχο έχουν να μας προστατέψουν από κινδύνους όπως περιγράφηκαν. Τα πιο γνωστά είναι τα antivirus τα firewalls και τα IDS/IPS. Επίσης υπάρχουν και άλλα τεχνικά μέσα για να προσπαθούμε να διασφαλίσουμε και να μετριάσουμε επιπτώσεις όπως οι τεχνικές sandboxing οι οποίες φέρνουν πολύ καλά αποτελέσματα στον μετριασμό και χρησιμοποιούνται πλέον πολύ συχνά σε πολλές εφαρμογές. Όσον αφορά τα antivirus στόχος τους είναι να εντοπίσουν κακόβουλο λογισμικό πριν εκτελεστεί ή κατά την διάρκεια που εκτελείται και όταν αναγνωριστεί να περιοριστεί και να διαγραφεί. Ο τρόπος που το πετυχαίνουν αυτό ποικίλει από πολύ απλές μεθόδους όπως υπογραφές γνωστών κακόβουλων λογισμικών και πιο προχωρημένες τεχνικές. Βασική προϋπόθεση για να έχει αποτέλεσμα τέτοιο λογισμικό είναι να έχει εγκατασταθεί πριν γίνει κάποια επίθεση καθώς αν το κακόβουλο λογισμικό εγκαταστήσει κάποιο rootkit ή αποκτήσει πρόσβαση στο BIOS τότε το antivirus δεν μπορεί να κάνει τίποτα. Τα firewalls αποτελούν μία πολύ καλή λύση η οποία πλέον βρίσκεται παντού. Στόχος είναι να φιλτράρει κίνηση στο δίκτυο ώστε να εμποδίσει κίνηση που δεν πρέπει να μπει ή να βγει από το δίκτυο. Ακόμα και αν ένα μηχάνημα είναι υπό τον έλεγχο του επιτιθέμενου ένα firewall είναι σε θέση να μην το αφήσει να επικοινωνήσει εκτός του τοπικού δικτύου. Τέλος τα IDS/*IPS αποτελούν μία λύση που στόχο έχει να παρακολουθεί σε πραγματικό χρόνο ή σε κοντά σε πραγματικό χρόνο την δραστηριότητα και την κίνηση σε ένα δίκτυο ή ένα host. Αποτελούν μία εύκολη και καλή λύση που μπορεί να εφαρμοστεί σε κάθε τμήμα του δικτύου και ανάλογα με την περίπτωση χρήσης μπορεί να δώσει πολύ καλά αποτελέσματα. Τα IDS έχουν στόχο να ανιχνεύουν και τα IPS έχουν επίσης στόχο και να προλαμβάνουν επιθέσεις. Παρά τα διαφορετικά ονόματα τα οποία προέκυψαν για εμπορικούς λόγους στην πράξη κάνουν το ίδιο πράγμα. Το πρόβλημα είναι ότι βασίζονται κυρίως σε ανίχνευση με βάση υπογραφές(signature based detection) το οποίο είναι πολύ περιοριστικό και δεν έχουν συνήθως την δυνατότητα να εντοπίσουν νέες άγνωστες επιθέσεις και η όλη διαδικασία δεν είναι πλήρως αυτοματοποιημένη σε μεγάλο βαθμό και περιλαμβάνει άλλο λογισμικό γνωστό ως SIEM το οποίο θα περιγράψουμε στην συνέχεια καθώς και ειδικούς οι οποίοι αναλύουν χειροκίνητα ύποπτα logs που εντόπισαν τα παραπάνω και επίσης σκοπός τους είναι να παράγουν χειροκήνητα υπογραφές επιθέσεων με βάση την εμπειρία τους από την ανάλυση των logs. Τα IDS/*IPS γενικά μπορούν να κατηγοριοποιηθούν με πολλά κριτήρια τα οποία θα παρουσιάσουμε στην συνέχεια.

Με βάση όλα αυτά στόχος είναι να έχουμε συστήματα αρκετά ευέλικτα χωρίς αυτό να σημαίνει ότι εγκαταλείπουμε δοκιμασμένες τεχνικές που αποδίδουν ώστε να αυξηθεί η ακρίβεια(accuracy) στην ανίχνευση καθώς και να μειωθούν οι λάθος προβλέψεις(false positives) οι οποίες μπορούν να προκαλέσουν σημαντικά προβλήματα από μόνα τους στον στόχο του availability. Σε αυτόν τον στόχο εισάγεται η έννοια της γενίκευσης(generalizability)**[1,2]** των καταστάσεων με χρήση μηχανικής μάθησης.

Η μηχανική μάθηση είναι μια υποκατηγορία της τεχνητής νοημοσύνης και στόχο έχει να εξάγει γνώση από δεδομένα, να δημιουργήσει και να εκπαιδεύσει ένα μοντέλο από αυτά τα δεδομένα και στην συνέχεια να κάνει βγάλει το μοντέλο σε χρήση και να εφαρμόσει την γνώση που απέκτησε σε καινούρια δεδομένα ώστε να παράγει αποτελέσματα. Αυτή η προσέγγιση μπορεί να κάνει την όλη διαδικασία πιο ευέλικτη και αποτελεσματική μειώνοντας αλλά όχι εξαλείφοντας τις λανθασμένες θετικές εκτιμήσεις(false positives) και αυξάνοντας την ακρίβεια(accuracy). Ανάλογα το σενάριο χρήσης μπορεί να δώσει πολύ καλά αποτελέσματα και με τα δύο αυτά μετρικά όπως στην περίπτωση απλών αλλά συνηθισμένων DoS επιθέσεων. Συγκεκριμένα μας ενδιαφέρει ο τομέας της επιβλεπόμενης μηχανικής μάθησης αν και αυτό δεν αποκλείει την χρήση άλλων όπως η συσταδοποίηση(clustering) η οποία κατηγοριοποιείται ως μή επιβλεπόμενη και συνήθως χρησιμοποιείται για ανίχνευση μη συνηθισμένων οντοτήτων(outlier detection) . Αυτή η κατηγορία ακολουθεί την λογική του ότι έχουμε δείγματα(datapoints) τα οποία έχουν διάφορα χαρακτηριστικά(features) τα οποία συλλέγονται από διάφορες πηγές όπως logs του λειτουργικού συστήματος κάποιας εφαρμογής ή βάσης δεδομένων ή από κάποιο firewall που βρίσκεται σε λειτουργία. Επίσης συσχετισμένο με κάθε δείγμα έχουμε και μία ετικέτα(label) γνωστή και ως target variable. Τροφοδοτώντας το μοντέλο με ζεύγη από δείγματα/ετικέτες κατά την διάρκεια της εκπαίδευσης το μοντέλο μαθαίνει σε έναν χώρο μεγάλης διάστασης να διαχωρίζει τα δείγματα έτσι ώστε να μειώνει το σφάλμα που παράγει. Παρόλο που η διαδικασία στην θεωρία φαίνεται απλή στην πράξη υπάρχουν πολλά πράγματα κατά την διάρκεια της διαδικασίας της εκπαίδευσης που μπορεί να χρειαστούν ρύθμιση(fine tuning) με πιο χαρακτηριστικό παράδειγμα την επιλογή των υπερ παραμέτρων(hyper parameters). Αυτές αποτελούν παραμέτρους του αλγορίθμου και δεν έχουν σχέση με τα δεδομένα μάθησης αλλά η σωστή επιλογή τους παίζει σημαντικό ρόλο στην επιτυχία ή αποτυχία του μοντέλου. Ο κάθε αλγόριθμος έχει τις δικές του υπερ παραμέτρους και θα αναφερθούμε αναλυτικά σε αυτό το σημαντικό σημείο σε ξεχωριστη ενότητα στην συνέχεια. Γενικά τα μοντέλα που μπορούμε να αναπτύξουμε χωρίζονται σε δύο κατηγορίες. Η πρώτη είναι τα λεγόμενα απλά(shallow) μοντέλα. Αποτελούν την πιο παραδοσιακή αντιμετώπιση και είναι κατάλληλα όταν δεν έχουμε πολλά δεδομένα και τα δεδομένα δεν βρίσκονται σε υψηλή διάσταση καθώς παράγουν καλά αποτελέσματα με λίγα. Επίσης είναι ιδανικά για δεδομένα τα οποία έχουν από μόνα τους κάποια δομή όπως οι κεφαλίδες ενός πακέτου σε αντίθεση με το φορτίο(payload). Παραδείγματα τέτοιων μοντέλων είναι τα Decision Trees, Random Forests, SVM, Naive Bayes και άλλα. Η δεύτερη κατηγορία είναι τα μοντέλα βαθιάς Μηχανικής Μάθησης. Η κατηγορία αυτή ονομάζεται Βαθιά Μηχανική μάθηση και περιλαμβάνει νευρωνικά δίκτυα τα οποία σε πολλές περιπτώσεις έχουν πολλά κρυμμένα επίπεδα και για αυτό το λόγο ονομάζονται deep. Παρόλο που σε παρεμφερείς εργασίες συνήθως χρησιμοποιούνται απλά μοντέλα υπάρχουν και περιπτώσεις όπου χρησιμοποιούνται νευρωνικά δίκτυα οπότε θα συμπεριλάβουμε και αυτά στην παρουσίαση που θα δημιουργήσουμε.

Γενικά όταν δημιουργούμε ένα απλό(shallow) μοντέλο και έχουμε πολλά χαρακτηριστικά(features) προσπαθούμε να ελαχιστοποιήσουμε το σύνολο από αυτά που θα χρησιμοποιήσουμε ώστε να συμπεριλάβουμε μόνο αυτά που παρέχουν σημαντική πληροφορία μή επαναλαμβανόμενη(redundancy) καθώς δημιουργείται προκατάληψη(bias) προς συγκεκριμένα δείγματα και χαρακτηριστικά(features) που πρακτικά είναι απλά θόρυβος. Η εφαρμογή τέτοιων τεχνικών αποτελεί πρόκληση στα σύγχρονα συστήματα. Ειδικά με τις τελευταίες εξελίξεις στα δίκτυα και την εικονικοποίηση(virtualization) με κύριο χαρακτηριστικό την αύξηση της λειτουργικότητας που εκτελείται και συνεπώς της πολυπλοκότητας δικτύων και συστημάτων το attack surface αυξάνεται καθιστώντας πλέον από αναγκαία την απαίτηση για ασφάλεια.

Στα σύγχρονα δίκτυα και με την έλευση και υιοθέτηση της εικονικοποίησης(virtualization) πλέον οι απλές συσκευές δικτύου που απλά αναλύουν ένα ή δύο headers και βασίζονται σε λογισμικό που παράγουν οι κατασκευαστές αντικαθίστανται από γενικής χρήσης εικονικά μηχανήματα στο cloud τα οποία εκτελούν πολύπλοκο λογισμικό με δυνητικά πολύπλοκη λειτουργικότητα κάτι το οποίο ανοίγει τον δρόμο για περισσότερες πιθανές αδυναμίες και μεγαλύτερη ποικιλότητα στο λογισμικό ανάμεσα σε διαφορετικούς παρόχους. Ένας από τους κανόνες της ασφάλειας είναι το γεγονός ότι όταν αυξάνει η πολυπλοκότητα, οι λειτουργίες και τα εμπλεκόμενα μέρη σε ένα σύστημα τότε αυξάνονται και οι πιθανές αδυναμίες. Αυτό δεν μπορεί παρα να ισχύει για τα μοντέρνα δίκτυα.

Σχετικά με την νεφουπολογιστικη παρόλο που αυξάνει τις πιθανές αδυναμίες έχει και πολλά θετικά. Η υιοθέτηση αυτού του μοντέλου προσφέρει χαμηλότερο κόστος καθώς γίνεται χρέωση με βάση την χρήση. Επίσης προσφέρει μεγαλύτερη ελαστικότητα καθώς όταν αυξάνουν οι ανάγκες είναι πολύ εύκολο να προστεθούν περισσότεροι πόροι και να γίνουν άμεσα διαθέσιμοι για όσο χρειάζεται χωρίς ο κάθε οργανισμός να προχωρήσει σε επένδυση σε υλικό που στο μέλλον ίσως να μη χρειάζεται. Επίσης θεωρητικά βελτιώνεται και η ασφάλεια παρόλο που αυξάνεται το attack surface διότι ο πάροχος έχει προνοήσει ώστε όλες οι τελευταίες ενημερώσεις ασφαλείας να είναι άμεσα διαθέσιμες σε όλα τα συστήματα, αν ο πελάτης έχει ζητήσει αυτήν την υπηρεσία, κάτι που πολλές φορές σε οργανισμούς που διατηρούν την δικιά τους υποδομή είναι πιο δύσκολο λόγω περιορισμένων ανθρώπινων πόρων ή άλλων ζητημάτων. Επίσης τα πρωτόκολλα διαχείρισης συσκευών δικτύου είναι πλέον πολύ πιο ώριμα σχετικά με την ασφάλεια σε σχέση με προκατόχους όπως η πρώτη έκδοση του SNMP. Με βάση όλα αυτά είναι δεδομένο ότι αυτή η αλλαγή στο μοντέλο θα είναι μία αλλαγή που θα μείνει και πρέπει να αντιμετωπιστούν οι προκλήσεις και οι ιδιαιτερότητες.

Στόχος της εργασίας είναι να παρουσιάσει όλη την βασική γνώση και τη διαδικασία καθώς και τα εργαλεία και διαδικασίες που εμπλέκονται στον τρόπο που αντιμετωπίζεται αυτή την στιγμή το πρόβλημα. Στην συνέχεια θα παρουσιάσουμε τρόπους που χρησιμοποιούν τις εξελίξεις στην διαθεσιμότητα δεδομένων που μπορούν να οδηγήσουν στην ανάπτυξη μοντέλων. Επίσης θα αναφερθούμε σε προηγούμενες εργασίες που κινούνται σε αυτή την κατεύθυνση. Επίσης θα

παρουσιάσουμε αναλυτικά όλη την προηγούμενη γνώση σχετικά με διαδικασίες και εργαλεία που απαιτείται για την κατανόηση και θα αναπτύξουμε ένα σύστημα βασισμένο σε μοντέλο μηχανικής μάθησης το οποίο θα εκπαιδεύσουμε. Όπως θα δούμε οι εργασίες σε αυτό το τομέα μένουν στην εκπαίδευση ενός μοντέλου μηχανικής μάθησης και δεν αναφέρονται σε όλη την διαδικασία και τα προβλήματα που πρέπει να αντιμετωπιστούν για να δημιουργηθεί ένα λειτουργικό IDS βασιμένο σε Μηχανική Μάθηση. Το υπόλοιπο της εργασίας είναι δομημένο ως εξής. Στην ενότητα 2 θα παρουσιάσουμε όλη την βασική γνώση που απαιτείται για την κατανόηση της θεωρίας και της υλοποίησης. Στην ενότητα 3 θα παρουσιάσουμε σχετικές δουλειές από την βιβλιογραφία. Στην ενότητα 4 θα παρουσιάσουμε την αρχιτεκτονική της υλοποίησης και τα components που αποτελούν την τελική λύση. Στην ενότητα 5 θα παρουσιάσουμε τις λεπτομέρειες της υλοποίησης και τέλος στην ενότητα 6 θα αναφερθούμε στα βασικά σημεία που πρέπει να δοθεί έμφαση καθώς και σε αδυναμίες και περιορισμούς που πρέπει να ληφθούν υπόψη και να αντιμετωπιστούν.

2.Προαπαιτούμενη Γνώση

2.1 Τεχνητή Νοημοσύνη

Η τεχνητή νοημοσύνη**[3,4]** αποτελεί ένα πεδίο με ρίζες σε πολλούς επιστημονικούς τομείς όπως τα μαθηματικά και πιο συγκεκριμένα στατιστική, ανάλυση, γραμμική άλγεβρα. Επίσης σχετίζεται με την βιολογία και συγκεκριμένα το πώς ζωντανοί νοήμονες οργανισμοί καταφέρνουν να φέρουν εις πέρας πολύπλοκες διαδικασίες και να μαθαίνουν μέσα από εξερεύνηση του περιβάλλοντος τους και των διαθέσιμων ενεργειών που μπορούν να πραγματοποιήσουν. Ο στόχος του general artificial intelligence**[5]** αποτελεί τον απόλυτο στόχο πολλών ερευνητών εδώ και πολλές δεκαετίες. Πρόκειται για την δημιουργία ευφυών πρακτόρων που είναι σε θέση να εκτελέσουν ένα μεγάλο εύρος δραστηριοτήτων κατα τα πρότυπα ευφυών οργανισμών όπως οι άνθρωποι και τα ζώα. Αυτό περιλαμβάνει την δυνατότητα να μαθαίνει να κινείται στο χώρο αν μιλάμε για πράκτορα που δεν είναι μόνο λογισμικό αλλά έχει υλική υπόσταση, περιλαμβάνει να μαθαίνει τους κανόνες που πρέπει να ακολουθεί αν αλληλεπιδρά με ανθρώπους και άλλα. Επίσης στην περίπτωση αλληλεπίδρασης με ανθρώπους θα πρέπει να μπορεί να καταλαβαίνει την φυσική γλώσσα και να είναι σε θέση και να ερμηνεύει αλλά και να παράγει σε αυτό το τομέα. Το τελευταίο είναι υποκλάδος της τεχνητής νοημοσύνης με τον τίτλο Επεξεργασία φυσικής γλώσσας**[6]**. Ο συγκεκριμένος κλάδος

υπάρχει δεκαετίες και περιλαμβάνει πολλά tasks όπως η αυτόματη μηχανική μετάφραση**[7]** πάνω στο οποίο είχε γίνει πολλή έρευνα κατα την διάρκεια του ψυχρού πολέμου. Ο συγκεκριμένος τομέας γνώρισε μεγάλη εξέλιξη την τελευταία δεκαετία λόγω των καινοτομιών στα νευρωνικά δίκτυα. Πριν τα νευρωνικά δίκτυα με πιο παραδοσιακές τεχνικές**[8]** που βασίζονταν σε tagged από ανθρώπους corpora και επιβλεπόμενη μάθηση, υπήρχε κάποια πρόοδος στον τομέα αλλά με την έλευση των νευρωνικών δικτύων η απόδοση εκτοξεύτηκε**[6]**. Με πολλή λίγη προεπεξεργασία, προεκπαιδευμένα μοντέλα όπως το BERT**[9]** και έξυπνες τεχνικές μάθησης με χρήση unlabeled κείμενο**[9]** όπου εκπαιδεύεται το μοντέλο κρύβοντας(mask) λέξεις στο κείμενο η απόδοση ξεπέρασε κατά πολύ προηγούμενες μεθόδους. Επίσης εκτός από το να μπορεί να αλληλεπιδρά με ανθρώπους ένας ευφυής πράκτορας θα πρέπει να είναι σε θέση να αλληλεπιδρά με το περιβάλλον του. Αυτό πολλές φορές περιλαμβάνει υψηλής διάστασης χώρο(high dimensional state space) και απαιτήσεις για πραγματικού χρόνου ανταπόκριση. Με βάση αυτά ο πράκτορας θα πρέπει να υλοποιεί μηχανισμούς αναπαράστασης γνώσης και μηχανισμούς εξαγωγής συμπερασμάτων(inference) τομέας της τεχνητής νοημοσύνης που αναφέρεται στην βιβλιογραφία ώς Αναπαράσταση Γνώσης και Συλλογιστική(Knowledge representation and reasoning)**[10]**. Επίσης βασικό ρόλο παίζουν οι τεχνικές αναζήτησης**[11]** που μπορούν να εφαρμόζονται από τον πράκτορα κατα την διάρκεια της εξαγωγής συμπερασμάτων(inference) σε ένα πιθανό χώρο. Παρόλο που ο στόχος του general AI βρίσκεται πιθανόν μακριά αν και με την εξέλιξη που βιώνουμε τα τελευταία χρόνια ίσως και να είναι πιο κοντά από όσο νομίζουμε η επιτυχία σε συγκεκριμένα προβλήματα περιλαμβανομένων και όσων αναφέραμε προηγουμένως είναι δεδομένη. Κάποια από αυτά τα προβλήματα**[12]** αφορούν chatbots, computer vision applications ,autonomous driving, recommendation engines,machine translation και πολλά άλλα. Επίσης οι εφαρμογές είναι πραγματικά απεριόριστες και οι επιστημονικοί τομείς που βρίσκουν εφαρμογή πολλοί με πιο ενδιαφέρον τομέα αυτον της μηχανικής μάθησης και της βαθιάς μηχανικής μάθησης στα οποία θα αναφερθούμε αναλυτικά στην συνέχεια. Γενικά η μηχανική μάθηση χωρίζεται σε 3 κατηγορίες**[12]** αν και όπως θα δούμε υπάρχουν και περιπτώσεις που αυτό δεν είναι απόλυτο. Παράδειγμα αποτελεί η ημι επιβλεπόμενη μάθηση(semi-supervised learning)**[13]** η οποία για παράδειγμα στην περίπτωση των δεδομένων γράφων και τα Graph Neural Networks**[14]** συνδυάζει επιβλεπόμενη και μη επιβλεπόμενη μάθηση.

Η πρώτη κατηγορία είναι η επιβλεπόμενη μάθηση**[12]** η οποία βασίζεται στην ιδέα της ύπαρξης ενός dataset που αποτελείται από χαρακτηριστικά(features) και ετικέτες(labels). Με χρήση μίας συνάρτησης λάθους(loss function)**[15]** και ενός αλγόριθμου βελτιστοποίησης**[16]** δημιουργούμε ένα μοντέλο που χαρακτηρίζεται από κάποιες παραμέτρους, αλλάζοντας με μικρά βήματα τις παραμέτρους του μοντέλου με στόχο να μηδενίσουμε ή να φέρουμε σε ένα ελάχιστο την συνάρτηση του κόστους η οποία μετράει την απόσταση της πρόβλεψης του μοντέλου και της πραγματικής τιμής. Εκπαιδεύοντας για πολλές εποχές το μοντέλο τελικά μαθαίνει να προβλέπει σωστά δεδομένα που δεν

έχει δεί. Περιπτώσεις επιβλεπόμενης μηχανικής μάθησης είναι το regression όπου η ετικέτα είναι ένας αριθμός και το classification όπου έχουμε πολλές κλάσεις όπου μπορεί να ανήκει ένα δείγμα.

Η δεύτερη κατηγορία είναι η μη επιβλεπόμενη μάθηση**[12]**. Εδώ δεν έχουμε ετικέτες αλλά μόνο χαρακτηριστικά(features). Σκοπός είναι να δημιουργήσουμε ομάδες από τα δεδομένα όπου με βάση κάποιο μέτρο απόστασης κοντινά δείγματα να βρίσκονται στην ίδια ομάδα και διαφορετικά δείγματα να βρίσκονται σε διαφορετικές ομάδες. Αυτός είναι ο πιο απλός ορισμός για να μετρήσουμε την ποιότητα του cluster αλλά υπάρχουν και άλλοι. Εφαρμογές της συσταδοποίησης είναι για παράδειγμα communities detection outlier detection, recommendation engines, market and customer segmentation social network analysis. Γενικά μπορούμε να χωρίσουμε τους αλγορίθμους σε κατηγορίες με βάση την έννοια της ομοιότητας που χρησιμοποιείται. Οι κατηγορίες αυτές**[17]** είναι γενικά οι connectivity based clustering, centroid based clustering, Distribution based clustering και density based clustering. Κάποιοι από τους πιο γνωστούς αλγορίθμους είναι οι k-means και spectral clustering. Περισσότερα στη αντίστοιχη ενότητα παρακάτω. Τέλος θα κάνουμε μία σύντομη αναφορά στην Ενισχυτική Μάθηση(Reinforcement Learning)**[18]**. Εδώ ο πράκτορας αλληλεπιδρά με το περιβάλλον προσπαθόντας να μεγιστοποιήσει την ανταμοιβή του μακροπρόθεσμα. Θα κάνουμε μία αναφορά σε αυτή την μέθοδο μηχανικής μάθησης στην αντίστοιχη ενότητα αλλά σύντομη διότι δεν θα μας απασχολήσει αυτή η μέθοδος σε αυτήν την εργασία.

2.2 Μηχανική Μάθηση

2.2.1 Βασικές Αρχές Μηχανικής Μάθησης

Η Μηχανική Μάθηση αδιαμφισβήτητα είναι μία από τις πιο υποσχόμενες και με επιρροή τεχνολογίες της σημερινής εποχής και χωρίς υπερβολή δεν έχουμε δει ακόμα τις πλήρης δυνατότητες της. Υπάρχει σαν κλάδος πολύ καιρό αλλα λόγω της παραγωγής και αποθήκευσης μεγάλου όγκου δεδομένων απέκτησε σκοπό και χρήση τις τελευταίες δεκαετίες διότι όλα αυτά τα δεδομένα είναι άχρηστα όταν δεν έχουμε τα κατάλληλα εργαλεία για να τα αξιοποιήσουμε. Οι μέθοδοι της Μηχανικής Μάθησης στόχο έχουν να βρουν κρυμμένα μοτίβα(patterns) σε δεδομένα που αλλιώς θα ήταν αδύνατο να βρεθούν. Αυτά τα κρυμμένα μοτίβα και η γνώση χρησιμοποιούνται στην συνέχεια για να κάνουμε προβλέψεις σε άγνωστα νέα δεδομένα. Οι περισσότεροι από εμάς δεν αντιλαμβανόμαστε ότι αλληλεπιδρούμε καθημερινά με αλγορίθμους Μηχανικής Μάθησης. Κάθε φορά που κάνουμε μία αναζήτηση σε κάποια μηχανή αναζήτησης ή ανεβάζουμε κάποιο video ή φωτογραφία σε κάποιο κοινωνικό δίκτυο ή κάνουμε μία συναλλαγή με πιστωτική κάρτα

αλληλεπιδρούμε με κάποιον αλγόριθμο Μηχανικής Μάθησης. Στην πρώτη περίπτωση η μηχανή αναζήτησης χρησιμοποιεί δεδομένα που έχουν συλλεχθεί στο παρελθόν για εμάς και δεδομένα που έχουν συλλεγεί από άλλους για να μας δώσει το βέλτιστο αποτέλεσμα και για να μας παρουσιάσει πιο στοχευμένη διαφήμιση. Στην δεύτερη περίπτωση η ανάλυση των δεδομένων επίσης έχει ως στόχο την πιο στοχευμένη διαφήμιση αλλα παρέχονται και εργαλεία τα οποία αναλύουν πολυτροπικό(multimodal)**[19]** υλικό ώστε να αναγνωρίζουν πρόσωπα ή φωτογραφίες και κείμενα των οποίων το περιεχόμενο δεν είναι αποδεκτό. Παραδείγματα μη αποδεκτού περιεχομένου είναι αναρτήσεις ήχου/εικόνας/κειμένου που υμνουν τον ρατσισμό την ομοφοβία ή την τρομοκρατία. Στο τελευταίο παράδειγμα με τις συναλλαγές μοντέλα μηχανικής μάθησης μαθαίνουν το προφίλ χρηστών της υπηρεσίας ώστε να κατηγοριοποιούν τις συναλλαγές ώς κανονικές ή όχι έτσι ώστε αν μια συναλλαγή είναι ύποπτη να εφαρμόζουν μέτρα όπως κλείδωμα της κάρτας ή υποχρέωση επιβεβαίωσης της συναλλαγής με PIN ή two factor authentication.

Υπάρχουν πολλοί ορισμοί για το τί είναι μηχανική μάθηση**[12]**. Ένας από τους πιο χαρακτηριστικους είναι αυτός που έδωσε ο Tom Mitchell. “Machine Learning is the study of algorithms that allow computer programs to automatically improve through experience ”. Γενικά μπορούμε να πούμε ότι η μηχανική μάθηση είναι ένα εργαλείο που μετατρέπει την πληροφορία σε γνώση. Γενικά η μηχανική μάθηση αφορά την αποκόμιση γνώσης από δεδομένα. Χωρίς δεδομένα η μηχανική μάθηση είναι απλά αλγόριθμοι χωρίς προοπτική. Τις τελευταίες δεκαετίες όπου η χρήση του διαδικτύου μπήκε στη ζωή πολλών ανθρώπων και οργανισμών άρχισαν να παράγονται μαζικά πολλά δεδομένα τα οποία είναι διαθέσιμα είτε ανοιχτά δηλαδή διαθέσιμα σε όλους όπως είναι οι πληροφορίες στο wikipedia και γενικότερα στο internet, είτε έγιναν διαθέσιμα σε εταιρείες ή οργανισμούς. Με το έναν ή τον άλλο τρόπο υπάρχει διαθέσιμος προς ανάλυση μεγάλος όγκος δεδομένων από τα οποία μπορεί να εξαχθεί γνώση που αν δεν υπήρχαν υπολογιστικοί πόροι δεν θα μπορούσαμε να ανακαλύψουμε και επίσης πολλά δεδομένα ώστε να αναπτυχθούν αλγόριθμοι που θα μάθουν από αυτά. Γενικά στην μηχανική μάθηση δεν έχουμε τους στόχους που έχει το general AI. Αντί για αυτό όπως και σε άλλους τομείς**[3,4]** της Τεχνητής Νοημοσύνης μας ενδιαφέρει να λύσουμε ένα πολύ συγκεκριμένο πρόβλημα. Γενικά ένα πρόβλημα όταν είναι απλό, είναι εύκολο να δημιουργηθεί ένα πρόγραμμα που να εφαρμόζει μία λύση. Για πολύπλοκα προβλήματα αντίθετα μπορεί να είναι πολύ δύσκολο έως αδύνατο ένας άνθρωπος να βρει μία λύση και να την υλοποιήσει σε έναν αλγόριθμο. Αντί για αυτό είναι πιο αποδοτικό ο υπολογιστής να δημιουργήσει έναν αλγόριθμο από παραδείγματα αντί να προσπαθούμε να βρούμε λύση για κάθε βήμα του αλγορίθμου. Όπως αναφέραμε και προηγουμένως η Μηχανική Μάθηση είναι υποκατηγορία της Τεχνητής Νοημοσύνης. Εκτός από αυτό σχετίζεται άμεσα και με άλλους κλάδους. Ένας από αυτούς είναι το Data Mining**[20]**. Σε αυτόν τον κλάδο χρησιμοποιούνται παρόμοιες τεχνικές αλλά για διαφορετικούς σκοπούς. Η μηχανική μάθηση έχει στόχο να γενικεύσει την γνώση ενώ στο data mining ο στόχος

είναι να ανακαλύψουμε προηγούμενα άγνωστα μοτίβα(patterns). Άλλος ένας κλάδος που σχετίζεται με την μηχανική μάθηση είναι η βελτιστοποίηση(optimization). Εδώ λόγω του τρόπου που ορίζεται ένα πρόβλημα μάθησης έχουμε μία συνάρτηση την οποία θέλουμε να ελαχιστοποιήσουμε ή να μεγιστοποιήσουμε. Επίσης η μηχανική μάθηση σχετίζεται με την θεωρία του generalization**[2]** καθώς εκτός από την ελαχιστοποίηση/μεγιστοποίηση μια συνάρτησης μας ενδιαφέρει και η απόδοση σε δεδομένα που δεν έχει ξαναδεί ένα μοντέλο μηχανικής μάθησης. Τέλος η μηχανική μάθηση σχετίζεται με την στατιστική. Αυτοί οι δύο κλάδοι σχετίζονται όσον αφορά εργαλεία που χρησιμοποιούν αλλα διαφέρουν στους στόχους τους οποίους έχουν. Η στατιστική έχει ως στόχο να παράξει γνώση για έναν πληθυσμό, ενώ η μηχανική μάθηση να γενικεύσει την γνώση.

Οι κλάδοι που αναφέραμε εδώ και οι σχέσεις τους με την Μηχανική Μάθηση δεν είναι σε καμία διάσταση εξαντλητικές. Η βιβλιογραφία που απαριθμεί κλάδους που σχετίζονται και τις σχέσεις μεταξύ τους είναι μεγάλη και δεν γίνεται να παρουσιαστεί σε αυτήν την εργασία. Παρόλα αυτά το σημείο κλειδί είναι ότι προκειται για διαδικασία που σχετίζεται με πολλούς κλάδους και δεν παρουσιάστηκε ξαφνικά και από το πουθενά.

2.2.2 Εφαρμογές Μηχανικής Μάθησης

Εκτός από την άμεση σχέση με άλλους επιστημονικούς κλάδους για τον τρόπο που η μηχανική μάθηση πετυχαίνει τον στόχο, δηλαδή την δημιουργία ενός μοντέλου που γενικεύει την γνώση και οι εφαρμογές της βρίσκουν χρησιμότητα σε πολλούς επιστημονικούς τομείς. Σε πολλούς από αυτούς τους τομείς η μηχανική μάθηση αποτελεί την μοναδική επιλογή καθώς δεν βρίσκουν εφαρμογή σε κάποια προβλήματα στατικές προκαθορισμένες λύσεις. Σε αυτή την ενότητα θα παρουσιάσουμε μερικές περιπτώσεις όπου η μηχανική μάθηση έδωσε λύσεις. Τα παραδείγματα που θα αναφέρουμε σαφώς δεν είναι εξαντλητικά και μπορούν να βρουν εφαρμογή σε άλλα παρόμοια domains.

- Κοινωνικά Δίκτυα(Social Media)

Η εφαρμογή αλγορίθμων μηχανικής μάθησης σε δεδομένα που προκύπτουν από χρήση σε κοινωνικά δίκτυα είναι πιθανόν μία από τις πιο διαδεδομένες εφαρμογές. Αλληλεπιδρώντας με άλλους χρήστες παράγουμε μεγάλο όγκο δεδομένων τα οποία αναλύονται στην συνέχεια για διάφορους σκοπούς. Πέρα από το προφανές όπου στόχος των εταιρειών είναι να κατηγοριοποιήσουν χρήστες με βάση τα ενδιαφέροντα τους ώστε να παραχθεί πληροφορία για πιο στοχευμένες διαφημίσεις που θα φέρουν μεγαλύτερα έσοδα στις εταιρείες υπάρχουν και άλλες χρήσεις. Για παράδειγμα μπορούν να προταθούν σε χρήστες άλλοι χρήστες ή ομάδες χρηστών με κοινά ενδιαφέροντα ή σελίδες όπου κάποιος μπορεί να βρεί πληροφορίες που τον ενδιαφέρουν. Επίσης όπως έχει γίνει κάποιες φορές στο παρελθόν τα κοινωνικά δίκτυα έχουν χρησιμοποιηθεί από

ενδιαφερόμενους για εξαγωγή πληροφορίας με στόχο να επηρεάσουν στοχευμένα ανθρώπους-ψηφοφόρους κλειδιά ώστε να επηρεάσουν εκλογικά αποτελέσματα μέσω στοχευμένης προπαγάνδας και ψευδών ειδήσεων. Ο συγκεκριμένος τομέας ο οποίος σχετίζεται με δεδομένα γράφων έχει γνωρίσει μεγάλη επιτυχία με τις εξελίξεις στον τομέα του Graph Representation Learning. Πριν από την εξέλιξη των GNNs**[14]** η πληροφορία από τους γράφους εξαγόταν και στην συνέχεια χρησιμοποιούνταν με τα υπόλοιπα χαρακτηριστικά με χρήση κάποιου shallow μοντέλου. Οι εξελίξεις στα GNNs επέτρεψαν την παραγωγή αναπαραστάσεων γνωστών και ως embendings, που κωδικοποιούν με πολύ αποτελεσματικό τρόπο πληροφορία σε χαμηλή διάσταση,σε σύγκριση με απλά χαρακτηριστικά και την εκπαίδευση μοντέλων.

- Σύσταση Προϊόντων(Product recommendation)

Σε αυτό το σενάριο χρήσης έχουμε παρόμοιους στόχους με το παραπάνω αλλά για πιο απλούς λόγους. Βασικό κομμάτι της εμπορικής δραστηριότητας είναι η πρόταση νέων προϊόντων σε πελάτες με βάση δικές τους προηγούμενες επιλογές ή επιλογές άλλων πελατών οι οποίοι έχουν παρόμοιο καταναλωτικό προφίλ με τον πελάτη στόχο. Όταν αυτό γίνεται αποτελεσματικά η εμπορική εταιρεία μπορεί να αυξήσει τις πωλήσεις. Επίσης μοντελοποιώντας τις καταναλωτικές συνήθειες μπορούν να δημιουργήσουν στοχευμένα σε ομάδες καταναλωτών πακέτα προσφορών ώστε και ο έμπορος να αυξήσει τον τζίρο αλλά και ο καταναλωτής να μειώσει το κόστος των επιλογών του.

- Αναγνώριση εικόνας(Image recognition)

Εδώ στόχος είναι να εξάγουμε πληροφορίες από δεδομένα εικόνας. Αυτό μπορεί να μας ενδιαφέρει για διάφορους λόγους. Μία εφαρμογή που έχει γίνει ευρέως διαθέσιμη είναι η αναγνώριση προσώπου. Εδώ είτε μας ενδιαφέρει να εντοπίσουμε ότι υπάρχουν πρόσωπα σε μία εικόνα είτε να εντοπίσουμε αν ένα πρόσωπο είναι αυτό που μας ενδιαφέρει. Για την πρώτη περίπτωση μία περίπτωση χρήσης είναι να βάλουμε ετικέτα σε ανθρώπους σε φωτογραφίες εύκολα και για την δεύτερη περίπτωση μία δημοφιλής εφαρμογή είναι η αυθεντικοποίηση ενός χρήστη. Η τελευταία έγινε δημοφιλής ώς ισχυρή τεχνική παρόλο που θεωρητικά δεν είναι. Ένα κλειδί που δεν αλλάζει είναι αδύναμο κλειδί.

- Αναγνώριση Συναισθήματος(Sentiment Analysis)

Πρόκειται για πρόβλημα**[21]** που έχει άμεση σχέση με τα προηγούμενα αλλά αποτελεί κατηγορία από μόνο του διότι έχει πολλές εφαρμογές σε πολλές περιπτώσεις. Μία χαρακτηριστική περίπτωση που χρησιμοποιείται είναι για αυτόματη αξιολόγηση κριτικών σε προϊόντα που έχουν αγοράσει καταναλωτές κάτι που θα ήταν απαγορευτικό έως αδύνατο να γίνει από ανθρώπους λόγω του μεγάλου όγκου πληροφορίας και του εργατικού κόστους. Φυσικά δεν περιορίζεται σε αυτό. Είναι επίσης χρήσιμο στην ανάλυση της επίδρασης που είχε στο κοινό μία δήλωση για παράδειγμα ενός πολιτικού. Σχετίζεται επίσης με την αναγνώριση εικόνας για ανάλυση συναισθήματος σε εικόνες/βίντεο.

- Μηχανική Μετάφραση(Machine Translation)

Πρόκειται για υποκατηγορία της επεξεργασίας φυσικής γλώσσας η οποία βρήκε εφαρμογή στην περίοδο του ψυχρού πολέμου με στόχο να μεταφέρει το νόημα κειμένων σε άλλες γλώσσες αυτόματα και με μεγάλη ακρίβεια, όπως για παράδειγμα επιστημονικών κειμένων. Παρόλο που υπήρχαν τεχνικές για να το φέρουν εις πέρας τα αποτελέσματα δέν ήταν καλά και επίσης υπήρχε ανάγκη για προεπεξεργασία σε μεγάλο βαθμό από τον χρήστη. Ο τομέας παρουσίασε μεγάλη βελτίωση όταν δημιουργήθηκαν συγκεκριμένες αρχιτεκτονικές νευρωνικών**[22]** δικτύων όπως το BERT**[9]**.

- Εφαρμογές στον τομέα της Υγείας(E-Health)

Πρόκειται για μία από τις πιο ενδιαφέρουσες εφαρμογές της Μηχανικής Μάθησης. Εδώ**[23]** στόχος είναι με χρήση έξυπνων συσκευών που επικοινωνούν να προβλέψουμε σε πραγματικό χρόνο πιθανά προβλήματα υγείας σε ασθενείς ώστε να γίνει έγκαιρη αντιμετώπιση τους. Αυτός ο τομέας είναι πολλά υποσχόμενος για το μέλλον και παρουσιάζει την πρόκληση του ότι το μοντέλο θα πρέπει να είναι πολύ αξιόπιστο δηλαδή να έχει πολύ μικρό ποσοστό false positives και μεγάλο recall.

Φυσικά υπάρχουν και πολλές άλλες περιπτώσεις χρήσης που θα μπορούσαμε να αναφέρουμε αλλά η λίστα είναι μεγάλη και δεν είναι σκοπός της εργασίας να αναφέρει τα πάντα με λεπτομέρεια. Η περίπτωση χρήσης που θα αναλύσουμε είναι η αυτόματη αξιολόγηση πληροφοριών για ανίχνευση κυβερνοεπιθέσεων.

2.2.3 Είδη και σκοπός της Μηχανικής Μάθησης

Όπως αναφέραμε και στην εισαγωγή ο σκοπός της Μηχανικής Μάθησης είναι να εξάγει γνώση από πραγματικά αντιπροσωπευτικά δεδομένα που έχουν συλλεγεί και με βάση αυτά να γενικεύσει γνώση. Σκοπός είναι να δημιουργηθεί ένα πρόγραμμα το οποίο παρουσιάζει γνωστικές ικανότητες παρόμοιες με των ανθρώπων και η ικανότητα του προγράμματος να γενικεύει την γνώση που απέκτησε σε καινούρια δεδομένα που βλέπει για πρώτη φορά. Πριν από αυτό τα προγράμματα είχαν την δυνατότητα απλά να εκτελούν αυτό που προγραμματιστηκαν να εκτελούν. Η γνώση που θα εξαχθεί μπορεί να διαφέρει για κάθε περίπτωση χρήσης. Φυσικά εκτός από την αποτελεσματικότητα και αποδοτικότητα των αλγορίθμων κεντρικό ρόλο παίζει η ποιότητα των δεδομένων που έχουμε στην διάθεση μας. Ένα dataset το οποίο για παράδειγμα έχει μετρήσεις οι οποίες δεν είναι ακριβής γενικά δεν θα δώσει σωστά αποτελέσματα άσχετα με όλες τις άλλες παραμέτρους. Να σημειωθεί ότι γενικά η δημιουργία ενός ποιοτικού dataset πολλές φορές δεν είναι απλή και φθηνή διαδικασία από την άποψη της διαθεσιμότητας και της ακρίβειας των μετρήσεων. Επίσης ένα ανοιχτό επιστημονικό θέμα που σχετίζεται με την ποιότητα του dataset είναι το adversarial robustness**[24]** για την διαδικασία της εκπαίδευσης αλλά και την εξαγωγή

αποτελεσμάτων**.** Όπως σε κάθε μέθοδο υπάρχουν διάφοροι τρόποι που μπορούμε να εκπαιδεύσουμε έναν αλγόριθμο ο καθένας με τα πλεονεκτήματα και τα μειονεκτήματα του. Για να καταλάβουμε τα υπέρ και τα κατά θα πρέπει πρώτα να δούμε τί δεδομένα καταναλώνει ένας αλγόριθμος. Γενικά μπορούμε να κατηγοριοποιήσουμε τις μεθόδους με βάση το είδος των δεδομένων που χρησιμοποιούν. Η πρώτη περίπτωση είναι να έχουμε δεδομένα με ετικέτες δηλαδή το κάθε δείγμα αποτελείται από χαρακτηριστικά(features) να συνοδεύεται από μία ή περισσότερες ετικέτες. Η δεύτερη περίπτωση είναι να έχουμε δεδομένα χωρίς ετικέτες. Επίσης μία περίπτωση που προκύπτει συχνά είναι να έχουμε ετικέτες αλλά να είναι πολύ λίγα τα δεδομένα που σχετίζονται με τις ετικέτες σε σχέση με το σύνολο. Οι κύριες κατηγορίες Μηχανικής Μάθησης με βάση αυτά είναι οι παρακάτω.

- Επιβλεπόμενη Μάθηση(Supervised Learning)

Πρόκειται για έναν από τους πιο απλούς τρόπους εκπαίδευσης ενός αλγορίθμου**[12]**. Στην περίπτωση αυτή έχουμε ένα σύνολο δεδομένων εκπαίδευσης το οποίο ο αλγόριθμος θα χρησιμοποιήσει για να εξάγει γνώση. Παρόλη την ευκολία και δύναμη που προσφέρουν τα δεδομένα με ετικέτες πρέπει να ξανα αναφέρουμε ότι η επιτυχία βασίζεται στην ποιότητα και ακρίβεια της διαδικασίας του annotation για το οποίο υπάρχει μεθοδολογία και αποτελει από μονο του ξεχωριστό πρόβλημα**[25]**. Σαν παράδειγμα προβλημάτων αναφέρουμε την διαφωνία μεταξύ annotators και την περίπτωση ανίχνευσης λάθος annotation. Πέρα από το συγκεκριμένο πρόβλημα η διαδικασία εξελίσσεται με τον αλγόριθμο κατα την φάση της εκπαίδευσης να βρίσκει πολύπλοκα μοτίβα μεταξύ των δειγμάτων τα οποία δεν είναι δυνατόν να βρεθούν χωρίς τους υπολογιστικούς πόρους τα οποία σχετίζει με τις δοσμένες ετικέτες και δημιουργεί ένα μοντέλο για να εξηγήσει αυτές τις συσχετίσεις. Φυσικά αυτό αποτελεί έναν γενικό ορισμό και ο κάθε αλγόριθμος χρησιμοποιεί διαφορετικές τακτικές για να λύσει το πρόβλημα. Παρόλη την δύναμη της τεχνικής αυτής μπορούν να παρουσιαστούν προβλήματα με πιο χαρακτηριστικά αυτά του overfitting της έλλειψης επαρκών δεδομένων εκπαίδευσης, του θορύβου στα δεδομένα , και του curse of dimensionality για να αναφέρουμε τα πιο γνωστά. Σε αυτά τα προβλήματα θα αναφερθούμε στην συνέχεια. Τέλος μετά την επιτυχή εκπαίδευση του μοντέλου ακολουθεί η διαδικασία της ανάπτυξης(deployment) του συστήματος. Αυτό το τελευταίο στάδιο είναι πολύ σημαντικό και μπορεί να καθορίσει την επιτυχία η την αποτυχία του project. Παρόλο που μπορεί να έχει πετύχει κάποιος αποδεκτά αποτελέσματα όσον αφορά μετρικές επίδοσης μπορεί το μοντέλο να έχει την απαίτηση να λειτουργεί real time οπότε αν ο χρόνος επεξεργασίας ξεπεράσει τον ρυθμό που φτάνουν δεδομένα τότε το μοντέλο δεν λειτουργεί σωστά. Επίσης το overhead σε υπολογιστικούς πόρους μπορεί να είναι σημαντική παράμετρος. Αυτό ισχύει ειδικά στην περίπτωση deep μοντέλων νευρωνικών δικτύων που έχουν την απαίτηση να λειτουργούν στο edge του δικτύου ή σε κινητές συσκευές η σε microcontrollers. Τέλος είναι αυτονόητο ότι το μοντέλο πρέπει να τροφοδοτείται με δεδομένα ίδιου τύπου με αυτά που εκπαιδεύτηκε. Το αν τελικά είναι εφικτο και εύκολο να βρούμε αυτά τα δεδομένα είναι ένα θέμα που

εξετάζεται ξεχωριστά οπότε πολλές φορές η επιλογή χαρακτηριστικών(feature selection) είναι προαπαιτούμενο.

- Μη Επιβλεπόμενη Μάθηση(Unsupervised Learning)

Η μέθοδος της μη επιβλεπόμενης μάθησης έχει το πλεονέκτημα του ότι δεν χρειάζεται την ακριβή διαδικασία του annotation. Αντί να συσχετίζει χαρακτηριστικά των δειγμάτων με ετικέτες στόχο έχει να βρει μοτίβα που χαρακτηρίζουν τα δείγματα χρησιμοποιώντας τις έννοιες της ομοιότητας και της απόστασης. Με αυτόν τον τρόπο βρίσκει παρόμοια χαρακτηριστικά που είναι αντιπροσωπευτικά των δειγμάτων με βάση τα οποία μπορεί να διαχωρίσει τα δεδομένα σε ομάδες. Η διαδικασία που περιγράφηκε είναι η πλέον αντιπροσωπευτική και ονομάζεται συσταδοποίηση(clustering)**[26]**. Η διαδικασία της συσταδοποίησης είναι μία ενότητα από μόνη της και θα μπορούσαμε να πούμε πολλά περισσότερα αλλά δεν είναι σκοπός της εργασίας. Γενικά πάντως αποτελεί την πρώτη λύση στην διαδικασία exploratory data analysis. Όσο για το πλήθος των αλγορίθμων**[27]** συσταδοποίησης αποτελείται από δεκάδες έως εκατοντάδες. Παρόλο που η ομαδοποίηση των δεδομένων είναι η προφανής εφαρμογή αυτής της μεθόδου μπορεί να χρησιμοποιηθεί σε προβλήματα classification, anomaly detection, outlier detection όπου εμέσως κατηγοριοποιεί ένα δείγμα ώς συνηθισμένο ή απόκλιση στο τομέα της ανίχνευσης απάτης σε συναλλαγές ή στην ανίχνευση μη συνηθισμένης συμπεριφοράς ενός κόμβου σε ένα δίκτυο. Άλλες χρήσεις είναι η εύρεση κοινοτήτων σε δεδομένα γράφων ή και η εύρεση σχέσεων μεταξύ κόμβων γράφων για παράδειγμα σε δεδομένα καταναλωτών-προϊόντων γνωστό ως link prediction**[28]**. Τέλος αξίζει να κάνουμε αναφορά στην εφαρμογή της μείωσης διαστάσεων(dimensionality reduction)**[29]**. Σε αυτή την χρήση και με άμεση σχέση με το πρόβλημα μείωσης διάστασης υπάρχει η δυνατότητα να αναπαραστήσουμε τα δεδομένα σε πολύ μικρότερες διαστάσεις από τις αρχικές χάνοντας ελεγχόμενο μέγεθος πληροφορίας καθώς και για σκοπούς οπτικοποίησης(visualization). Αυτό μπορεί να επιτευχθεί χωρίς επίβλεψη με πιο γνωστή μέθοδο την PCA**[29]**. Η μέθοδος αυτή έχει ως στόχο να μειώσει την επανάληψη της πληροφορίας καθώς και να συμπιέσει το dataset. Αποτελεί γραμμικό μετασχηματισμό στα δεδομένα και αναπαράσταση σε νέο χώρο όπου τα principal components είναι ορθογώνια. Με αυτό τον τρόπο κρατώντας το πρώτο principal component, το οποίο έχει το μέγιστο variance δηλαδή φέρει την μεγαλύτερη πληροφορία, έχουμε μεγάλο μέρος της πληροφορίας σε μία διάσταση. Στην συνέχεια κρατώντας το δεύτερο προσθέτουμε το μεγαλύτερο μέρος της εναπομείνασας πληροφορίας και ούτω καθεξής. Άλλη μία μέθοδος μείωσης της διάστασης είναι η SVD**[29]**. Τέλος αξίζει να αναφέρουμε τα νευρωνικά δίκτυα Autoencoders**[30]**. Η μέθοδος αυτή λειτουργεί ως εξής. Πρώτα ο encoder μειώνει την διάσταση της εισόδου περνώντας την από ένα feed forward δίκτυο μέχρι να φτάσει στο ενδιάμεσο επίπεδο(bottleneck) το οποίο αποτελεί την αναπαράσταση σε μικρότερη διάσταση. Η αναπαράσταση αυτή στην συνέχεια περνάει στον decoder ο οποίος προσπαθεί να ανακατασκευάσει την αρχική πληροφορία. Το αποτέλεσμα μετά την

εκπαίδευση είναι να έχουμε έναν encoder ο οποίος παράγει σωστές αναπαραστάσεις σε μικρή διάσταση.

- Ενισχυτική Μάθηση(Reinforcement Learning)

Η μέθοδος αυτή**[31]** είναι η τρίτη μεγαλύτερη κατηγορία Μηχανικής Μάθησης. Εδώ στόχος είναι μέσω της διαδικασίας της επιλογής μίας ενέργειας σε μία κατάσταση και της ανταπόκρισης του περιβάλλοντος με μία νέα κατάσταση και μία ανταμοιβή να εκπαιδευτεί ένας ευφυής πράκτορας να λειτουργεί σε ένα περιβάλλον με στόχο να μεγιστοποιήσει την ανταμοιβή που θα αποκομίσει μακροπρόθεσμα. Συνήθως το περιβάλλον μοντελοποιείται ως ένα MDP**[31]**. Τα MDPs είναι χρήσιμα για την επίλυση προβλημάτων μεγιστοποίησης μέσω δυναμικού προγραμματισμού**[31]**. Σε αυτή την μοντελοποίηση ο έλεγχος του περιβάλλοντος βασίζεται και στην τύχη δηλαδή το περιβάλλον είναι στοχαστικό αλλά και εν μέρη υπό τον έλεγχο του πράκτορα που παίρνει την επιλογή. Η νέα κατάσταση και η ανταμοιβή εξαρτώνται από την προηγούμενη κατάσταση και την επιλογή ενέργειας που επιλέγει ο πράκτορας. Βασική στρατηγική αποτελεί η μέθοδος exploration-exploitation**[31]**. Σε αρχικό στάδιο ο πράκτορας γνωρίζει ελάχιστα για το περιβάλλον, έτσι επιλέγει με μεγάλη πιθανότητα ενέργειες τυχαία. Όσο εξελίσσεται η διαδικασία και ο πράκτορας μαθαίνει ακολουθεί την greedy επιλογή δηλαδή επιλέγει εκείνη την ενέργεια που μεγιστοποιεί το κέρδος του. Βασικό ρόλο στην μέθοδο αυτή αποτελούν η συνάρτηση Value Function**[31]** η οποία αναπαριστά την αξία του να βρίσκεται σε μία κατάσταση και η συνάρτηση Q**[31]** η οποία αναπαριστά την αξία μιας συγκεκριμένης ενέργειας σε μία δεδομένη κατάσταση. Στόχος της διαδικασίας είναι να προσεγγιστεί η συνάρτηση Q ή η βέλτιστη πολιτική. Η μέθοδος αυτή ήταν αρχικά περιορισμένη σε περιβάλλοντα μικρής διάστασης αλλά με την έλευση των νευρωνικών δικτύων και συγκεκριμένα των Deep Q Networks**[31]** έγινε εφικτή η χρήση της μεθόδου για χώρους μεγάλης διάστασης όπως εικόνες. Δεν θα πούμε περισσότερα για αυτήν την μέθοδο διότι δεν θα την χρησιμοποιήσουμε σε αυτήν την εργασία.

2.2.4 Αλγόριθμοι Μηχανικής Μάθησης(Shallow)

Υπάρχουν πολλοί αλγόριθμοι μηχανικής μάθησης που κατατάσσονται στην κατηγορία shallow ML. Γενικά η επιβλεπόμενη μάθηση μπορεί να κατηγοριοποιηθεί σε regression/classification προβλήματα**[12]**. Σε αυτήν την ενότητα θα παρουσιάσουμε τους πιο γνωστούς που ταιριάζουν στο πρόβλημα που θέλουμε να λύσουμε δηλαδή το πρόβλημα του classification. Ο κάθε αλγόριθμος έχει τα υπέρ και τα κατά και η επιλογή βασίζεται πάντα στην περίπτωση χρήσης και στο διαθέσιμο dataset. Η αξιολόγηση της καταλληλότητας του καθενός δεν είναι κάτι που μπορεί γενικά να αξιολογηθεί θεωρητικά. Γενικά συνήθως αφού κάνουμε όλη την απαραίτητη προ επεξεργασία στα δεδομένα δοκιμάζουμε πολλούς αλγορίθμους πραγματοποιώντας παράλληλα την διαδικασία της

ρύθμισης των υπερ παραμέτρων και επιλέγουμε εκείνον τον αλγόριθμο που έχει καλύτερα αποτελέσματα με βάση το τι μας ενδιαφέρει. Τα κριτήρια μπορεί να είναι διάφορα μετρικά όπως accuracy recall precision ή μπορεί να είναι το πόσο γρήγορα κάνει προβλέψεις το μοντέλο ή το πόσο εύκολο είναι να γίνει incremental learning**[32]** άν αυτό είναι προαπαιτούμενο. Σε αυτά θα αναφερθούμε αναλυτικά στην συνέχεια. Γενικά μπορούμε να κατηγοριοποιήσουμε τους αλγορίθμους σε regression και classification**[12]**. Στην περίπτωση του regression στόχος είναι να προβλέψουμε μία συνήθως συνεχή τιμή όπως πχ την τιμή ενός σπιτιού με βάση τα χαρακτηριστικά του. Στην περίπτωση του classification στόχος είναι να προβλέψουμε την κατηγορία στην οποία ανήκει ένα δείγμα. Σε αυτήν την ενότητα θα παρουσιάσουμε συνοπτικά τους πιο γνωστούς αλγορίθμους classification. Να υπενθυμίσουμε ότι η περίπτωση χρήσης αυτής της εργασίας είναι το classification σε νόμιμη δραστηριότητα ή κυβερνοεπίθεση.

- Logistic Regression

Παρά το όνομα του αλγορίθμου**[33]** αυτού συνήθως αναφέρεται σε προβλήματα classification. Με χρήση της σιγμοειδούς συνάρτησης το αποτέλεσμα που δίνει είναι μία τιμή που αναφέρεται στην πιθανότητα να ανήκει ένα datapoint στη μία κλάση ή στην άλλη με βάση τα χαρακτηριστικά του δείγματος. Υπάρχουν τρία είδη logistic regression.

Binary Logistic Regression[33]. Πρόκειται για την πιο συνηθισμένη περίπτωση. Εδώ έχουμε την κατηγοριοποίηση με βάση ένα όριο(threshold) το οποίο είναι συνήθως το 0.5 σε binary προβλήματα όπως το αν ένα email είναι spam ή όχι ή αν η δραστηριότητα είναι νόμιμη ή πρόκειται για κυβερνοεπίθεση. Πρόκειται για έναν από τους πιο συνηθισμένους binary classifiers.

Multinomial logistic regression[34]. Σε αυτό το είδος logistic regression η εξαρτώμενη μεταβλητή παίρνει τρείς ή παραπάνω τιμές αλλά δεν υπάρχει κάποια σειρά ή ιεραρχία ανάμεσα τους. Πιο συγκεκριμένα δεν υπάρχει κάποιο νόημα στην ιεραρχία τους. Αποτελούν απλά διαφορετικές κατηγορίες. Είναι μία επέκταση της πρώτης περίπτωσης σε περισσότερες από δύο κατηγορίες.

Ordinal Logistic Regression[34]. Σε αυτήν την περίπτωση έχουμε πάλι τρείς ή παραπάνω κατηγορίες αλλά η διαφορά με την προηγούμενη περίπτωση είναι ότι εδώ υπάρχει κάποια ιεραρχική σχέση μεταξύ των κατηγοριών όπως πχ στην περίπτωση rating σε κλίμακα 1-5.

- Decision Tree

Αυτός ο αλγόριθμος**[35]** μπορεί να χρησιμοποιηθεί και για classification αλλά και για regression προβλήματα μηχανικής μάθησης. Συνήθως προτιμάται για classification. Η δομή του μοιάζει με αυτή ενός δέντρου. Σε αυτόν τον αλγόριθμο υπάρχουν δύο είδη κόμβων. Ο πρώτος κόμβος ονομάζεται κόμβος απόφασης και ελέγχει σε κάθε βήμα ένα feature από το dataset. Ο δεύτερος κόμβος ονομάζεται κόμβος φύλο και αναπαριστά την τελική απόφαση της διαδικασίας. Οι διακλαδώσεις του δέντρου αναπαριστούν την απόφαση για κάθε κόμβο απόφασης δηλαδή για το αν ικανοποιείται ή οχι η συνθήκη. Όταν φτάσουμε σε κόμβο φύλο ο αλγόριθμος τερματίζει. Κάποια από

τα πλεονεκτήματα του αλγορίθμου είναι το ότι μιμείται τον τρόπο που σκέφτονται οι άνθρωποι και είναι πολύ εύκολο να καταλάβουμε/αιτιολογήσουμε γιατί πάρθηκε μία απόφαση. Με βάση αυτά προκύπτει το ερώτημα πώς διαλέγουμε ποια από τα χαρακτηριστικά είναι τα καλύτερα ώστε να τα εξετάσουμε στην αρχή της διαδικασίας και πια από αυτά μπορούμε να αγνοήσουμε τελείως ή να τους δώσουμε μικρή σημασία. Γενικά αυτή η διαδικασία ονομάζεται Attribute Selection Measure(ASM)**[36]**. Υπάρχουν δύο μέθοδοι για αυτό. Η πρώτη μέθοδος ονομάζεται **Information Gain[37]** και λειτουργεί ως εξής. Information Gain ονομάζεται ένα μέτρο στην αλλαγή της εντροπίας μετά τον διαχωρισμό του dataset με βάση ένα feature. Υπολογίζει το κέρδος σε πληροφορία που θα έχουμε αν διαχωρίσουμε με ένα συγκεκριμένο feature συνεπώς διαλέγουμε στους πρώτους κόμβους τα χαρακτηριστικά εκείνα που δίνουν την μέγιστη πληροφορία. Η δεύτερη μέθοδος ονομάζεται Gini Index**[37]**. Είναι ένα ακόμη μέτρο για επιλογή χαρακτηριστικών. Εδώ διαλέγουμε τα χαρακτηριστικά εκείνα με το χαμηλότερο Gini Index. Τέλος να αναφέρουμε την διαδικασία του pruning**[35]**. Σε αυτή την διαδικασία διαγράφουμε κόμβους από το δέντρο για να έχουμε το βέλτιστο δέντρο. Στην περίπτωση που έχουμε ένα πολύπλοκο δέντρο αυξάνεται ο κίνδυνος να έχουμε overfitting**[38]** οπότε στόχος είναι να μικρύνει το δέντρο χωρίς να χάσουμε κάτι στην ακρίβεια. Υπάρχουν δύο τεχνικές για να γίνει αυτό. Η πρώτη ονομάζεται Cost Complexity Pruning**[39]** και η δεύτερη Reduced Error Pruning**[40]**. Γενικά τα πλεονεκτήματα του αλγορίθμου αυτού είναι ότι μπορεί να δουλέψει και με categorical και continual δεδομένα. Συνήθως δεν χρειάζεται scaling και κανονικοποίηση των δεδομένων και είναι πιο ανθεκτικός στις περιπτώσεις που τα δεδομένα προς κατηγοριοποίηση είναι ελλιπή. Το πιο σημαντικό από τα αρνητικά είναι ότι τείνει να κάνει overfit όπως αναφέραμε και προηγουμένως καθώς και το ότι χρειάζεται όλο το dataset να βρίσκεται στην μνήμη δηλαδή δεν εκπαιδεύεται σε mini batches . Για να αντιμετωπιστεί το πρόβλημα του overfitting προτείνεται ο επόμενος αλγόριθμος.

- Random Forest

Αυτός ο αλγόριθμος**[41]** ανήκει στην κατηγορία ensemble models**[42]**. Σε αυτήν την κατηγορία αλγορίθμων συνδυάζουμε πολλούς classifiers με στόχο να έχουμε στο τέλος μία ψηφοφορία(majority vote) κάτι που έχει αποδειχθεί ότι μειώνει την πιθανότητα overfitting και αυξάνει την ακρίβεια του μοντέλου. Η περίπτωση της ψηφοφορίας δεν είναι πάντα επιθυμητή και υπάρχουν άλλες προσεγγίσεις.Υπάρχουν τρεις προσεγγίσεις για το πως μπορούμε να συνδυάσουμε τους classifiers**[42]**. Η πρώτη μέθοδος ονομάζεται **Bagging** και λειτουργεί ώς εξής. Εκπαιδεύουμε παράλληλα πολλούς ίδιους classifiers αλλά με διαφορετικά δείγματα τον καθένα και στο τέλος τους συνδυάζουμε ώστε να παραχθεί το τελικό αποτέλεσμα. Για το πώς θα συνδυαστούν υπάρχουν πολλοί μέθοδοι και η επιλογή εξαρτάται από άλλους παράγοντες. Η δεύτερη μέθοδος ονομάζεται **Boosting.** Σε αυτήν την μέθοδο εκπαιδεύουμε σειριακά classifiers με στόχο ο επόμενος classifier να εκπαιδευτεί να κατηγοριοποιεί τα λάθη του προηγούμενου. Τέλος η τρίτη μέθοδος ονομάζεται **Stacking.** Σε

αυτήν την μέθοδο εκπαιδεύουμε παράλληλα classifiers και τους συνδυάζουμε ώστε να εκπαιδεύσουμε ένα μετα μοντέλο. Με βάση αυτά ο αλγόριθμος Random Forest ορίζεται ώς ο συνδυασμός με μία από τις τρεις μεθόδους και τον αλγόριθμο Decision Tree. Φυσικά η μέθοδος των ensemble models δεν περιορίζεται στα Decision Trees.

- SVM

Ο αλγόριθμος αυτός**[43]** είναι πολύ δημοφιλής και βασίζεται στην υπόθεση ότι τα δεδομένα είναι γραμμικά διαχωρίσιμα. Στόχος τους SVM είναι να δημιουργήσει το καλύτερο δυνατό υπερ επίπεδο που διαχωρίζει τα δεδομένα. Ένα σημαντικό σημείο είναι τι σημαίνει ιδανικό σε αυτήν την περίπτωση. Με βάση τον ορισμό στόχος είναι να εκπαιδευτούν οι παράμετροι του μοντέλου έτσι ώστε το όριο του υπερ επιπέδου για την χειρότερη περίπτωση να είναι όσο το δυνατόν μεγαλύτερο ή πιο σωστά απομακρυσμένο και όλα τα άλλα σημεία του dataset να έχουν μεγαλύτερη απόσταση από αυτό. Με βάση αυτά το βέλτιστο αυτό όριο καθορίζεται από συγκεκριμένα δείγματα τα οποία ονομάζονται support vectors. Αν όμως μας ενδιαφέρει η περίπτωση του incremental learning σε μία συγκεκριμένη περίπτωση χρήσης πρέπει να λάβουμε υπόψη ότι αν αλλάξουν τα support vectors πρέπει να αλλάξουν και οι παράμετροι του μοντέλου πράγμα που δυσκολεύει την διαδικασία σε μια περίπτωση που έχουμε ροή νέων δεδομένων. Η περίπτωση που περιγράφηκε ονομάζεται hard margin SVM**[43]** διότι έχει τον περιορισμό που αναφέρθηκε για το γεωμετρικό όριο. Στην βιβλιογραφία αναφέρεται και η περίπτωση του soft margin SVM**[43]**. Εδώ χαλαρώνεται ο περιορισμός και δίνεται η δυνατότητα σε κάποια δείγματα να βρίσκονται εντός του ορίου που ορίζουν τα support vectors. Αυτό χρησιμεύει στην περίπτωση όπου τα δεδομένα είναι διαχωρίσιμα αλλά μόνο μερικά δείγματα από αυτά δεν μπορούν να διαχωριστούν. Στην αρχή αυτής της ενότητας αναφέραμε ότι υπάρχει η υπόθεση ότι τα δεδομένα είναι γραμμικά διαχωρίσιμα. Τί συμβαίνει όμως για τις περιπτώσεις που δεν είναι. Σε αυτήν την περίπτωση στην βιβλιογραφία αναφέρεται το ότι μπορούμε να βρούμε έναν χώρο μεγαλύτερης διάστασης από τον αρχικό όπου είναι γραμμικά διαχωρίσιμα και να τρέξουμε τον αλγόριθμο σε αυτόν τον χώρο. Μία τεχνική που λύνει αυτό το πρόβλημα είναι γνωστή ως kernel trick και πετυχαίνει ακριβώς αυτό χωρίς να προβάλει τα δεδομένα σε χώρο μεγαλύτερης διάστασης.

- K-Nearest Neighbour

Σε αυτόν τον αλγόριθμο**[44]** κάθε δείγμα αναπαρίσταται σε χώρο n διαστάσεων όπου n είναι ο αριθμός των χαρακτηριστικών. Έτσι για να βρούμε την κλάση που ανήκει ένα άγνωστο δείγμα υπολογίζουμε την απόσταση του από όλα τα δείγματα του dataset και με βάση την κλάση που ανήκουν οι κοντινότεροι κ γείτονες επιλέγουμε την κλάση που προκύπτει περισσότερες φορές. Ανήκει στην κατηγορία των nonparametric αλγορίθμων**[45]**. Τα θετικά αυτής της μεθόδου είναι ότι δεν χρειάζεται εκπαίδευση, είναι ιδανική για incremental learning δεδομένου ότι μπορούμε ανα πάσα στιγμή να προσθέσουμε καινούρια δεδομένα. Τα αρνητικά είναι ότι δεδομένου ότι πρέπει να υπολογίσουμε τις αποστάσεις από όλα τα δείγματα και ο χρόνος που χρειάζεται για προβλεψη μπορεί

να γίνει απαγορευτικός για πραγματικού χρόνου σενάρια χρήσης όπως αυτό που εξετάζουμε σε αυτήν την εργασία. Τέλος η επιλογή της τιμής της υπερ παραμέτρου δεν είναι προφανής.

- Naive Bayes

Αλγόριθμοι όπως ο logistic regression ανήκουν στην κατηγορία των discriminative**[46]** αλγορίθμων. Αυτοί οι αλγόριθμοι μαθαίνουν μία συνάρτηση που διαχωρίζει τα δεδομένα στο χώρο. Ο αλγόριθμος Naive Bayes**[47]** ανήκει στην κατηγορία των generative αλγορίθμων**[46]**. Σε αυτήν την περίπτωση μοντελοποιούνται οι κλάσεις ξεχωριστά και για κάθε κλάση υπολογίζουμε το p(x|y) και το p(y).Στην συνέχεια για κάθε νέο δείγμα και με χρήση του κανόνα του Bayes υπολογίζουμε την πιθανότητα να ανήκει σε κάθε κλάση και επιλέγουμε την κλάση με την μεγαλύτερη πιθανότητα. Ο αλγόριθμος ονομάζεται Naive διότι κάνει την υπόθεση ότι τα δείγματα είναι μεταξύ τους ανεξάρτητα. Στις περιπτώσεις όπου δεν ισχύει κάτι τέτοιο σε μεγάλο βαθμό είναι καλύτερα να επιλέξουμε άλλο αλγόριθμο. Για τις περιπτώσεις όμως που ισχύει ο Naive Bayes δίνει πολύ καλά αποτελέσματα. Το μεγαλύτερο πλεονέκτημα του αλγορίθμου αυτού είναι το ότι μπορεί να δουλέψει με πολύ λίγα δεδομένα εκπαίδευσης.

2.2.5 Βήματα διαδικασίας Μάθησης

Σε αυτήν την ενότητα θα παρουσιάσουμε τα βήματα που ακολουθούνται γενικά για την εκπαίδευση ενός μοντέλου καθώς και κάποια πρακτικά ζητήματα που προκύπτουν όταν προετοιμάζουμε ένα μοντέλο. Τα βήματα αυτά γενικά μπορούν να χωριστούν σε τέσσερις ενότητες όπως θα τις παρουσιάσουμε στην συνέχεια αν και συνήθως παρουσιάζονται με διαφορετικούς τρόπους/ορολογία όπως εδώ**[48]** και εδώ**[49].**

- Συλλογή και προετοιμασία(Get/Prepare)

Σε πρώτο στάδιο βασικό είναι να υπάρχουν δεδομένα και να δημιουργηθεί ένα dataset επαρκές για εκπαίδευση. Αυτό το βήμα πολλές φορές δεν είναι απλό καθώς στην πλειοψηφία των σεναρίων χρήσης χρειαζόμαστε πολλά δεδομένα τα οποία πρέπει να είναι annotated. Επίσης πολλές φορές χρειάζονται δεδομένα τα οποία είναι προσωπικού χαρακτήρα ή αφορούν ιατρικά δεδομένα των οποίων η χρήση είναι νομικά δεσμευτική ή περιορισμένη. Στην πολύ συχνή περίπτωση που δεν έχουμε annotated δεδομένα η διαδικασία για να γίνουν annotated μπορεί να είναι χρονοβόρα, δύσκολη δηλαδή η διαδικασία να μην είναι απλή με χαρακτηριστικό παράδειγμα το sentiment analysis. Επίσης σε πολλές περιπτώσεις υπάρχουν πολλοί annotators και υπάρχει διαφωνία μεταξύ τους και πρέπει να υπάρχουν διαδικασίες ώστε να βρίσκονται οι καλύτερες λύσεις. Επίσης υπάρχει πάντα το ενδεχόμενο ένας annotator να μην κάνει καλά την δουλειά του κάτι που θα πρέπει να ανιχνευθεί. Επίσης για την περίπτωση όπου τα δεδομένα έχουν συλλεγεί και έχουν γίνει annotated από χρήστες όπως για παράδειγμα συμπληρώνοντας φόρμες υπάρχει η πιθανότητα ελλιπών

δεδομένων ή δεδομένων που έχουν λάθος καταχωρήσεις όπως πχ λάθος ημερομηνία γέννησης. Σε αυτές τις περιπτώσεις θα πρέπει να αναγνωριστούν αυτά τα δείγματα και να διαγραφούν ή να συμπληρωθούν με κατάλληλες τιμές. Αυτή η διαδικασία αναφέρεται συχνά ώς καθαρισμός(clean up). Τέλος τα δεδομένα που συλλέχθηκαν με κάποιο τρόπο σπανίως είναι σε μορφή κατάλληλη για να τροφοδοτηθούν σε έναν αλγόριθμο μηχανικής μάθησης. Τα δεδομένα πρέπει πολλές φορές να προ επεξεραστουν ώστε να μπορούν να είναι χρήσιμα για τον αλγόριθμο. Χαρακτηριστικό παράδειγμα αποτελούν τα categorical δεδομένα τα οποία πρέπει να έρθουν σε κατάλληλη μορφή. Για αυτήν την περίπτωση μπορούν να χρησημοποιηθούν μέθοδοι όπως one hot encoding**[50]** ή Label Encoding**[50]**. Επίσης βασικό ρόλο παίζει το scaling. Επειδή διάφορα χαρακτηριστικά έχουν διαφορετικό εύρος τιμών, αυτό δημιουργεί προβλήματα για κάποιους αλγορίθμους επειδή κατα την διάρκεια της εκπαίδευσης δίνεται μεγαλύτερη σημασία σε features με μεγαλύτερο εύρος τιμών. Το scaling, για παράδειγμα το max/min φέρνει τις τιμές στο ίδιο εύρος.

- Ανάλυση(Exploration)

Σε αυτό το βήμα σκοπός είναι να πάρουμε μια πρώτη εικόνα για τα δεδομένα του dataset. Αρχικά μπορούμε με κατάλληλα γραφήματα να βρούμε το εύρος τιμών ή τις κατανομές μεταβλητών. Στην συνέχεια με χρήση ιστογραμμάτων μπορούμε να βρούμε συσχετίσεις μεταξύ ανεξάρτητων μεταβλητών ή/και εξαρτήσεις μεταξύ εξαρτημένων και ανεξάρτητων μεταβλητών κάτι που θα βοηθήσει στην συνέχεια στην διαδικασία του feature engineering και feature selection. Η φάση του feature engineering**[51]** αφορά την εξαγωγή από τα μη επεξεργασμένα δεδομένα(raw data) χαρακτηριστικών που έχουν νόημα για το πρόβλημα που καλούμαστε να λύσουμε. Πολλές φορές σε αυτή την διαδικασία συνδυάζουμε δεδομένα από τα μη επεξεργασμένα δεδομένα και προκύπτουν χαρακτηριστικά που μπορούν να είναι χρήσιμα από τον αλγόριθμο. Στην φάση του feature selection σκοπός είναι να διαλέξουμε τα χαρακτηριστικά εκείνα που δίνουν την μεγαλύτερη πληροφορία ώστε να μειώσουμε το πλήθος τους. Αυτό πολλές φορές βοηθάει στο να αποφύγουμε το overfitting**[38]**, επίσης το μοντέλο γίνεται πιο απλό και ο χρόνος πρόβλεψης θεωρητικά μειώνεται. Η πιο απλή και αποτελεσματική τεχνική είναι να δούμε αν δύο ανεξάρτητες μεταβλητές είναι συσχετισμένες και να μην χρησιμοποιήσουμε την μία ή να δούμε αν μία ανεξάρτητη μεταβλητή είναι ασυσχέτιστη με την εξαρτημένη, σε αυτήν την περίπτωση η ανεξάρτητη είναι απλά θόρυβος. Επίσης μία βασική παράμετρος που πρέπει να δούμε σε αυτό το βήμα είναι αν το classification πρόβλημα που αντιμετωπίζουμε είναι imbalanced και πόσο. Imbalanced**[52]** dataset σημαίνει ότι οι κλάσεις δεν έχουν τον ίδιο αριθμό δειγμάτων.

- Δημιουργία Μοντέλου(Model)

Σε αυτό το βήμα στόχος είναι η δημιουργία του μοντέλου που θα κάνει την πρόβλεψη. Θεωρώντας ότι έχουν πραγματοποιηθεί τα προηγούμενα βήματα και αφού επιλέξουμε τον κατάλληλο αλγόριθμο για το πρόβλημα με βάση κριτήρια που θα αναλύσουμε στην συνέχεια διότι

αποτελούν μία ενότητα από μόνα τους και ακολουθώντας την διαδικασία εκπαίδευσης που θα αναλύσουμε στην συνέχεια δημιουργούμε και αξιολογούμε το μοντέλο.

- Συμπεράσματα και αξιολόγηση(Communicate)

Σε αυτό το τελευταίο βήμα και με βάση τα αποτελέσματα που έχουμε από την αξιολόγηση του μοντέλου από το προηγούμενο βήμα αναλύουμε την περίπτωση χρήσης και τις απαιτήσεις(requirements) που έχουν θέσει τα ενδιαφερόμενα μέρη και αποφασίζουμε αν η επίδοση του μοντέλου είναι μέσα στα πλαίσια του αποδεκτού. Στην περίπτωση που δεν είναι πρέπει να αξιολογήσουμε αν υπάρχουν και ποιοι είναι οι τρόποι ώστε να ικανοποιηθούν οι απαιτήσεις ή η απόδοση κάνοντας αλλαγές στο μοντέλο. Επίσης πρέπει να περιγράψουμε την μεθοδολογία της προσέγγισης στα ενδιαφερόμενα μέρη καθώς και τι σημαίνει στην πράξη ένα μετρικό από αυτά που χρησιμοποιούμε για να κάνουμε την αξιολόγηση διότι πολλές φορές αυτό δεν είναι σαφές και ακριβές όπως για παράδειγμα η ακρίβεια(accuracy) του μοντέλου είναι ένα μετρικό που μπορεί να μην είναι αντιπροσωπευτικό της πραγματικής απόδοση όπως θα δούμε παρακάτω.

2.2.6 Εκπαίδευση και Αξιολόγηση μοντέλου

Σε αυτήν την ενότητα θα περιγράψουμε αναλυτικα την διαδικασία της εκπαίδευσης ενός μοντέλου για την περίπτωση της επιβλεπόμενης μάθησης. Θεωρώντας ότι έχουν πραγματοποιηθεί τα βήματα του feature engineering και feature selection αρχικά πρέπει να φέρουμε το dataset σε μορφή δύο δομών δεδομένων. Η πρώτη θα περιέχει τα χαρακτηριστικά(features) που έχουν επιλεχθεί για όλα τα δείγματα που έχουμε στην διάθεση μας. Η δεύτερη δομή δεδομένων είναι αυτή που έχει τις εξαρτώμενες μεταβλητές για κάθε δείγμα. Στην συνέχεια σπάμε τα δεδομένα σε τρία διαφορετικά μέρη**[53]**. Το πρώτο θα το χρησιμοποιήσουμε για να εκπαιδεύσουμε το μοντέλο και γενικά αποτελεί το μεγαλύτερο μέρος των δεδομένων και ονομάζεται training set. Το δεύτερο μέρος του dataset το χρησιμοποιούμε όταν κάνουμε ρύθμιση υπερ παραμέτρων(hyperparameter tuning) και αξιολογούμε την επίδοση του σε δεδομένα που δεν έχει ξαναδεί. Αυτή η διαδικασία αφορά την δοκιμή υπερ παραμέτρων του μοντέλου. Οι υπερ παράμετροι είναι παράμετροι του αλγορίθμου που εκπαιδεύουμε οι οποίες δεν έχουν σχέση με το dataset. Ο κάθε αλγόριθμος έχει διαφορετικές υπερ παραμέτρους όπως για παράδειγμα στον k-NN πρέπει να διαλέξουμε τον αριθμό των γειτόνων που θα λάβουμε υπόψιν. Γενικά αυτές οι παράμετροι επιλέγονται πειραματικά. Η διαδικασία γίνεται συνήθως χρησιμοποιώντας την μέθοδο του GridSearch**[54].** Η μέθοδος αυτή δοκιμάζει μεγάλο εύρος τιμών και κρατόντας την ιστορία επιλέγει εκείνες που έδωσαν τα καλύτερα αποτελέσματα. Για αυτή την διαδικασία χρησιμοποιούμε το δεύτερο set που ονομάζεται validation set. Ο διαχωρισμός μπορεί να γίνει με διάφορους τρόπους αλλά εδώ απλά περιγράψαμε τον πιο απλό. Τέλος το τρίτο μέρος από το διαχωρισμένο dataset αποτελεί το test set. Αυτό το μέρος των δεδομένων σκοπό έχει να κάνει μία

ανεξάρτητη αξιολόγηση του καλύτερου μοντέλου που προέκυψε από την παραπάνω διαδικασία. Είναι σημαντικό τα δεδομένα του test set να μην έχουν χρησιμοποιηθεί ούτε στο train ούτε στο validation.

2.2.7 Επιλογή αλγορίθμου

Σε αυτήν την ενότητα θα αναλύσουμε κριτήρια επιλογής**[55]** του καταλληλου αλγορίθμου. Να σημειωθεί ότι δεν υπάρχει καλύτερος ή χειρότερος αλγόριθμος και η επιλογή γενικά εξαρτάται από πολλούς παράγοντες. Πολλές φορές ακολουθείται η διαδικασία της δοκιμής στα πρότυπα του hyperparameter tuning και δοκιμάζεται ένα εύρος αλγορίθμων. Παρόλα αυτά ο κάθε αλγόριθμος έχει χαρακτηριστικά όπως η ταχύτητα εκπαίδευσης ή ο χρόνος πρόβλεψης μεταξύ άλλων. Αυτά τα χαρακτηριστικά όπως και το μέγεθος του dataset ή ο τύπος των δεδομένων εκπαίδευσης μπορούν να καθοδηγήσουν την επιλογή. Γενικά μπορούμε να λάβουμε υπόψη τα παρακάτω.

Είδος χαρακτηριστικών

Κάποιοι αλγόριθμοι είναι πιο κατάλληλοι για categorical features ενώ άλλοι οχι.

Αριθμός χαρακτηριστικών

Κάποιοι αλγόριθμοι είναι πιο ανθεκτικοί στο curse of dimensionality σε σχέση με άλλους που δεν είναι τόσο.

Ταχύτητα εκπαίδευσης

Κάποιοι αλγόριθμοι είναι πιο γρήγοροι στην εκπαίδευση σε σχέση με άλλους ή ακόμα και δεν χρειάζονται καθόλου εκπαίδευση όπως για παράδειγμα ο k-NN

Ταχύτητα Πρόβλεψης

Αυτό το κριτήριο μπορεί να παίζει σημαντικό ρόλο στην επιλογή ειδικά για use cases όπως αυτό της παρούσας εργασίας όπου η πρόβλεψη πρέπει να γίνεται σχεδόν σε πραγματικό χρόνο.

Μέγεθος Dataset

Υπάρχουν αλγόριθμοι που δεν μπορούν να λειτουργήσουν αποδοτικά όταν το dataset είναι πολύ μικρό ή πολύ μεγάλο. Ένα παράδειγμα αλγορίθμου που δεν μπορεί να δουλέψει με μεγάλο dataset είναι ο k-NN και ένα παράδειγμα αλγορίθμου που δουλεύει καλά με μικρό dataset είναι ο Naive Bayes.

Incremental Training

Σε κάποια use cases είναι βασικό requirement το dataset να ανανεώνεται με νέα δεδομένα**[32]**. Αλγόριθμοι σαν τον SVM δεν είναι κατάλληλοι για τέτοια σενάρια ενώ ο k-NN είναι.

Επεξηγησιμότητα(Explainability)[56]

Σε πολλές περιπτώσεις βασική απαίτηση είναι να μπορούμε να ερμηνεύσουμε και να δικαιολογήσουμε την πρόβλεψη του αλγορίθμου. Αλγόριθμοι όπως ο Decision Tree προσφέρουν

αυτήν την δυνατότητα σε κάποιο βαθμό ενώ τα Νευρωνικά Δίκτυα όπως θα δούμε στην συνέχεια δεν την προσφέρουν καθόλου.

Μέγεθος Βεβαιότητας(Probabilistic output)

Σε πολλές περιπτώσεις μας ενδιαφέρει εκτός από την πρόβλεψη της κλάσης ενός δείγματος μας ενδιαφέρει και το μέτρο που δείχνει την βεβαιότητα αυτής της απόφασης. Τα νευρωνικά δίκτυα μπορούν να το κάνουν αυτό ενώ ο Decision Tree οχι.

Φυσικά ότι αναφέραμε εδώ δεν εξαντλεί όλα τα κριτήρια απλά παρουσιάζει τα βασικα κριτήρια που πρέπει να λάβουμε υπόψη για την διαδικασία της επιλογής.

2.2.8 Μετρικές Αξιολόγησης Επίδοσης

Σε αυτήν την ενότητα θα παρουσιάσουμε τα βασικά μετρικά**[57]** που συνήθως χρησιμοποιούνται σε προβλήματα επιβλεπόμενης Μηχανικής Μάθησης που αφορούν περιπτώσεις προβλημάτων classification. Το πιο από αυτά χρησιμοποιείται για την αξιολόγηση της επίδοσης του μοντέλου σχετίζεται με το είδος του προβλήματος και τις απαιτήσεις που έχουν οριστεί. Για παράδειγμα το μετρικό του accuracy δεν είναι ιδανικό και αντιπροσωπευτικό της επίδοσης όταν το classification πρόβλημα είναι imbalanced. Επίσης σε ένα πρόβλημα όπως η διάγνωση μιας ασθένειας μας ενδιαφέρει να έχουμε μεγάλο accuracy για τα δείγματα που βγαίνουν θετικά αλλά δεν μας ενδιαφέρει το ίδιο η πιθανότητα να έχουμε λάθος θετική διάγνωση δηλαδή False Positives. Πριν δώσουμε τους ορισμούς των βασικών μετρικών που χρησιμοποιούνται πρέπει να δώσουμε τους ορισμούς τεσσάρων βασικών εννοιών. Αυτές είναι false positive,false negative,true positive,true negative. Αυτές οι έννοιες ορίζονται ως εξής.

False Positive

Αυτό σημαίνει λανθασμένη εκτίμηση ότι ένα γεγονός η μία κατάσταση πραγματοποιήθηκε.

False Negative

Αυτό σημαίνει λανθασμένη εκτίμηση ότι ένα γεγονός ή μια κατάσταση δεν πραγματοποιήθηκε.

True Positive

Αυτό σημαίνει ότι ορθώς αναγνωρίστηκε ότι ένα γεγονός ή μία κατάσταση πραγματοποιήθηκε.

True Negative

Αυτό σημαίνει ότι ορθώς αναγνωρίστηκε ότι ένα γεγονός ή μία κατάσταση δεν πραγματοποιήθηκε.

Ένα σημαντικό εργαλείο που χρησιμοποιεί αυτές τις έννοιες είναι το Confusion Matrix**[58]** γνωστό και ως πίνακας λάθους(error matrix) και χρησιμεύει στην οπτικοποίηση της επίδοσης του

αλγορίθμου κατηγοριοποίησης. Σε αυτόν τον πίνακα και για προβλημα classification με πολλές κλάσεις έχουμε γραμμές και στήλες ίσες με τον αριθμό των κλάσεων. Οι στήλες δείχνουν τα datapoints που προέβλεψε το μοντέλο και οι γραμμές δείχνουν την πραγματική τιμή(ground truth) δηλαδή τις σωστές εκτιμήσεις με βάση το train/validation dataset. Τα στοιχεία της διαγωνίου δείχνουν τις σωστές εκτιμήσεις για κάθε κλάση. Τα στοιχεία μιας γραμμής δείχνουν το σύνολο των πραγματικών δειγμάτων για κάθε κλάση. Τα στοιχεία μιας στήλης δείχνουν πόσα δείγματα προβλέφθηκαν για την συγκεκριμένη κλάση. Με βάση αυτά μπορούμε τώρα να ορίσουμε τα βασικά metrics που χρησιμοποιούνται συνήθως. Αυτά είναι το accuracy το recall και το precision και Fscore.

- Accuracy

Ορίζεται ως το άθροισμα της διαγωνίου του Confusion Matrix διά το συνολικό άθροισμα των δειγμάτων στο validation ή στο test set. Αποτελεί αξιόπιστο metric μόνο στην περίπτωση του balanced classification.

- Recall

Πρόκειται για μετρικό ανά κλάση και ορίζεται ως το κλάσμα των σωστών προβλέψεων ανα κλάση προς το άθροισμα της γραμμής που σχετίζεται με την συγκεκριμένη κλάση. Δηλαδή οι σωστές προβλέψεις δια τον πραγματικό αριθμό των δειγμάτων για την συγκεκριμένη κλάση.

- Precision

Πρόκειται για μετρικό ανα κλάση και ορίζεται ως το κλάσμα των σωστών προβλέψεων για την συγκεκριμένη κλάση προς το άθροισμα της στήλης που σχετίζεται με την συγκεκριμένη κλάση. Δηλαδή οι σωστές προβλέψεις δια τον αριθμο που ο αλγόριθμος προέβλεψε για την συγκεκριμένη κλάση.

Αυτά τα τρία metrics είναι τα βασικά. Με βάση αυτά ορίζονται μερικά ακόμα που συνδυάζουν τα παραπάνω είτε για κάθε κλάση ξεχωριστά είτε για όλες τις κλάσεις ωστε να δώσουν μία πιο ακριβή εικόνα σε σχέση με το απλό accuracy. Αυτά είναι το F1 macro-F1 και weighted-F1 και ορίζονται ως εξής.

- F1

Προκειται για μετρικό ανα κλάση που συνδυάζει το recall και το precision και ορίζεται το κλάσμα του recal επί precision επι δύο διά το άθροισμα recall και precision.

- macro-F1

Ορίζεται ως το μέσο όρο των F1 για κάθε κλάση. Υπολογίζεται ως εξής. Πρώτα υπολογίζουμε το μέσο recall από όλες τις κλάσεις, στην συνέχεια το μέσο precision και με βάση αυτά υπολογίζεται το macro-F1.

- weighted F1

Υπολογίζεται με βάση το macro-F1 αλλά το F1 ανά κλάση έχει βάρος αντίστοιχο με τον αριθμό δειγμάτων ανα κλάση.

Άλλα δύο εργαλεία που μπορούν να φανούν χρήσιμα είναι οι καμπύλες ROC και PR και τις περιγράφουμε στην συνέχεια.

- ROC

Ονομάζεται Receiver Operating Characteristic και μετράει το recall διά το false positive rate. Πρόκειται για metric ανά κλάση και η καμπύλη δημιουργείται από τους classifiers που δοκιμάζουμε. Υπολογίζουμε μία καμπύλη ανά κλάση.

- PR

Ονομάζεται Precision-Recall. Πρόκειται για καμπύλη που δείχνει την σχέση του precision προς το recall. Πάλι μία καμπύλη ανα κλάση η οποία προκύπτει από τους διαφορετικούς classifiers που δοκιμάζουμε.

Με βάση τα μετρικά που περιγράφηκαν προκύπτει το ερώτημα πως επιλέγουμε την αξιολόγηση που θα κάνουμε στον ή στους classifiers που εκπαιδεύτηκαν. Γενικά εξαρτάται από το πρόβλημα αλλά υπάρχουν κάποιες κατευθύνσεις(guidelines). Αν μας ενδιαφέρει ποιός είναι συνολικά ο καλύτερος classifier είναι καλό να λαμβάνουμε υπόψη τις καμπύλες ROC. Αν μας ενδιαφέρει μία ή περισσότερες κλάσεις τότε μπορούμε να δούμε το recall ή το precision για αυτήν/αυτές τις κλάσεις μόνο όπως το αναφέραμε στο παράδειγμα της διάγνωσης ή μη μίας ασθένειας.

2.2.9 Συνηθισμένα Προβλήματα

Σε αυτήν την ενότητα θα περιγράψουμε μερικά από τα πιο συνηθισμένα προβλήματα που παρουσιάζονται στην διαδικασία της εκπαίδευσης και τρόπους που αντιμετωπίζονται καθώς αφορούν γενικά την διαδικασία αλλά και ειδικά την περίπτωση αυτής της εργασίας.

Αυτά είναι τα εξής Imbalanced Classes, overfitting, underfitting. Στην συνέχεια θα τα περιγράψουμε.

- Under Fitting

Ονομάζεται το σενάριο**[59]** όπου το μοντέλο που εκπαιδεύεται με επιβλεπόμενη μάθηση δεν καταφέρνει να βρει τις απαραίτητες συσχετίσεις ανάμεσα στις ανεξάρτητες μεταβλητές και την εξαρτημένη μεταβλητή. Αποτέλεσμα είναι το μοντέλο να έχει υψηλό σφάλμα(error rate) και στα δεδομένα εκπαίδευσης αλλά και στα validation και test set. Το πρόβλημα αυτό μπορεί να προκύψει αν ένα μοντέλο είναι πολύ απλό, αν χρειάζεται περισσότερο χρόνο εκπαίδευσης, αν τα χαρακτηριστικά που χρησιμοποιούνται δεν περιέχουν αρκετή πληροφορία, αν χρειάζονται περισσότερα χαρακτηριστικά ή μπορεί να είναι αποτέλεσμα πολύ αυστηρού regularization. Για να βρούμε την λύση σε αυτό το πρόβλημα η πρώτη επιλογή είναι να κάνουμε το μοντέλο πιο πολύπλοκο και να μειώσουμε το regularization**[60]**. Στην συνέχεια αν δεν έχουμε αποτέλεσμα θα πρέπει να

βρούμε περισσότερα δεδομένα εκπαίδευσης είτε κανοντας annotate περισσότερα δεδομένα είτε χρησιμοποιώντας τεχνικές όπως το SMOTE**[61,65]** τεχνική που παράγει τεχνητά δεδομένα εκπαίδευσης με βάση τα υπάρχοντα μέθοδος που θα αναφέρουμε αναλυτικά στην συνέχεια. Το υψηλό bias και το χαμηλό variance είναι χαρακτηριστικές ενδείξεις underfitting. Το bias ορίζεται ως η μέση διαφορά ανάμεσα σε πρόβλεψη και πραγματική τιμή. Το variance ορίζεται ως η ευαισθησια του μοντέλου σε παρόμοιες εισόδους. Υψηλό bias σημαίνει ότι το μοντέλο κάνει πολλά λάθη στα δεδομένα εκπαίδευσης. Χαμηλό bias σημαίνει ότι το μοντέλο κάνει σωστές προβλέψεις στα δεδομένα εκπαίδευσης. Υψηλό variance σημαίνει ότι το μοντέλο δίνει μεγάλη σημασία στα δεδομένα εκπαίδευσης με αποτέλεσμα να μην γενικεύει σωστά. Χαμηλό variance σημαίνει ότι το μοντέλο γενικεύει καλά.

- Over fitting

Το over fitting**[59]** συμβαίνει όταν το μοντέλο έχει μάθει πολύ καλά τα δεδομένα εκπαίδευσης με αποτέλεσμα να μην γενικευει σωστά σε δεδομένα που δεν έχει δει δηλαδή στο validation και train set. Αυτή η περίπτωση είναι πιο συνηθισμένη από το under fitting. Συμβαίνει όταν το μοντέλο είναι περισσότερο πολύπλοκο από όσο θα έπρεπε. Επίσης μπορεί να συμβεί όταν η εκπαίδευση του μοντέλου διαρκέσει πολύ με αποτέλεσμα να αρχίσει να μαθαίνει και τον θόρυβο πέρα από την πληροφορία στα δεδομένα εκπαίδευσης. Μπορεί να συμβεί και στην περίπτωση που έχουμε πολλά χαρακτηριστικά ή όταν έχουμε πολύ λίγα δεδομένα εκπαίδευσης. Ενδείξεις της συμπεριφοράς αυτής είναι το υψηλό variance και το πολύ μικρό error rate στο dataset εκπαίδευσης. Αυτός είναι ο κύριος λόγος που χρησιμοποιούμε το validation dataset. Οι τρόποι που μπορούν να δοκιμαστούν για να ξεπεραστεί το πρόβλημα είναι ή να κάνουμε το μοντέλο λιγότερο πολύπλοκο ή να μειώσουμε τα χαρακτηριστικά ή να βρούμε περισσότερα δεδομένα εκπαίδευσης ή τέλος να χρησιμοποιήσουμε τεχνικές regularization. Οι τεχνικές regularization**[60]** στοχεύουν στο να κάνουν το μοντέλο πιο απλό προσθέτοντας σε πολλές περιπτώσεις έναν όρο ποινής(penalty) στο loss function για πολύπλοκα μοντέλα με χαρακτηριστικό παράδειγμα το L2 Regularization. Φυσικά υπάρχουν και άλλες τεχνικές ανάλογα αν μιλάμε για shallow ή Deep μοντέλα. Για deep μοντέλα μπορούμε να χρησιμοποιήσουμε τεχνικές όπως το data augmentation**[63]** που χρησιμοποιείται πολύ συχνά σε CNNs νευρωνικά δίκτυα που επεξεργάζονται εικόνες. Αυτή η μέθοδος πραγματοποιεί μετασχηματισμούς στις εικόνες όπως random rotation random scaling και άλλα. Άλλη μία τεχνική που χρησιμοποιείται συχνά σε νευρωνικά χωρίς να περιορίζεται στα CNNs όπως η προηγούμενη είναι η τεχνική του dropout**[62]**. Εδώ και στην διάρκεια της εκπαίδευσης μόνο αγνοούμε τα αποτελέσματα μεγάλου μέρους νευρώνων σε κάθε κρυμμένο επίπεδο ώστε το μοντέλο να μήν βασίζεται σε συγκεκριμένους νευρώνες για την πρόβλεψη. Αυτή η τεχνική χρησιμοποιείται σχεδόν πάντα σε όλους τους τύπους νευρωνικών δικτύων.

- Class Imbalance

Πρόκειται για πολύ συνηθισμένο πρόβλημα**[64]** σε datasets που προκύπτουν σε σενάρια του πραγματικού κόσμου. Σε πολλές περιπτώσεις υπάρχει μία κυρίαρχη κλάση στην οποία ανήκουν τα περισσότερα δείγματα και οι υπόλοιπες κλάσεις έχουν πολύ λίγα δείγματα. Αυτό συμβαίνει πολύ συχνά σε datasets που αφορούν δεδομένα υγείας όπου το μεγαλύτερο μέρος είναι αρνητικά και ένα μόνο μικρό μέρος είναι θετικά. Επίσης συμβαίνει πολύ συχνά σε δεδομένα συναλλαγών όπου σκοπός είναι η ανίχνευση απάτης και ένα πολύ μικρό ποσοστό είναι πραγματική απάτη. Όπως αναφέραμε και προηγουμένως στην ενότητα με τα μετρικά το accuracy σε τέτοιες περιπτώσεις δεν είναι αντιπροσωπευτικό της επίδοσης του αλγορίθμου. Αν δεν μπορέσουμε να αντιμετωπίσουμε αυτό το πρόβλημα συνήθως προκύπτουν δύο ενδεχόμενα. Το πρώτο είναι ότι το μοντέλο που θα εκπαιδευτεί θα δίνει σημασία μόνο στην κλάση με τα πολλά δείγματα και δεν θα μάθει τις κλάσεις με τα λίγα δείγματα(minority classes). Η δεύτερη περίπτωση είναι να αποτύχει η εκπαίδευση εξ ολοκλήρου λόγω λίγων δειγμάτων. Στην συνέχεια θα παρουσιάσουμε βασικές τεχνικές και μεθόδους που επιχειρούν να αντιμετωπίσουν το πρόβλημα. Αυτές οι τεχνικές φυσικά μπορούν να συνδυαστούν ανάλογα την περίπτωση.

Μέτρικό Αξιολόγησης

Η πρώτη επιλογή είναι να αλλάξουμε τον τρόπο που αξιολογούμε το μοντέλο από τον συνηθισμένο τρόπο που είναι η χρήση του συνολικού accuracy σε άλλα μετρικά τα οποία σχετίζονται με κάποια συγκεκριμένη κλάση όπως το recall για τις κλάσεις μειοψηφίας. Έτσι το μοντέλο μπορεί να εστιάσει σε αυτές.

Δοκιμή άλλων Αλγορίθμων

Πολλοί αλγόριθμοι δεν είναι ιδανικοί για imbalanced datasets ενώ άλλοι όπως ο Decision Tree αποδίδουν καλύτερα σε κάποιο βαθμό όταν υπάρχει αυτό το πρόβλημα. Έτσι αποτελεί καλή επιλογή να εξετάσουμε πολλές περιπτώσεις.

Υπερ δειγματοληψία(Over Sampling)

Αυτή η μέθοδος**[64]** μπορεί να εφαρμοστεί με δύο τρόπους. Ο πρώτος είναι να βρούμε περισσότερα δεδομένα για τις κλάσεις με τα λίγα δείγματα. Αυτό απαιτεί έναν νέο κύκλο απόκτησης δεδομένων και annotation. Αυτή η διαδικασία όπως αναφέραμε και προηγουμένως μπορεί να είναι από χρονοβόρα και ακριβή έως αδύνατη πράγμα που μας φέρνει στην δεύτερη επιλογή. Η δεύτερη επιλογή είναι η δημιουργία τεχνιτών δεδομένων**[61]** για τις κλάσεις με τα λίγα δείγματα, με τεχνικές όπως αυτή του SMOTE. Η πιο απλή περίπτωση αυτής της μεθόδου είναι απλά να δημιουργήσουμε αντίγραφα των δειγμάτων από τις κλάσεις με τα λίγα δείγματα με αποτέλεσμα η διαδικασία της εκπαίδευσης να μην είναι biased προς την κλάση με τα πολλά δείγματα. Το πρόβλημα με αυτό είναι ότι δεν δημιουργούμε παραπάνω πληροφορία. Αυτο μας οδηγεί στην επιλογή τεχνικών όπως αυτή του SMOTE**[61,65]**. Εδώ δημιουργούνται συνθετικά δεδομένα και όχι απλά αντίγραφα. Ο αλγόριθμος δουλευει ως εξής. Πρώτα βρίσκει τους k κοντινοτερους γειτονες ενός δείγματος που

ανήκουν στην ίδια κλάση. Στην συνέχεια διαλέγει έναν από τους γείτονες με τυχαίο τρόπο και δημιουργεί ένα καινούριο δείγμα με βάση το πρώτο και το δεύτερο. Αυτή η τεχνική σε πολλές περιπτώσεις δίνει πολύ καλά αποτελέσματα. Τέλος μία πολύ σημαντική παρατήρηση είναι ότι τέτοιου είδους μέθοδοι εφαρμόζονται μόνο στο train dataset και πάντα μετά το διαχωρισμό σε train/validation/test.

Υποδειγματοληψία(Under Sampling)

Αυτή η μέθοδος**[64]** είναι η αντίθετη με την προηγούμενη. Εδώ θεωρώντας ότι έχουμε αρκετά samples για train/validation/test για τις minority classes και για να λύσουμε το πρόβλημα του imbalance μπορούμε να κρατήσουμε μονο όσα samples χρειάζονται και να προχωρήσουμε στην εκπαίδευση. Βασικό σημείο εδώ είναι ότι ο τρόπος που θα γίνει η επιλογή των samples πρέπει να είναι τυχαίος διότι πολλές φορές υπάρχουν συσχετίσεις μεταξύ των samples. Ακόμα και έτσι όμως υπάρχει περίπτωση να χαθεί πληροφορία. Αν δεν έχουμε αρκετά samples για να γίνει αυτό κινδυνεύουμε από underfitting.

Επαύξηση Δεδομένων(Data Augmentation)

Η τελευταία αυτή τεχνική**[63]** χρησιμοποιείται κυρίως σε δεδομένα εικόνας και μοιάζει με την τεχνική που περιγράψαμε προηγουμένως για τα CNNs. Εδώ κάνουμε αλλαγές στις εικόνες με στόχο να υπάρχει diversity στο dataset. Γενικά μπορεί να επεκταθεί σαν μέθοδος σε άλλα domains όπως ο ήχος και γενικά μπορεί να χαρακτηριστεί σαν εισαγωγή θορύβου στο dataset.

2.3 Βαθιά Μηχανική Μάθηση

Η βαθιά Μηχανική Μάθηση**[66]** είναι υποσύνολο της Μηχανικής Μάθησης και για αυτό πολλά από όσα αναφέραμε στην προηγούμενη ενότητα ισχύουν και εδώ αλλά υπάρχουν διαφορές. Για παράδειγμα η διαδικασία της εκπαίδευσης, όπου χωρίζουμε το dataset σε train/validation/test είναι πανομοιότυπο και στο Deep Learning. Τα προβλήματα που αναφέραμε επίσης δηλαδή underfitting overfitting class imbalance ισχύουν και εδώ. Επίσης υπάρχουν και κοινές τακτικές για regularization όπως το L2. Σε αυτήν την ενότητα δεν θα επαναλάβουμε όσα είπαμε στην προηγούμενη αλλά θα τονίζουμε όπου χρειάζεται ότι είναι κοινό. Θα αναφέρουμε λίγα πράγματα για το τί είναι τα Νευρωνικά δίκτυα και πως προέκυψαν. Στην συνέχεια θα αναφέρουμε τεχνικές για reguralization**[60]** που εφαρμόζονται αποκλειστικά για νευρωνικά δίκτυα. Τέλος όπως κάναμε και στην προηγούμενη ενότητα θα παρουσιάσουμε βασικούς αλγορίθμους Βαθιάς Μηχανικής Μάθησης εστιάζοντας σε προβλήματα classification.

2.3.1 Νευρωνικά Δίκτυα

Ο τομέας της Βαθιάς Μηχανικής Μάθησης απέκτησε μεγάλη σημασία λόγω προόδου κυρίως στο Automatic Speech Recognition και το computer vision όταν άρχισαν να γίνονται διαθέσιμα δεδομένα ώστε να είναι εφικτή η εκπαίδευση. Οι shallow αλγόριθμοι ML όμως δεν λειτουργούν καλά με μη προ επεξεργασμένα δεδομένα και τις περισσότερες φορές χρειάζονται προ επεξεργασία. Επίσης η απόδοση των shallow αλγορίθμων φτάνει σε τέλμα σε μεγάλα datasets πράγμα που σημαίνει ότι δεν εκμεταλλευόμαστε τα διαθέσιμα δεδομένα. Τέλος αρχιτεκτονικές νευρωνικών δικτύων όπως τα Convolutional Neural Networks**[67]** στον τομέα του Computer Vision και οι Transformers**[22]** στον τομέα της Επεξεργασίας Φυσικής Γλώσσας έδωσαν εντυπωσιακά αποτελέσματα με σχεδόν μηδενική προ επεξεργασία συγκρίνοντας με τις προηγούμενες λύσεις Μηχανικής Μάθησης. Τα νευρωνικά δίκτυα εκτός από αυτούς τους τομείς βρήκαν εφαρμογή και σε άλλους όπως η Ενισχυτική μάθηση**[31]** η οποία γενικά περιοριζόταν σε χαμηλής διάστασης χώρους με χειροποίητα(handcrafted) χαρακτηριστικά. Τα νευρωνικά Δίκτυα Deep Q Networks**[68]** κατάφεραν να δώσουν πολύ καλά αποτελέσματα σε σενάρια χρήσης όπως atari games όπου το input είναι συνεχές και υψηλής διάστασης εικόνα. Άλλος ένας τομέας εξαιρετικής απόδοσης είναι η Μηχανική Μάθηση σε γράφους με νευρωνικά δίκτυα που ονομάζονται Graph Neural Networks. Σε αυτόν τον τομέα οι περισσότερες λύσεις βασίζονταν σε χειροποίητα(handcrafted) χαρακτηριστικά και δεν μπορούσαν να αξιολογήσουν συνδυαστικά και πληροφορία από την θέση ενός κόμβου σε έναν γράφο, τους γείτονες του σε μεγαλύτερη απόσταση από ένα καθώς τα χαρακτηριστικά ενός κόμβου ή μίας ακμής. Με την έλευση των GNNs**[14]** η απόδοση των προβλημάτων όπως κατηγοριοποίηση κόμβου(node classification) κατηγοριοποίηση γράφου και πρόβλεψη σχέσης και άλλα. Τέλος τα GNNs άνοιξαν τον δρόμο για Inductive Learning**[69]** σε γράφους. Η διαφορά με το Transductive είναι ότι υπάρχει η δυνατότητα το μοντέλο να παράγει αναπαραστάσεις(embendings) για γράφους που δεν έχει ξαναδεί όπως ο αλγόριθμος GraphSAGE**[69]**. Η λύσεις μέχρι τότε ήταν περιορισμένες σε στατικούς γράφους. Γενικά τα μεγάλα πλεονεκτήματα των νευρωνικών δικτύων είναι ότι μπορούν να αξιοποιήσουν μεγάλο όγκο δεδομένων και μπορούν να μάθουν μόνα τους τα απαραίτητα χαρακτηριστικά μέθοδος που αποδίδει πολύ καλύτερα σε σχέση με χειροποίητα(handcrafted) αν και αυτό μπορεί να εξαρτάται από τα διαθέσιμα δεδομένα.

2.3.2 Δομικά στοιχεία Νευρωνικών Δικτύων

Σε αυτήν την ενότητα θα περιγράψουμε τα βασικά στοιχεία**[70]** ενός νευρωνικού δικτύου καθώς και βασικές έννοιες.

- Νευρώνας

Πρόκειται για το βασικό στοιχείο ενός νευρωνικού. Αυτό που κάνει είναι να αθροίζει εισόδους αφού πρώτα έχει ζυγίσει την κάθε μία με ένα βάρος. Στην συνεχεια δίνει σαν έξοδο το αποτέλεσμα της προηγούμενης εισόδου από μία συνάρτηση ενεργοποίησης(activation function).

- Συνάρτηση Ενεργοποίησης(Activation Function)

Αυτό είναι το συστατικό που διαχωρίζει τα Deep μοντέλα από τα απλά γραμμικά μοντέλα. Μέσω της συνάρτησης αυτής εισάγεται μη γραμμικότητα στο μοντέλο. Στην βιβλιογραφία αποδεικνύεται ότι χωρίς activation function ένα νευρωνικό δίκτυο είναι απλά ένα γραμμικό μοντέλο. Επίσης αποδεικνύεται ότι ένα νευρωνικό δίκτυο με μη γραμμικότητα μπορεί να προσεγγίσει οποιαδήποτε συνάρτηση. Παραδείγματα συναρτήσεων ενεργοποίησης είναι οι RELU Leaky RELU tanh. Επίσης εκτός από την μη γραμμικότητα είναι επίσης χρήσιμες στο να περιορίζουν την έξοδο σε μικρές τιμές κάτι που χρειάζεται διότι σε διαφορετική περίπτωση προκύπτουν υπολογιστικές δυσκολίες λόγω πολύ μεγάλων ή μικρών αριθμών. Τα βασικά προαπαιτούμενα για μία συνάρτηση ενεργοποίησης είναι τα ακόλουθα. Πρώτον πρέπει να είναι zero centered, δεύτερον πρέπει να είναι εύκολη στον υπολογισμό και τρίτον πρέπει να είναι διαφοροποιήσιμη ή μερικώς διαφοροποιήσιμη .

- Επίπεδο Εισόδου(Input Layer)

Αυτό είναι το πρώτο επίπεδο του δικτύου. Εδώ εισάγουμε δεδομένα ή χαρακτηριστικά στο δίκτυο. Η είσοδος μπορεί να είναι είτε scalar είτε vector matrix ή tensor. Το επίπεδο αυτό δεν το μετράμε συνήθως.

- Επίπεδο Εξόδου(Output Layer)

Το επίπεδο αυτό είναι το τελευταίο. Εδώ παίρνουμε το αποτέλεσμα του υπολογισμού. Για προβλήματα classification έχουμε τόσες εξόδους όσες και πιθανές κλάσεις και χρησιμοποιείται η softmax για να μετατρέψουμε την έξοδο σε πιθανότητα. Αυτό επιτυγχάνεται διαιρώντας την κάθε έξοδο με το άθροισμα όλων των εξόδων.

- Κρυμμένα Επίπεδα(Hidden Layer)

Αυτά τα επίπεδα χρησιμεύουν γενικά για να μάθουν features. Τα features αυτά δεν έχουν κάποιο νόημα ή κάποια εξήγηση στα μάτια ενός ανθρώπου αλλά το δίκτυο καταφέρνει να βρίσκει πολύπλοκα μοτίβα που κρύβονται στα δεδομένα. Για το παράδειγμα των συνελικτικών(convolutional) το δίκτυο αντί να χρησιμοποιεί έτοιμα φίλτρα που ανιχνεύουν για παράδειγμα ακμές μαθαίνει φίλτρα που ταιριάζουν στο συγκεκριμένο πρόβλημα. Προσθέτοντας και άλλα κρυμμένα επίπεδα καταφέρνουμε να βρίσκουμε πιο πολύπλοκα features.

- Back Propagation**[71]**

Η εκπαίδευση γενικά και στα shallow και στα deep μοντέλα είναι ίδια. Το πρώτο βήμα είναι το forward propagation όπου γίνεται η πρόβλεψη. Στην συνέχεια συγκρίνουμε την πρόβλεψη με την πραγματική τιμή και υπολογίζεται το σφάλμα(loss). Αφού υπολογιστεί το σφάλμα πραγματοποιείται μία μικρή αλλαγή στα βάρη προς την κατεύθυνση διόρθωσης του λάθους. Η διαδικασία είναι απλή

για τα shallow μοντέλα αλλά τι συμβαίνει με τα νευρωνικά δίκτυα με πολλά κρυμμένα επίπεδα. Η διαδικασία ονομάζεται backpropagation και χρησιμοποιώντας το chain rule υπολογίζει το λάθος ξεκινώντας από το τελευταίο κρυμμένο επίπεδο αλλάζοντας τις παραμέτρους με μικρά βήματα προς τα επίπεδα πιο κοντά στο input layer.

- Ρυθμός Ανανέωσης Παραμέτρων(Learning rate)

Πρόκειται για τον ρυθμό με τον οποίο αλλάζουν οι παράμετροι του μοντέλου. Πολύ μικρή τιμή αυτής της παραμέτρου σημαίνει περισσότερος χρόνος εκπαίδευσης αλλα ταυτόχρονα πιο ομαλή αλλαγή των βαρών. Μεγάλη τιμή της παραμέτρου σημαίνει πιο γρήγορη εκπαίδευση αλλα και πιο θορυβώδης. Αυτή η παράμετρος αποτελεί υπερ παράμετρο του μοντέλου.

- Αλγόριθμος Βελτιστοποίησης(Optimizer) **[72]**

Ο πιο γνωστός τέτοιος αλγόριθμος είναι ο Stochastic Gradient Descent. Αυτός έχει ως στόχο με χρήση του learning rate και των gradients να ανανεώνει τις παραμέτρους του μοντέλου. Επειδή πολλές φορές το μέγεθος του dataset είναι μεγάλο και δεν χωράει στην μνήμη υπάρχει ο stochastic gradient descent και ο mini batch stochastic gradient descent. Ο πρώτος χρησιμοποιεί ένα μόνο δείγμα για την ανανέωση των βαρών ενώ ο δεύτερος το λεγόμενο mini batch δηλαδή ένα μικρό μέρος των διαθέσιμων δειγμάτων. Εκτός από τον SGD υπάρχουν και άλλοι αλγόριθμοι που μοιάζουν αλλα έχουν ώς κύρια διαφορά ότι αλλάζουν το learning rate κατα την διάρκεια της εκπαίδευσης

- Εποχή(Epoch)

  Πρόκειται για μία πλήρη χρήση του dataset.

- Regularization**[60]**

Πρόκειται για τεχνικές αντιμετώπισης του overfitting όπως αναφέραμε στην αντίστοιχη ενότητα. Στην περίπτωση του Deep Learning και των νευρωνικών υπάρχουν και άλλες τεχνικές από αυτές που αναφέραμε. Η μέθοδος L2 υπάρχει και εδώ. Αυτή η μέθοδος δίνει μία ποινή στην συνάρτηση κόστους για πολύπλοκα μοντέλα. Άλλη μέθοδος που υπάρχει για τα νευρωνικά δίκτυα είναι το early stopping. Σε αυτήν την μέθοδο παρακολουθούμε συνήθως το validation σφάλμα ή το validation loss και αν δεν βελτιωθεί για ένα συγκεκριμένο πλήθος εποχών τερματίζεται η εκπαίδευση και επαναφέρονται οι παράμετροι του καλύτερου μοντέλου που είχε καταγραφεί. Άλλη μέθοδος είναι αυτή του weight decay η οποία μοιάζει με την L2. Τέλος βασική μέθοδος είναι αυτή του Dropout για την οποία έγινε ήδη μια σύντομη αναφορά. Εδώ σε κάθε κρυμμένο επίπεδο συνήθως και με τυχαίο τρόπο αγνοούμε την ύπαρξη νεωρώνων στο forward propagation ώστε το αποτέλεσμα της πρόβλεψης να μην βασίζεται μόνο σε μικρό μέρος του νευρωνικού δικτύου. Αυτό προφανώς συμβαίνει μόνο στην διάρκεια της εκπαίδευσης.

Όπως αναφέραμε προηγουμένως πολλά από αυτά που ισχύουν για τα shallow μοντέλα της Μηχανικής Μάθησης ισχύουν και για την περίπτωση των deep Neural Networks όπως η διαδικασία της εκπαίδευσης και αξιολόγησης του μοντέλου και τα είδη μάθησης. Δέν θα αναφέρουμε πολλά για

την περίπτωση της Βαθιάς Μηχανικής Μάθησης διότι στην λύση που θα υλοποιήσουμε δεν θα εκπαιδεύσουμε τέτοια μοντέλα διότι θα δείξουμε ότι τα shallow μοντέλα είναι αρκετά για αυτόν τον σκοπό. Παρόλα αυτά δώσαμε κάποιες βασικές έννοιες ώστε να μπορούν να γίνουν αντιληπτές οι μέθοδοι που χρησιμοποιούν άλλοι για να δώσουν λύσεις στο συγκεκριμένο πρόβλημα λύσεις οι οποίες βασίζονται είτε εν μέρη είτε εξ ολοκλήρου σε Βαθιά Μηχανική Μάθηση. Στην αντίστοιχη ενότητα θα παρουσιάσουμε την μεθοδολογία και τα αποτελέσματα που υπάρχουν από λύσεις που βασίζονται όχι μόνο σε Βαθιά Μηχανική Μάθηση. Γενικά τα μοντέλα νευρωνικών δικτύων είναι πολύ χρήσιμα στο να εξάγουν χαρακτηριστικά που υπάρχουν στα δεδομένα αλλά δεν είναι προφανή σε αντίθεση με τα shallow μοντέλα τα οποία χρειάζονται χειροποίητα(handcrafted) χαρακτηριστικά και για αυτόν τον λόγο υπάρχουν σχετικές εργασίες που χρησιμοποιούν νευρωνικά. Από την στιγμή που θα έχουν εξαχθεί τα χαρακτηριστικά είναι απλό με μερικά fully connected επίπεδα να δημιουργηθεί ένας classifier. Στην συνέχεια παρουσιάζουμε κάποιους βασικούς αλγορίθμους Βαθιάς Μηχανικής Μάθησης.

2.3.3 Βασικοί αλγόριθμοι Βαθιάς Μηχανικής Μάθησης

Σε αυτή την ενότητα θα παρουσιάσουμε συνοπτικά τους πιο γνωστούς αλγορίθμους Βαθιάς Μηχανικής Μάθησης παρόλο που δεν θα υλοποιήσουμε τέτοιο μοντέλο σε αυτήν την εργασία. Αυτό επειδή πλέον η τάση είναι να χρησιμοποιούνται τέτοια μοντέλα καθώς δίνουν πολύ καλά αποτελέσματα. Πολλά από τα μοντέλα που θα παρουσιάσουμε στην ενότητα Σχετικες εργασίες βασίζονται σε τεχνικές Βαθιάς Μηχανικής Μάθησης. Οι αλγόριθμοι που θα παρουσιάσουμε είναι οι CNN, GAN, AutoEncoders, RNNs.

- Συνελικτικά Νευρωνικά Δίκτυα(Convolutional Neural Networks)

Αυτός ο αλγόριθμος**[67]** βρίσκει εφαρμογή κυρίως σε εφαρμογές computer vision. Παράδειγμα εφαρμογών είναι image classification, object detection, image generation. Ο αλγόριθμος αυτός μιμείται την διαδικασία της ανθρώπινης όρασης και επιτυγχάνει τον σκοπό. Η είσοδος σε ένα CNN είναι μία δισδιάστατη δομή κατα τα πρότυπα που αποθηκεύονται οι εικόνες στους υπολογιστές. Δίνοντας σαν είσοδο δομές 2D αντί για την εναλλακτική του να αλλάξουμε την 2D δομή σε ένα διάνυσμα είναι το ότι με το διάνυσμα χάνουμε (contextual) πληροφορία και χρειαζόμαστε εντυπωσιακά μεγαλύτερο αριθμο παραμέτρων πράγμα που σημαίνει ανάγκη για περισσότερα δεδομένα, μεγαλύτερο χρόνο εκπαίδευσης και μεγαλύτερο χρόνο πρόβλεψης. Το CNN σε κάθε κρυμμένο επίπεδο περιλαμβάνει εκπαιδευόμενες παραμέτρους οι οποίες ονομάζονται kernels. Τα kernels δεν είναι τίποτα περισσότερο από φίλτρα που εξάγουν χαρακτηριστικά μέσω της πράξης της συνέλιξης(convolution) με την εικόνα. Στην συνέχεια ακολουθεί ένα επίπεδο ομαδοποίησης(pooling) που σκοπό έχει να μειώσει τις διαστάσεις των δομών που προκύπτουν από την συνέλιξη με κάθε

kernel. Βάζοντας πολλά τέτοια επίπεδα το ένα μετά το άλλο το δίκτυο μαθαίνει τα σωστά φίλτρα έτσι ώστε να μειώσει το σφάλμα εκπαίδευσης. Στο κάθε επίπεδο ουσιαστικά εξάγονται χαρακτηριστικά από την εικόνα. Στο πρώτο επίπεδο λέμε συνήθως ότι ανιχνεύονται ακμές γωνίες καμπύλες και σε κάθε επίπεδο εξάγονται όλο και πιο γενικά χαρακτηριστικά ανάλογα το σενάριο χρήσης. Στο τέλος για παράδειγμα στην περίπτωση του classification έχουμε κάποια fully connected επίπεδα που λειτουργούν ως classsifier. Αυτός ο αλγόριθμος είναι γνωστός για την χρήση σε computer vision προβλήματα αλλά όπως θα δούμε μπορεί να χρησιμοποιηθεί και αλλού. Ακόμα και αν τα δεδομένα που έχουμε είναι μονοδιάστατα δεν υπάρχει κάποιος λόγος να μην τα μετατρέψουμε σε δισδιάστατα και να χρησιμοποιήσουμε CNNs. Όπως θα δούμε στην ενότητα related work υπάρχουν τέτοιες προσεγγίσεις και θα τις παρουσιάσουμε.

- Generative Adversarial Networks

Πρόκειται για μια μέθοδο**[73]** η οποία στόχο έχει να εκπαιδεύσει δύο διαφορετικά δίκτυα με μία διαδικασία εκπαίδευσης. Το πρώτο μοντέλο έχει σαν σκοπό να μάθει να παράγει δεδομένα από την κατανομή των δεδομένων εκπαίδευσης όπως για παράδειγμα εικόνες από γάτες. Το δεύτερο μοντέλο σκοπό έχει να μάθει να ξεχωρίζει ψεύτικα δεδομένα από τα αληθινά. Με αυτό τον τρόπο τα δύο δίκτυα ανταγωνίζονται το ένα το άλλο κατα την διάρκεια της εκπαίδευσης και γίνονται καλύτερα για τον σκοπό του generation και discrimination αντίστοιχα. Υπάρχουν πολλές εφαρμογές για αυτό τον αλγόριθμο άλλες προφανείς όπως το generation περιεχομένου και η επαύξηση δεδομένων(data augmentation). Επίσης υπάρχουν και χρήσεις που δεν είναι προφανής όπως κάποιες λύσεις που θα δούμε στην ενότητα Σχετικες Εργασίες.

- AutoEncoders

Ένας πολύ γνωστός αλγόριθμος**[30]** Βαθιάς Μηχανικής Μάθησης για μάθηση αναπαραστάσεων σε χαμηλή διάσταση από υψηλής διάστασης δεδομένα είναι ο αλγόριθμος AutoEncoder. Πρόκειται για αλγόριθμο μη επιβλεπόμενης μάθησης που λειτουργεί ως εξής. Στο πρώτο επίπεδο εισάγονται τα μή προ επεξεργασμένα δεδομένα. Στην συνέχεια ακολουθούν κρυμμένα επίπεδα με διαστάσεις που μειώνονται σταδιακά μέχρι να φτάσουμε στο λεγόμενο bottleneck όπου αποτελεί και την χαμηλής διάστασης αναπαράσταση δηλαδή το χαρακτηριστικό. Στην συνέχεια ακολουθούν κρυμμένα επίπεδα όλο και μεγαλύτερων διαστάσεων με το τελικό να είναι ίδιας διάστασης με το επίπεδο εισόδου. Οι παράμετροι του μοντέλου ανανεώνονται με βάση το σφάλμα που προκύπτει από την ανακατασκευη των δεδομένων και όταν ολοκληρωθεί η εκπαίδευση το νευρωνικό έχει μάθει να παράγει σωστές αναπαραστάσεις των δεδομένων και μπορεί πλέον να χρησιμοποιηθεί σαν feature extractor.

- Επαναλαμβανόμενα Νευρωνικά Δίκτυα(Recurrent Neural Networks)

Πρόκειται για μία πολύ γνωστή κατηγορία**[74]** αλγορίθμων. Σκοπός αυτών των αλγορίθμων είναι να λαμβάνουν υπόψη όχι μόνο την τρέχουσα είσοδο αλλά και πληροφορία που είδαν στο

παρελθόν ή θα δουν στο μέλλον. Αυτό αναφέρεται ως contextual πληροφορία. Αυτοί οι αλγόριθμοι βρίσκουν εφαρμογή κυρίως σε προβλήματα από τον χώρο της Επεξεργασίας Φυσικής Γλώσσας. Στην πιο απλή περίπτωση ένα RNN σε κάθε βήμα παίρνει σαν είσοδο τα δεδομένα αλλά και πληροφορία από την προηγούμενη έξοδο. Με αυτόν τον τρόπο λαμβάνει υπόψη στην επεξεργασία τα "συμφραζόμενα"(context). Αυτή η απλοική μέθοδος έχει περιορισμούς όπως το ότι όσο ξεδιπλώνει το δίκτυο η πληροφορία από προηγούμενα βήματα χάνεται. Αυτό σημαίνει ότι η μνήμη που έχει είναι μικρή. Για αυτό το λόγο έχουν προταθεί πιο πολύπλοκες αρχιτεκτονικές που ονομάζονται Gated RNNs**[74]**. Μία από αυτές είναι τα LSTMs**[74]**. Εδώ εισάγεται ο όρος του κελιού(cell). Το κάθε κελί έχει forget, input και output gates και μαθαίνει ποια πληροφορία να ξεχνάει και ποια πληροφορία να κρατάει και να την προωθεί στο επόμενο βήμα. Αυτό βελτίωσε πολύ την απόδοση. Παρόλη την επιτυχία τους τα LSTMs έχουν και αυτά περιορισμούς. Η τελευταία μέθοδος σε αυτόν τον τομέα που έδωσε εντυπωσιακά αποτελέσματα είναι η μέθοδος sequence to sequence. Εδώ κεντρικό ρόλο παίζει ο encoder και ο decoder. Ο encoder επεξεργάζεται την ακολουθία εισόδου και παράγει ένα σύνολο από hidden states και ο decoder χρησιμοποιεί αυτή την πληροφορία για να επεξεργαστεί τα δεδομένα εισόδου. Μία άλλη περίπτωση αυτής της κατηγορίας είναι η περίπτωση των transformers**[22]** που λειτουργούν με το τρόπο που περιγράφηκε χρησιμοποιώντας έναν άλλο μηχανισμό που ονομάζεται μηχανισμός προσοχής(attention). Σε εφαρμογές Machine Translation έχουν δώσει εντυπωσιακά αποτελέσματα. Ο μηχανισμός Attention στη ουσία καθοδηγεί το μοντέλο στο να λάβει υπόψη συγκεκριμένη πληροφορία όταν επεξεργάζεται μία συγκεκριμένη είσοδο. Τα RNNs αποτελούν ένα μεγάλο πεδίο έρευνας και είναι πέρα από τους σκοπούς αυτής της εργασίας να τα παρουσιάσει με λεπτομέρεια. Όπως θα δούμε στην αντίστοιχη ενότητα υπάρχουν πολλές εργασίες που τα χρησιμοποιούν και για αυτό το λογο τα αναφέραμε και εδώ. Γενικά όταν θέλουμε να λάβουμε υπόψη ακολουθιακή πληροφορία αξίζει πάντα να σκεφτούμε την λύση των RNNs.

Με βάση αυτά προκύπτει το πρόβλημα της επιλογής ανάμεσα σε shallow ML μοντέλα και Deep μοντέλα. Εδώ εξαρτάται από το σενάριο χρήσης που υπάρχει καθώς και τον όγκο αλλά και το είδος των δεδομένων που έχουμε στην διάθεση μας. Γενικά αν ένα πρόβλημα μπορεί να λυθεί με shallow μοντέλα δεν υπάρχει λόγος να πάμε σε deep μοντέλα**[75]**. Όταν τα δεδομένα είναι πολλά τα shallow μοντέλα δεν μπορούν να τα χρησιμοποιήσουν αποδοτικά οπότε διαλέγουμε deep. Επίσης όταν έχουμε δεδομένα σε υψηλή διάσταση με πολλά χαρακτηριστικά πάλι η επιλογή είναι Βαθιά Μηχανική Μάθηση. Η τελευταία περίπτωση αναφέρεται στην βιβλιογραφία σαν curse of dimensionality. Άλλο ένα κριτήριο είναι το αν τα δεδομένα είναι καλά δομημένα ή όχι. Ένα παράδειγμα από το σενάριο χρήσης που εξετάζουμε σε αυτήν την εργασία είναι η περίπτωση ενός πακέτου που μεταφέρεται από το δίκτυο. Οι κεφαλίδες του πακέτου είναι παράδειγμα δομημένων δεδομένων ενώ το φορτίο(payload) του πακέτου είναι παράδειγμα μη δομημένων δεδομένων. Έτσι

για την πρώτη περίπτωση διαλέγουμε shallow μοντέλα και για την δεύτερη deep. Άλλο ένα κριτήριο που μπορούμε να αναφέρουμε είναι το αν θέλουμε να λάβουμε υπόψη την αλληλουχία γεγονότων. Σε αυτή την περίπτωση η επιλογή είναι τα deep μοντέλα και συγκεκριμένα τα RNNs.

2.4 Ασφάλεια

2.4.1 Τι είναι η ασφάλεια

Ο όρος ασφάλεια**[76]** είναι ένας όρος που εμφανίζεται πολύ συχνά την τελευταία δεκαετία. Αυτό συμβαίνει διότι πολλές από τις δραστηριότητες που πραγματοποιούν οι άνθρωποι έχουν μεταφερθεί στον εικονικό κόσμο. Πλέον οι υπολογιστές χρησιμοποιούνται για δραστηριότητες πλέον της απλής αποθήκευσης και διάθεσης στατικής πληροφορίας. Πλέον υπάρχουν δραστηριότητες όπως αποστολή και αποθήκευση αλληλογραφίας μέσω e-mail που πολλές φορές αφορά προσωπικά δεδομένα ή ιατρικά δεδομένα όπως αποτελέσματα εξετάσεων. Το email ήταν από τις πρώτες δραστηριότητες που μεταφέρθηκαν στον εικονικό κόσμο και φυσικά ακολούθησαν και άλλες. Τα κοινωνικά δίκτυα διαχειρίζονται επίσης δεδομένα που οι περισσότεροι άνθρωποι δεν θα ήθελαν να γίνουν γνωστά πολλές φορές για προσωπικούς λόγους όπως τα κοινωνικά δίκτυα γνωριμιών. Επίσης πολλές φορές άνθρωποι έπεσαν θύματα identity theft εξαιτίας διαρροής επίσημων εγγράφων ψηφιακών ή φυσικών ή απλώς επειδή έχασαν ή τους έκλεψαν το smartphone. Η τελευταία αυτή περίπτωση μαλλον θα γίνει πιο κοινή και συνηθισμένη με την νέα ψηφιακή ταυτότητα που θα τεθεί σε λειτουργία σύντομα στην Ελλάδα. Πλέον με την ψηφιακή ταυτότητα και χωρίς την συμμετοχή ανθρώπων ο πολίτης θα μπορεί να πραγματοποιεί όλες τις δραστηριότητες που πραγματοποιούσε με το κράτος ή ιδιώτες χωρίς καμία φυσική αλληλεπίδραση και με αυτόματο τρόπο. Τέτοιες δραστηριότητες περιλαμβάνουν υπογραφή εγγράφων ή αυτόματη έκδοση δανείων. Τέλος μία από τις πιο γνωστές δραστηριότητες που μεταφέρθηκαν στον εικονικό κόσμο είναι το e-banking. Πρόκειται για μία από τις πιο γνωστές και πιο ευαίσθητες δραστηριότητες και πλέον οι τράπεζες όχι απλά ενθαρρύνουν αλλά επιβάλλουν την πραγματοποιηση τους μέσω διαδικτύου. Αυτό συμβαίνει διότι η εξοικονόμηση χρημάτων από το λιγότερο προσωπικό και εγκαταστάσεις είναι μεγάλη. Επίσης η χρήση χρημάτων έχει μειωθεί πολύ τα τελευταία χρόνια και αντικαθίσταται με χρήση smart cards αντί για φυσικό νόμισμα ή ακόμα και τελείως άυλο νόμισμα όπως τα κρυπτονομίσματα. Ο συγκεκριμένος τομέας υπέφερε πολύ από παράνομες πράξεις κυρίως πριν την έλευση του chip and pin όπου αντί για smart cards χρησιμοποιούνταν μαγνητικές ταινίες με αποτέλεσμα να είναι πολύ εύκολη η κλωνοποιηση τους. Άλλος ένας τομέας ο οποίος είναι και πιο σημαντικός από αυτόν των

ηλεκτρονικών συναλλαγών είναι η ψηφιοποιημένη και αυτοματοποιημένη διαχείριση και έλεγχος κρίσιμων υποδομών. Το εύρος των κατηγοριών κρίσιμων υποδομών είναι μεγάλο. Υπο κατηγορία αποτελεί και το τραπεζικό σύστημα αλλά υπάρχουν παραδείγματα που δεν σχετίζονται με άυλα αγαθά. Τέτοια παραδείγματα είναι δίκτυα παροχής ηλεκτρικού ρεύματος δίκτυα διανομής καυσίμων εργοστάσια παραγωγής ενέργειας όπως πυρηνικά εργοστάσια και επίσης δίκτυα δημοσίων υπηρεσιών των οποίων η παροχή υπηρεσιών είναι σημαντικό να μην διακοπεί καθώς και νοσοκομείων των οποίων η διατάραξη των δραστηριοτήτων μπορεί να κοστίσει σε ανθρώπινες ζωές. Σε αυτή την κατηγορία στόχων υπάρχουν παραπάνω από λίγα πολύ σοβαρά περιστατικά που συνέβησαν τα τελευταία χρόνια. Παραδείγματα τέτοιων επιθέσεων είναι η περίπτωση του stuxnet**[77]** όπου στόχο είχε και πέτυχε να σαμποτάρει το πυρηνικό πρόγραμμα του Ιράν. Άλλο περιστατικό είναι η κυβερνοεπίθεση εναντίον του δικτύου παροχής ηλεκτρικής ενέργειας της Ουκρανίας που κατάφερε να διακόψει την παροχή υπηρεσιών για μεγάλο διάστημα. Άλλο πιο πρόσφατο παράδειγμα ήταν παράλυση δικτύου διανομής καυσίμων από μεγάλη εταιρεία σε πολλές πολιτείες των Ηνωμένων Πολιτειών για μέρες με σκοπό τον εκβιασμό και πληρωμή λύτρων. Άλλο ένα παράδειγμα που σχετίζεται με ransomware είναι η κυβερνοεπίθεση εναντίων νοσοκομείων στην Μεγάλη Βρετανία. Επίσης υπάρχουν περιστατικά που δεν είναι αποτέλεσμα κακόβουλων ενεργειών αλλά σχετίζονται με την ασφάλεια και ιδιαίτερα με την διαθεσιμότητα υπηρεσιών όπως η αποτυχία του συστήματος επικοινωνίας των ασθενοφόρων και του κέντρου ελέγχου κατα την διαδικασία μετάβασης από το παλιό σύστημα στο καινούργιο περιστατικό που κόστισε σε ανθρώπινες ζωές. Γενικά και με την έλευση και άνοδο της χρήσης κρυπτονομισμάτων οι επιθέσεις που σχετίζονται με ransomware είναι πλέον τελείως μη ανιχνεύσιμες όσον αφορά το τεχνικό κομμάτι που αφορά την πληρωμή πράγμα που κάνει τέτοιου είδους επιθέσεις πιο σοβαρές από άποψη ψηφιακής εγκληματολογίας**[78]**. Όλα αυτά αναδεικνύουν την σημασία και τον ρόλο της ασφαλειας. Πρόκειται για πεδίο της επιστήμης υπολογιστών και έχει πολλά πεδία εφαρμογής σε αυτόν τον τομέα οπότε όταν μιλάμε για ασφάλεια θα πρέπει να μιλήσουμε για όλο το εύρος ενός πληροφοριακού συστήματος αλλά και ανθρώπινων δραστηριοτήτων. Οι πιο γνωστες κατηγορίες ασφάλειας είναι αυτές της ασφάλειας δικτύων, ασφάλειας εφαρμογών, ασφάλειας τηλεπικοινωνιών και φυσικής ασφάλειας. Η περίπτωση της ασφάλειας δικτύων έχει ως στόχο την ομαλή και ασφαλή λειτουργία του υλικού. Η ασφάλεια εφαρμογών αφορά την ασφαλή χρήση και την ομαλή λειτουργία του λογισμικού. Για την ασφάλεια τηλεπικοινωνιών αφορά πάλι την ομαλή λειτουργία των επικοινωνιών. Η περίπτωση της φυσικής ασφάλειας αφορά την προστασία υλικού από κακόβουλη πρόσβαση ή καταστροφή λόγω λάθους ή φυσικής καταστροφής. Σε όλες τις περιπτώσεις οι βασικοί πυλώνες που πρέπει να προστατευθούν είναι το confidentiality, integrity, availability γνωστό ως CIA. Σε αυτήν την εργασία θα εστιάσουμε στην ασφάλεια δικτύων. Γενικά οι στόχοι της ασφάλειας είναι πολύ δύσκολο να επιτευχθούν είτε από αμυνόμενους είτε από επιτιθέμενους. Σαν αντιπαράδειγμα

μπορούμε να αναφέρουμε ότι είναι σχετικά εύκολο να επιτευχθεί ο στόχος της διαθεσιμότητας ενός τηλεπικοινωνιακού δικτύου όπως μιάς δορυφορικής ζεύξης. Στην ασφάλεια πληροφοριακών συστημάτων είναι πολύ δύσκολο να είναι κάποιος σίγουρος ή/και να μπορεί να αποδείξει κάτι παρόμοιο. Αυτό συμβαίνει διότι στα μεγάλα πληροφοριακά συστήματα και δίκτυα υπάρχει μεγάλη ποικιλότητα υλικού και λογισμικού. Στην διαχείριση και δημιουργία και χρήση και των δύο εμπλέκονται πολλοί άνθρωποι και εταιρείες πράγμα που σημαίνει ότι η πιθανότητα ύπαρξης λάθους ή αστοχίας ή παράλειψης είναι μεγάλη. Αυτό σημαίνει ότι είναι πολύ πιθανό να υπάρχουν ευπάθεις. Φυσικά είναι πολύ δύσκολο και έχει μεγάλο κόστος και για τους επιτιθέμενους και για τους αμυνομενους να βρουν αυτές τις ευπάθειες και να τις εκμεταλλευτούν ή να τις διορθώσουν.

2.4.2 Βασικές Έννοιες που σχετίζονται με την Ασφάλεια

Όπως και στην περίπτωση της Μηχανικής Μάθησης που αναφέραμε προηγουμένως είναι βασικό να παρουσιάσουμε και να δώσουμε την ερμηνεία κάποιον βασικών όρων που σχετίζονται με την ασφάλεια. Η σημασία των περισσότερων όρων που θα παρουσιάσουμε ορίζεται εδώ**[79].** Παρόλα αυτά θα δίνουμε αναφορές όπου πιστεύουμε ότι υπάρχει λόγος να διευκρινίσουμε κάτι με μεγαλύτερη λεπτομέρεια.

- Advanced Persistent Threat Advanced Persistent Protection

Αυτές οι δύο έννοιες**[80]** αφορούν την χρήση εξελιγμένων και πολύπλοκων μεθόδων για την προστασία από επιθέσεις και τις επιθέσεις αυτές που συνήθως έχουν στόχους υψηλού προφίλ, στόχους με υψηλές προδιαγραφές ασφάλειας. Αυτές οι επιθέσεις είναι δύσκολο να ανιχνευθούν από απλές στατικές υπογραφές διότι δεν είναι άμεσες επιθέσεις. Ο επιτιθέμενος εδώ στόχο έχει να μην γίνει αντιληπτός και η επίθεση προχωράει βήμα βήμα συνήθως με αργούς ρυθμούς και η επίθεση μπορεί να διαρκέσει μήνες ή ακόμα και χρόνια με την δραστηριότητα να λαμβάνει χώρο όταν παρουσιαστούν οι κατάλληλες ευκαιρίες. Σε αυτές τις περιπτώσεις και η άμυνα και η επίθεση είναι εξαιρετικά δύσκολη διαδικασία. Παράδειγμα τέτοιας επίθεσης είναι το stuxnet και η επίθεση που πραγματοποιήθηκε εναντίον του τραπεζικού συστήματος συναλλαγών swift.

- Κακόβουλο Λογισμικό(Malware)

Πρόκειται για λογισμικό που στόχο έχει να κάνει κάποια κακόβουλη ενέργεια. Αυτό το λογισμικό συνήθως είναι πολύ μικρό σε μέγεθος αλλά υπάρχουν και περιπτώσεις όπου το malware κρύβεται μέσα σε λογισμικό το οποίο πραγματοποιεί χρήσιμες νόμιμες ενέργειες. Παράδειγμα της τελευταίας περίπτωσης αποτελεί το πειρατικό λογισμικό είτε πρόκειται για λειτουργικά συστήματα είτε πρόκειται για άλλου είδους λογισμικό. Στην περίπτωση των πειρατικών λειτουργικών συστημάτων η ανίχνευση της ύπαρξης του malware και ο περιορισμός της δραστηριότητας του είναι σχεδόν αδύνατος διότι έχει πρόσβαση και ελέγχει το kernel. Για άλλου είδους πειρατικό λογισμικό η

ανίχνευση και ο περιορισμός της κακόβουλης δραστηριότητας μπορεί να επιτευχθεί με χρήση antivirus. Η κατηγορία του malware μπορεί να αναλυθεί σε πολλές υποκατηγορίες. Αυτές είναι virus, spyware, Trojan, worms.

- Botnet

Πρόκειται για ένα σύνολο υπολογιστών που βρίσκονται υπό τον έλεγχο του επιτιθέμενου κόμβοι που αναφέρονται και ως zombies. Οι πιο συχνές χρήσεις ενός botnet είναι η χρήση των κόμβων για να κρύψει ο επιτιθέμενος την πραγματική του διεύθυνση και να προβεί σε παράνομες πράξεις ή στην περίπτωση των DDoS επιθέσεων που χρησιμοποιεί τους κόμβους για να παράξει υπερβολικά μεγάλη κίνηση εναντίον ενός server με σκοπό να τον βγάλει εκτός λειτουργίας.

- Διατάραξη Λειτουργιών(Business Disruption)

  Πρόκειται για την παρεμπόδιση λειτουργίας ενός συστήματος ή μίας διεργασίας.

- Firewall

Πρόκειται για συσκευή ή λογισμικό που περιορίζει την πρόσβαση σε ένα εσωτερικό δίκτυο από άλλο εσωτερικό δίκτυο ή το Internet. Σκοπός είναι η απομόνωση και προφύλαξη του δικτύου και αυτό επιτυγχάνεται απαγορεύοντας κίνηση από το εσωτερικό δίκτυο προς το internet ή από το internet προς το εσωτερικό δίκτυο.

- Αυθεντικοποίηση(Authentication)

Πρόκειται για την διαδικασία ταυτοποίησης ενός χρήστη ενός συστήματος ο οποίος ισχυρίζεται ότι του ανήκει μία ταυτότητα. Η διαδικασία αυτή επιτυγχάνεται με διάφορα τεχνικά μέσα τα οποία γενικά χωρίζονται σε τρεις κατηγορίες. Η πρώτη αφορά κάτι που ο χρήστης ξέρει. Παραδειγματα τετοιας πληροφορίας είναι ένα PIN, ένας κωδικός πρόσβασης ή μία πληροφορία που είναι γνωστή μόνο στον χρήστη όπως ΑΦΜ ΑΜΚΑ και άλλα. Η δεύτερη κατηγορία είναι κάτι που ο χρήστης κατέχει. Παράδειγμα αυτής της κατηγορίας είναι μία smartcard ή NFC κάρτα. Η τρίτη κατηγορία είναι κάτι που ο χρήστης είναι. Παραδείγματα αυτής της περίπτωσης είναι τα βιομετρικά χαρακτηριστικά τα οποία περιλαμβάνουν δακτυλικό αποτύπωμα την ίριδα του ματιού ή την φωνή. Σε πολλές περιπτώσεις και ανάλογα το πόσο σημαντική είναι η εφαρμογή χρησιμοποιούνται και οι τρεις μέθοδοι σε συνδυασμό.

- Εξουσιοδότηση(Authorization)

Αφού έχει προηγηθεί η ταυτοποίηση ενός χρήστη με μία από τις μεθόδους που περιγράφηκαν προηγουμένως, το σύστημα πρέπει να μπορεί να προσδιορίζει σε ποιους πόρους επιτρέπεται να έχει πρόσβαση ο χρήστης και ποιες ενέργειες μπορεί να πραγματοποιήσει σε αυτούς τους πόρους. Η διαδικασία του authorization συνήθως εκφράζεται μέσω πολιτικών που προσδιορίζουν με ακρίβεια ποιά από τα εμπλεκόμενα μέρη έχουν δικαίωμα να πραγματοποιήσουν μία πράξη όπως πχ η διαγραφή/δημιουργία ενός χρήστη. Στην συνέχεια δημιουργούνται μηχανισμοί ελέγχου πρόσβασης που επιβάλλουν αυτές τις πολιτικές. Τέτοιοι μηχανισμοί ελέγχου είναι file permissions ενός

λειτουργικού συστήματος, Access Control Lists, capability tables και άλλα. Υπάρχουν τρία είδη συστημάτων Access Control, το Discretionary Access Control(DAC), Role Based Access Control(RBAC) και το Mandatory Access Control(MAC). Το DAC είναι ένα σύστημα ελέγχου πρόσβασης του οποίου η βασική ιδέα είναι οι χρήστες να γνωρίζουν ποιοι άλλοι χρήστες έχουν πρόσβαση στα αντικείμενα που τους ανήκουν για παράδειγμα αρχεία. Οι πιο συνηθισμένοι μηχανισμοί αυτής της μεθόδου είναι τα Access Control List και capability tables. Τα capability tables στην ουσία είναι πίνακες με τις γραμμές να είναι χρήστες και τις στήλες να είναι αντικείμενα πχ αρχεία ή PCs. Το RBAC σύστημα βασίζεται στη ιδέα ότι οι administrators προσδίδουν δικαιώματα με βάση τον ρόλο ενός χρήστη στο οργανισμό όπως αυτός περιγράφεται από την πολιτική αντί να δίνει δικαιώματα με βάση τον λογαριασμό του κάθε χρήστη. Μία βασική υλοποίηση αυτού του μηχανισμού σε σύγχρονα λειτουργικά συστήματα είναι η έννοια της ομάδας(Group). Αυτή η μέθοδος είναι γνωστή και ως non-Discretionary Access Control. Το MAC είναι ένα σύστημα access control που αποτελεί την πιο περιοριστική και αυστηρή λύση. Χρησιμοποιείται σε υψηλής σημασίας συστήματα όπου η ασφάλεια είναι ιδιαίτερα σημαντική. Χρησιμοποιεί μια ιεραρχική προσέγγιση για τα δικαιώματα σε αντικείμενα τα οποία αποδίδονται στους χρήστες από τους διαχειριστές. Χρησιμοποιείται η έννοια των security labels και οι χρήστες δεν μπορούν να αλλάξουν τα δικαιώματα ακόμα και σε αντικείμενα που τους ανήκουν.

- Asset

Asset ονομάζεται ένα αγαθό του οργανισμού υλικό ή άυλο το οποίο έχει αξία και πρέπει να προστατευτεί.

- Data Breach

Αυτός ο όρος περιγράφει το γεγονός όπου ένας επιτιθέμενος καταφέρνει εκμεταλλευόμενος μία ευπάθεια σε κάποιο σύστημα να αποκτήσει πρόσβαση στα αρχεία και δεδομένα αυτού του συστήματος και να έχει πλήρη έλεγχο του συστήματος και πιθανόν να χρησιμοποιήσει αυτό τον έλεγχο για να διεισδύσει περισσότερο στο πληροφοριακό σύστημα.

- Ευπάθεια(Vulnerability)

Ευπάθεια ονομάζεται μία αστοχία του λογισμικού την οποία μπορεί να εκμεταλλευτεί ένας επιτιθέμενος για να θέσει την εφαρμογή υπό τον έλεγχο του. Οι ευπάθειες τις περισσότερες φορές είναι γνωστές και η εκμετάλλευση τους συμβαίνει γιατι ο υπεύθυνος διαχειριστής δεν έχει εγκαταστήσει έγκαιρα τις ενημερώσεις. Ειδική περίπτωση ευπάθειας είναι η zero day. Αυτή η περίπτωση περιγράφει το ενδεχόμενο ανακάλυψης από τον επιτιθέμενο άγνωστης ευπάθειας. Είναι ιδιαίτερη η σημασία ενός τέτοιου ενδεχομένου διότι τα περισσότερα συστήματα ανίχνευσης επιθέσεων βασίζονται σε γνωστές ευπάθειες και αδυνατούν να αναγνωρίσουν νέες άγνωστες.

- Ψηφιακή Εγκληματολογία(Digital Forensics)

Πρόκειται για μεταφορά του όρου εγκληματολογία στον ψηφιακό κόσμο**[78]** και παίζει βασικό ρόλο στην ανίχνευση και διερεύνηση πιθανών επιθέσεων. Η διαδικασία αποτελείται από τα βήματα της εύρεσης και επιλογής στοιχείων που μαρτυρούν το πως έγινε μία επίθεση καθώς και στοιχεία που πιθανόν να οδηγήσουν στην ταυτότητα του δράστη. Πρόκειται για πολύ επίπονη και δύσκολη διαδικασία ειδικά στην περίπτωση επιτυχημένης επίθεσης που έγινε αντιληπτή μετά από αρκετό χρονικό διάστημα.

- Exploit

Το exploit αποτελεί ένα πολύ μικρό συνήθως κομμάτι δεδομένων το οποίο δίνεται σαν είσοδος σε μια εφαρμογή με σκοπό να εκμεταλλευτεί μία ευπάθεια ώστε ο επιτιθέμενος να θέσει υπό τον έλεγχο του την ευάλωτη εφαρμογή.

- Insider Threat

Προκειται για την περίπτωση όπου σε έναν οργανισμο υπάρχει ένας άνθρωπος που έχει πρόσβαση εσωτερικά του οργανισμού και έχει πρόσβαση στο πληροφοριακό σύστημα είτε φυσική ή απλά έχει ένα κάποιο επίπεδο πρόσβασης. Πρόκειται για ιδιαίτερα επικίνδυνη περίπτωση γιατί μπορεί να παρακάμψει όλα τα περιμετρικά μέτρα ασφαλείας του εσωτερικού δικτύου μέτρα ή/και να δώσει πληροφορίες για ανθρώπους του οργανισμού πληροφορία που μπορεί να φανεί χρήσιμη σε σενάρια κοινωνικής μηχανικής. Ένα πιο απλό σενάριο είναι να αποκτήσει ο επιτιθέμενος μία πρώτη πρόσβαση στο δίκτυο. Επίσης θα μπορούσε να αποκτήσει πρόσβαση σε πληροφορίες ασφαλείας στις οποίες ο επιτιθέμενος δεν θα είχε εύκολη πρόσβαση.

- Intrusion Detection System

Πρόκειται για λογισμικό ή υλικό που σκοπό έχει να ανιχνεύσει επιθέσεις την στιγμή που αυτές πραγματοποιούνται. Αυτό επιτυγχάνεται συλλέγοντας δεδομένα όπως logs εφαρμογών ή logs του λειτουργικού συστήματος. Πέρα από τα logs συνήθως και ανάλογα τί είδους IDS είναι αναλύει και τις επικεφαλίδες των πρωτοκόλλων κάθε πακέτου που εισέρχεται ή εξέρχεται από το δίκτυο. Επίσης πολλές φορές αν και αυτό είναι πιο δύσκολο και χρονοβόρο αναλύει και το φορτίο(payload) πακέτων. Με αυτού του είδους το λογισμικό θα ασχοληθούμε σε αυτήν την εργασία. Γενικά οι περισσότερες εφαρμογές IDS που χρησιμοποιούνται σήμερα βασίζονται σε στατικές υπογραφές επιθέσεων.

- Intrusion Prevention System

Πρόκειται για όρο που προέκυψε για εμπορικούς λόγους. Παρέχει την ίδια λειτουργικότητα με τα IDSs αλλά με την διαφορά ότι λαμβάνει μέτρα για να σταματήσει ή να μετριάσει τις επιπτώσεις μία επίθεσης. Παραδείγματα τέτοιων μέτρων είναι ο τερματισμός μιας σύνδεσης, η προσθήκη μιας διεύθυνσης IP σε μία blacklist ή η αναστολή λειτουργίας ενός λογαριασμού πρόσβασης σε ένα δίκτυο ή μια εφαρμογή.

- Man in the middle attack

Αυτή η επίθεση είναι μία από τις πιο συνηθισμένες επιθέσεις. Στην περίπτωση που ο επιτιθέμενος έχει πρόσβαση σε μία συσκευή που βρίσκεται στην διαδρομή ανάμεσα σε δύο κόμβους που επικοινωνούν μπορεί να υποκλέψει την επικοινωνία παραβιάζοντας το confidentiality, να παραποιησει τα δεδομένα χωρίς να το καταλάβουν οι κόμβοι παραβιάζοντας το integrity. Αυτή είναι η απλή περίπτωση. Στην περίπτωση που χρησιμοποιείται κρυπτογράφηση με την βοήθεια των Certification Authorities(CAs) αυτή η επίθεση μπορεί να ανιχνευθεί.

- Ransomware

Πρόκειται για λογισμικό που στόχο έχει να παρεμποδίσει την πρόσβαση σε δεδομένα ή συσκευές νόμιμων χρηστών και την πληρωμή χρημάτων για την επαναφορά της πρόσβασης. Αυτό συνήθως επιτυγχάνεται με χρήση κρυπτογραφίας ιδιωτικού κλειδιού.

- Attack Vector

Πρόκειται για το σύνολο όλων των τρόπων με τους οποίους ένας επιτιθέμενος μπορεί εκτελεσει μία επίθεση εναντίον του στόχου. Αυτοί οι τρόποι δεν είναι απαραίτητο να είναι τεχνικοί. όπως θα αναφέρουμε κάτι τέτοιο μπορεί να επιτευχθεί και μέσω κοινωνικής Μηχανικής. Παράδειγμα attack vectors αποτελούν τα emails μέσω των οποίων μπορεί να σταλεί ένα κακόβουλο λογισμικό ή μία ηλεκτρονική φόρμα όπου μπορεί να σταλεί ένα exploit που πραγματοποιεί μία επίθεση τύπου injection εναντίων της βάσης δεδομένων.

- Attack Surface

Πρόκειται για το σύνολο των σημείων με τα οποία μπορεί να αλληλεπιδράσει ένας επιτιθέμενος με ένα σύστημα με σκοπό να ανακαλύψει ή να εκμεταλλευτεί μία γνωστή ευπάθεια ώστε να αποκτήσει μη εξουσιοδοτημένη πρόσβαση σε ένα σύστημα. Όσο πιο πολύπλοκο είναι ένα σύστημα είτε σε λογισμικό είτε σε υλικό τόσο μεγάλο είναι και το attack surface. Στην περίπτωση του λογισμικού όσο περισσότερες τεχνολογίες χρησιμοποιούνται και όσο περισσότεροι άνθρωποι εμπλέκονται στην δημιουργία και την διαχείριση του τόσο αυξάνονται οι πιθανότητες μιας παράλειψης ή ενός λάθους. Γενικά όσο μικρότερο είναι το attack surface τόσο πιο εύκολη είναι η δουλειά των αμυνόμενων. Όσο μεγαλύτερο τόσο πιο δύσκολη θωράκιση του αλλά και η παρακολούθηση για συμβάντα ενδιαφέροντος.

- Rootkit

Πρόκειται για συλλογή προγραμμάτων τα οποία εγκαθίστανται στο kernel του λειτουργικού συστήματος. Αυτά τα προγράμματα μπορούν να έχουν διάφορες χρήσεις. Η πιο προφανής είναι η δημιουργία ενός backdoor για να αποκτήσει μόνιμη πρόσβαση ο επιτιθέμενος στην συσκευή. Φυσικά δεν σταματάει εκεί. Άλλη μία συνηθισμένη ενέργεια είναι να κρύβει διεργασίες που εκτελούνται ή ανοιχτές δικτυακές συνδέσεις. Επίσης αν ένα rootkit εγκατασταθεί πριν από ένα anti virus λογισμικό τότε το τελευταίο είναι άχρηστο διότι το rootkit δεν θα το αφήνει να έχει πρόσβαση στην πραγματική πληροφορία της δραστηριότητας. Τα rootkits εγκαθίστανται μετά από μία επιτυχημένη επίθεση και

της διαδικασίας που είναι γνωστή ως privilege escalation διαδικασία που έχει ώς στόχο την απόκτηση δικαιωμάτων υπερχρήστη στην συσκευή. Όταν επιτευχθεί αυτό ακολουθεί η εγκατάσταση του rootkit.

- Κοινωνική Μηχανική(Social Engineering)

Αυτή η μέθοδος αφορά την χρήση ψυχολογικών μεθόδων και εξαπάτησης ανθρώπων με σκοπό την πρόσβαση σε συστήματα που είναι προστατευμένα. Σε αυτήν την μέθοδο δεν χρησιμοποιούνται τεχνικές μέθοδοι όπως η εκμετάλλευση ευπαθειών. Παρόλα αυτά για να πετύχει αυτή η μέθοδος εναντίον ενός στόχου πρέπει να γίνει μία προεργασία από τον επιτιθέμενο με κύρια μέρη να είναι η συλλογή πληροφοριών και για τον οργανισμό αλλά και συγκεκριμένα τον ανθρώπινο στόχο. Το δεύτερο μέρος είναι ο καθορισμός της μεθοδολογίας που θα εφαρμοστεί ώστε να επιτευχθεί ο στόχος.

- Security Incident Response

Πρόκειται για την προκαθορισμένη διαδικασία που ακολουθεί μία επιτυχημένη επίθεση από την πλευρά του αμυνόμενου. Εδώ στόχος είναι να υπάρχουν από πρίν προβλεφθεί οι διαδικασίες και οι ενέργειες που πρέπει να εκτελεστούν για να αντιμετωπιστουν οι επιπτώσεις της επίθεσης. Στόχος είναι να μετριαστούν οι αρνητικές συνέπειες της επίθεσης καθώς και να μειωθεί ο χρόνος της ανάκαμψης. Όπως αναφέραμε αυτές οι διαδικασίες πρέπει να είναι καλά καθορισμένες πριν από την επίθεση.

- Incident Response Team

Πρόκειται για τα άτομα εκείνα που ανήκουν στον οργανισμό τα οποία είναι εξουσιοδοτημένα να αξιολογήσουν μία πιθανή επίθεση και να εκκινήσουν διαδικασίες για την αντιμετώπιση της. Ακόμα αρμοδιότητα τους είναι να βρουν πως ξεκίνησε η επίθεση, ποιοι εμπλέκονται, τον στόχο της επίθεσης και να παρουσιάσουν τα παραπάνω σε άλλα μέλη του οργανισμού με τρόπο απλό και σαφή.

- Περίμετρος Ασφαλείας(Security Perimeter)

Πρόκειται για ένα ψηφιακό όριο στο οποίο εφαρμόζεται μία πολιτική ασφαλείας και τα αντίστοιχα τεχνικά και διαχειριστικά μέτρα

- Sniffing

Πρόκειται για την διαδικασία στην οποία ο επιτιθέμενος με παθητικό τρόπο συλλέγει και επεξεργάζεται δικτυακή κίνηση μεταξύ κόμβων. Αυτή η διαδικασία είναι συνήθης σε δίκτυα τύπου ethernet ή δίκτυα ασύρματης πρόσβασης.

- Sandboxing

Πρόκειται για τεχνική μετριασμού των επιπτώσεων μιας επιτυχημένης επίθεσης. Αυτή η τεχνική είναι πολύ αποτελεσματική και χρησιμοποιείται σε πολλά σενάρια χρήσης. Ένα παράδειγμα είναι η εκτέλεση της εικονικοποίησης(virtualization) λειτουργικών συστημάτων όπου εκτελούνται στην ίδια συσκευή πολλά λειτουργικά πάνω από το host λειτουργικό. Όλα τα εικονικά λειτουργικά

είναι απομονωμενα από τα υπόλοιπα αλλά και to host και σε περίπτωση επιτυχημένης επίθεσης είναι πολύ εύκολο να διαγραφεί απλώς το το μολυσμένο σύστημα και να πραγματοποιηθεί η επανεγκατάσταση του. Επίσης πολλές φορές χρησιμοποιούνται sandboxes για να εκτελεστεί ύποπτο λογισμικό.

- Black List White List Gray List

Αυτές οι τρεις έννοιες συνδέονται συνεπώς θα τις παρουσιάσουμε μαζί. Ο όρος white list αναφέρεται στην περίπτωση όπου επιτρέπεται η πρόσβαση σε ένα σύστημα μόνο σε όσους βρίσκονται στην λίστα και μπλοκάρει όλους τους άλλους. Ο όρος black list αναφέρεται στην περίπτωση όπου απαγορεύεται η πρόσβαση σε όλους όσους βρίσκονται στην λίστα και την επιτρέπουν σε όλους τους άλλους. Ο όρος grey list αναφέρεται στην περίπτωση όπου προσωρινά επιβάλλονται περιορισμοί είτε καθολικοί είτε μερικοί δεδομένου ενός γεγονότος που προκάλεσε υποψία για μη εξουσιοδοτημένη δράση ενός actor μέχρι να πραγματοποιηθεί μία ανάλυση των γεγονότων ώστε να γίνει σαφές αν πρόκειται για false alarm ή πραγματική παράνομη δραστηριότητα. Οι λίστες αυτές θα μπορούσαν να καταγράφουν ονόματα χρηστών συγκεκριμένες διευθύνσεις IP ή IPs από συγκεκριμένες χώρες. Οι συγκεκριμένες μέθοδοι είναι πολύ αποτελεσματικές απέναντι σε επιθέσεις Denial of Service.

- Siem

Ο όρος αυτός σημαίνει Security Information and Event Management. Τα συστήματα αυτά είναι η βάση για το πως γίνεται η ανίχνευση των επιθέσεων στις περισσότερες περιπτώσεις. Πρόκειται για μια σαφώς καθορισμένη διαδικασία για τον τρόπο με τον οποίο η ασφάλεια του οργανισμού παρακολουθείτε και αξιολογείται. Τα συστήματα αυτά βοηθούν στο να αναγνωρίζονται αυτόματα συστήματα του οργανισμού που δεν συμμορφώνονται με τις πολιτικές που βρίσκονται σε ισχύ καθώς και για την άμεση ενημέρωση της Incident Response Team για πιθανά συμβάντα παραβίασης της ασφάλειας.

- Zero Day Vulnerability

Πρόκειται για υποκατηγορία του όρου ευπάθεια. Είναι πιο συγκεκριμένος όρος και αναφέρεται σε ευπάθεια που γίνεται αντιληπτή για πρώτη φορά από έναν οργανισμό αλλά και από τους κατασκευαστές του λογισμικού. Με βάση τον τρόπο που λειτουργούν αυτήν τη στιγμή τα περισσότερα συστήματα IDS οι zero day ευπάθειες είναι δύσκολο να ανιχνευθούν και είναι ιδιαίτερα αποτελεσματικές.

- Phishing

Πρόκειται για μία πολύ απλή επίθεση που βασίζεται στην κοινωνική μηχανική και στην εξαπάτηση νόμιμων χρηστών μία εφαρμογής, ενός δικτύου ή ενός πληροφοριακού συστήματος με σκοπό να αποκτήσουν στοιχεία πρόσβασης. Αυτή η επίθεση συνήθως περιλαμβάνει την χρήση περιεχομένου που μοιάζει με νόμιμο όπως για παράδειγμα μίας σελίδας εισόδου σε μία εφαρμογή

που είναι ίδια με την αληθινή. Επίσης στην περίπτωση της κοινωνικής μηχανικής ο επιτιθέμενος μπορεί να προσποιείται ένα υπαρκτό πρόσωπο και να καταφέρει να πείσει τον στόχο να προβεί σε ενέργειες όπως αποκάλυψη στοιχείων πρόσβασης ή αλλαγή τους

- Κυβερνο επίθεση(Cyber Attack)

Πρόκειται για πολύ γενικό όρο και περιλαμβάνει μία πληθώρα από ιεραρχικά κατηγοριοποιημένες επιθέσεις. Στις κατηγορίες και στις συγκεκριμένες επιθέσεις θα αναφερθούμε σε ξεχωριστή ενότητα στην συνέχεια. Εδώ απλά θα αναφέρουμε ότι γενικά κυβερνοεπίθεση σημαίνει η προσπάθεια με τεχνικά ή άλλα μέσα η απόπειρα να αποκτήσει πρόσβαση ο επιτιθέμενος με σκοπό να διαταράξει το confidentiality, integrity και availability.

- Φυσική Ασφάλεια(Physical Security)

Πρόκειται για κατηγορία ασφάλειας η οποία συνήθως είναι υπο εκτιμημένη. Οι περισσότεροι οργανισμοί δίνουν απόλυτη έμφαση στα μέτρα που τους προστατεύουν από απομακρυσμένες επιθέσεις κυρίως μέσω διαδικτύου και δεν δίνουν σημασία στην απλούστερη περίπτωση όπου ο επιτιθέμενος έχει φυσική πρόσβαση σε έναν υπολογιστή ή συσκευή δικτύου.

2.4.3 Σκοπός της Ασφάλειας

Με τον όρο ασφάλεια συστημάτων εννοείται η προστασία υλικών ή άυλων αγαθών που είναι αποθηκευμένα ή μεταφέρονται από πληροφοριακά συστήματα ή δίκτυα**[76]**. Η προστασία αυτή αφορά την αποφυγή καταστροφής παραποίησης κλοπής δεδομένων ή τη διατάραξη ομαλής λειτουργίας πληροφοριακών συστημάτων και δικτύων. Πιό συγκεκριμένα και όπως αναφέρεται πολύ συχνά στην βιβλιογραφία στόχος είναι να διασφαλιστεί το CIA το οποίο σημαίνει Confidentiality Integrity Availability. Αυτές οι έννοιες αποτελούν τον κορμό που οδηγεί την ανάπτυξη λύσεων για την κυβερνοασφάλεια. Επίσης θα μιλήσουμε για μέτρα που λαμβάνονται για την επίτευξη αυτών των στόχων. Υπάρχουν δύο ειδών μέτρα τα τεχνικά και τα διαχειριστικά**[81]**. Τα τεχνικά μέτρα αφορούν λογισμικό ή υλικό που συνήθως στόχο έχει να εμποδίσει ένα γεγονός ή να αναγνωρίσει ένα γεγονός παραδείγματα είναι η χρήση firewalls και IDSs και antivirus. Τα διαχειριστικά μέτρα αφορούν την δημιουργία και επιβολή πολιτικών, κανόνων συμπεριφοράς και επιμόρφωσης των εμπλεκόμενων μερών, μέτρα τα οποία έχουν στόχο να προλαμβάνουν πιθανά επικίνδυνα σενάρια. Παράδειγμα διαχειριστικών μέτρων αποτελεί η επιβολή της πολιτικής ισχυρών κωδικών πρόσβασης από τους χρήστες των συστημάτων, η εκπαίδευση των χρηστών σε επιθέσεις τύπου phishing καθώς και ο τακτικός έλεγχος(audit). Η καταλληλότητα της κάθε κατηγορίας εξαρτάται από το σενάριο χρήσης αλλά τις περισσότερες φορές η καλύτερη λύση περιλαμβάνει μέτρα και από τις δύο κατηγορίες. Με λίγα λόγια η μία κατηγορία συμπληρώνει την άλλη. Στην συνέχεια δίνουμε ορισμούς για αυτές τις τρεις έννοιες**[82]**.

- Εμπιστευτικότητα(Confidentiality)

Αυτή η έννοια περιγράφει την ανάγκη ενός οργανισμού και τις προσπάθειες ώστε δεδομένα που είναι αποθηκευμένα ή μεταφέρονται να παραμείνουν ασφαλή και κρυφά. Υπάρχουν πολλοί λόγοι ύπαρξης αυτής της απαίτησης και αναφέραμε στην αρχή της ενότητας κάποιους από αυτούς. Αφορά προσωπικά δεδομένα, ιατρικά δεδομένα, επιχειρηματικά ή κρατικά μυστικά. Για να επιτευχθεί αυτό πρέπει να υπάρχουν μηχανισμοί πρόσβασης που να περιορίζουν την πρόσβαση με καταλληλα τεχνικά μέτρα μόνο στα εμπλεκόμενα μέρη που έχουν δικαίωμα πρόσβασης. Τα μέτρα αυτά πρέπει να προστατεύουν από εσκεμμένη ή τυχαία διαρροή δεδομένων. Διαρροή δεδομένων δεν αποτελεί μόνο η αποκάλυψη και πρόσβαση ενός επιτιθέμενου μέρους σε δεδομένα ενός οργανισμού. Το σενάριο μπορεί να επεκταθεί σε μη νόμιμη πρόσβαση ενός μέλους του οργανισμού σε δεδομένα που δεν θα έπρεπε να έχει πρόσβαση για παράδειγμα ηλεκτρονικοί ιατρικοί φάκελοι που πρέπει να είναι προσβάσιμοι μόνο σε ιατρικό προσωπικό ενός νοσοκομείου γίνονται προσβάσιμοι σε διοικητικό προσωπικό. Για να επιτευχθεί αυτό αυστηρά μέτρα πρέπει να βρίσκονται σε ισχύ ώστε να περιορίζουν την πρόσβαση. Υπάρχουν πολλοί τρόποι που μπορεί να το πετύχει αυτό ένας επιτιθέμενος. Θα μπορούσε να πραγματοποιήσει μία επίθεση κατά της εφαρμογής μέσω της οποίας γίνονται διαθέσιμα τα δεδομένα αυτά και να δοκιμάσει όλες τις γνωστές επιθέσεις όπως injection, memory corruption, cross site scripting man in the middle και άλλα. Επίσης μπορεί να δοκιμάσει να εκμεταλλευτεί συγκεκριμένες γνωστές ευπάθειες που υπάρχουν στο λογισμικό για παράδειγμα στον web server και δεν έχουν τις πρόσφατες ενημερώσεις ασφάλειας από τον υπεύθυνο administrator λόγω παράλειψης. Τέλος υπάρχουν εκτός από τις επιθέσεις που στοχεύουν την ίδια την εφαρμογή και οι επιθέσεις εναντίων χρηστών. Αυτές οι επιθέσεις βασίζονται σε μεγάλο βαθμό σε τεχνικές κοινωνικής μηχανικής και όχι σε καθαρά τεχνικά θέματα. Εδώ σκοπός είναι να καταφέρει ο επιτιθέμενος να αποσπάσει κωδικούς πρόσβασης ή να καταφέρει να πείσει τον στόχο να εκτελέσει σε υπολογιστή εσωτερικά του δικτύου κακόβουλο λογισμικό. Αυτές οι επιθέσεις είναι ιδιαίτερα επικίνδυνες διότι μπορούν να παρακάμψουν μέτρα περιμετρικής ασφάλειας του οργανισμού όπως τα firewalls και στην συνέχεια επειδή συνήθως τα μέτρα ασφάλειας που λαμβάνονται εσωτερικά του δικτύου δεν είναι το ίδιο αυστηρά σε σχέση με τα περιμετρικά μέτρα ασφάλειας ή ακόμα είναι και ανύπαρκτα μπορεί να οδηγήσει σε πλήρη έλεγχο από τον επιτιθέμενο κάθε μηχανήματος στο πληροφοριακό σύστημα. Κάποια από τα πιο κοινά μέτρα για να επιτευχθεί το confidentiality είναι η επιμόρφωση η επιβολή επιλογής ισχυρών κωδικών πρόσβασης, η επιβολή συχνής αλλαγής τους η επιβολή two factor authentication κάτι τα τελευταία χρόνια έχει γίνει πολύ δημοφιλές και άλλα. Οσον αφορα το two factor authentication μπορεί να επιτευχθεί με βιομετρικά στοιχεία όπως δακτυλικό αποτύπωμα, αναγνώριση προσώπου ή ένα δεύτερο token που αποστέλλεται σε άλλη συσκευή ή λογαριασμό(email). Αυτά τα μέτρα που αναφέραμε είναι τα πιο απλά και προφανή. Άλλα μέτρα που μπορούν να ληφθούν είναι ο τρόπος που παράγονται τα tokens πρόσβασης όπως είναι τα cookies για

web εφαρμογές. Είναι βασικό αυτά τα tokens να μην μπορούν να προβλεφθούν από τον επιτιθέμενο γιατί έχουν ως αποτέλεσμα να μηδενίσουν την αξία των ισχυρών κωδικών πρόσβασης. Επίσης ο χρόνος ζωής των tokens πρέπει να είναι όσο το δυνατόν περιορισμένος. Φυσικά όλα αυτά τα μέτρα έχουν αξία στην περίπτωση όπου η πληροφορία πρέπει να είναι διαθέσιμη για κάποια μέρη. Όταν αυτή η υπόθεση δεν ισχύει μία απλή λύση είναι η διατήρηση της πληροφορίας σε συσκευές που ο χρήστης έχει μόνο φυσική πρόσβαση και δεν είναι συνδεδεμένες σε κάποιο προσβάσιμο δίκτυο.

- Ακεραιότητα(Integrity)

Η έννοια αυτή αφορά την διαβεβαίωση ότι τα δεδομένα δεν έχουν παραποιηθεί είτε κατά την αποθήκευσή τους είτε κατά την διάρκεια που μεταδίδονται. Η ακεραιότητα των δεδομένων επιτυγχάνεται μόνο όταν τα δεδομένα είναι ακριβή αυθεντικά και αξιόπιστα. Η παραποίηση είναι τις περισσότερες φορές σκόπιμη και στόχο έχει το προσωπικό όφελος ή την παραπλάνηση και εξαπάτηση ενός ανθρώπου. Για παράδειγμα κάποιος θα μπορούσε να στοχεύσει την παραποίηση της τιμής ενός προϊόντος πριν προχωρήσει στην πληρωμή με αποτέλεσμα να εξαπατήσει τον έμπορο. Επίσης κάποιος θα μπορούσε να παραποιησει τα στοιχεία επικοινωνίας ενός μέλους ενός οργανισμού στην ιστοσελίδα με αποτέλεσμα αντί για το νόμιμο πρόσωπο να επικοινωνεί με τον επιτιθέμενο και να αποκαλύψει δεδομένα τα οποία δεν θα έπρεπε να αποκαλυφθούν. Επίσης η παραποίηση των δεδομένων μπορεί να γίνει κατά λάθος ή από ατύχημα. Για παράδειγμα κάποιος μπορεί να διαγράψει ή αλλάξει δεδομένα λόγω ανθρώπινου λάθους. Οι τρόποι και μέθοδοι που μπορεί να επιτευχθεί ο στόχος του integrity είναι με χρήση κρυπτογραφίας, συναρτήσεις σύνοψης, ψηφιακές υπογραφές και ψηφιακά πιστοποιητικά. Για παράδειγμα σε μία ιστοσελίδα μπορούν να χρησημοποιηθούν πιστοποιητικά ελεγμένα από Certification Authorities ώστε ο επισκέπτης του ιστότοπου να είναι σίγουρος ότι βλέπει το αυθεντικό περιεχόμενο. Εκτός από την διασφάλιση της ακεραιότητας βασικό σημείο είναι και η αναγνώριση του γεγονότος ότι κάποιος προσπάθησε να παραποιησει τα δεδομένα ώστε τα ενδιαφερόμενα μέρη να γνωρίζουν ότι δέχθηκαν επίθεση. Αυτό μπορεί να επιτευχθεί με τα εργαλεία κρυπτογραφίας που αναφέρθηκαν. Επίσης υποκατηγορία στόχου που κατατάσσεται στην κατηγορία του integrity είναι και το non-repudiation. Ο όρος αυτός σημαίνει ότι ένα εμπλεκόμενο μέρος σε μία διαδικασία όπως για παράδειγμα μία συναλλαγή ή υπογραφή ενός εγγράφου δεν μπορεί στο μέλλον να ισχυριστεί ότι δεν πραγματοποίησε μία πράξη. Και αυτό επιτυγχάνεται με χρήση κρυπτογραφίας.

- Διαθεσιμότητα(Availability)

Αυτή η κατηγορία δεν αφορά συγκεκριμένα τα δεδομένα όπως η προηγούμενες δύο. Στις προηγούμενες έχουν κεντρικό ρόλο ενω σε αυτήν είναι απλά ένα υπομέρος. Ο όρος availability σημαίνει ότι κάθε συστατικό μέρος του συστήματος πρέπει να είναι διαθέσιμο στους χρήστες που απευθύνεται και να μην υπερβεί ένα προκαθορισμένο χρόνο μη διαθεσιμότητας. Η μη διαθεσιμότητα μπορεί να οφείλεται σε πολλούς παράγοντες όπως συντήρηση διακοπή ρεύματος ή διακοπή μιας

τηλεπικοινωνιακής ζεύξης. Στον τομέα των δορυφορικών επικοινωνιών για παράδειγμα υπάρχουν προδιαγραφές και απαιτήσεις που προβλέπουν την διακοπή της ζεύξης είτε λόγο κακού καιρού είτε σφαλματος στον δορυφόρο. Στην περίπτωση ενός πληροφοριακού συστήματος είναι πιο δύσκολο να γίνει αυτό διότι υπάρχουν πολλές παράμετροι που μπορούν να οδηγήσουν σε διακοπή της υπηρεσίας. Παραδείγματα τέτοιων περιστατικών που οφείλονται σε κακόβουλες ενέργειες δώσαμε προηγουμένως. Έτσι εδώ εκτός από την διαθεσιμότητα των δεδομένων έχουμε και την διαθεσιμότητα δικτυακών συνδέσεων όπως οπτικές ίνες, διαθεσιμότητα δικτυακών συσκευών όπως routers και διαθεσιμότητα servers. Όποιο συστατικό μέρος και να αποτύχει το αποτέλεσμα είναι ίδιο διακοπή της υπηρεσίας. Ο στόχος της διαθεσιμότητας είναι πιο δύσκολος να επιτευχθεί σε σχέση με τους δύο προηγούμενους διότι προυποθέτει την συνεργασία πολλών εμπλεκόμενων μερών όπως τεχνικών δικτύων administrators και προγραμματιστών. Με βάση αυτά μπορούμε να πούμε όπως και στο confidentiality και το integrity ότι μπορεί να γίνει είτε από κακόβουλη ενέργεια είτε από ανθρώπινο σφάλμα είτε από φυσική καταστροφή. Στην περίπτωση της κακόβουλης ενέργειας ο επιτιθέμενος μπορεί να στοχεύσει ένα οποιοδήποτε συστατικό μέρος ώστε να πετύχει την διακοπή της υπηρεσίας. Μία από τις πιο γνωστές και εύκολες επιθέσεις που έχουν αυτό τον στόχο είναι η Distributed Denial of Service στην οποία θα αναφερθούμε αναλυτικά σε επόμενη ενότητα. Στην περίπτωση του ανθρώπινου λάθους υπάρχουν πολλά σενάρια που μπορούν να συμβούν όπως καταστροφή υλικού ή λογισμικού λόγω απροσεξίας ή έλλειψης γνώσης. Στην περίπτωση φυσικής καταστροφής υπάρχουν επίσης πολλά σενάρια όπως φωτιά πλημμύρα διακοπή ηλεκτροδότησης και άλλα.

Αυτές οι τρεις έννοιες αποτελούν τους πυλώνες της ασφάλειας και όλα τα μέτρα που έχουν σκοπό να ανιχνεύσουν και μετριάσουν ή να αποτρέψουν τις επιπτώσεις τέτοιων γεγονότων αποτελούν τον σκοπό της ασφάλειας.

2.4.4 Είδη επιθέσεων και Επιπτώσεις

Το πλήθος των κυβερνοεπιθέσεων είναι μεγάλο και για αυτό τον λόγο πολλοί έχουν προσπαθήσει να τις κατηγοριοποιήσουν**[83,84]**. Οι επιθέσεις γίνονται όλο και πιο πολύπλοκες και είναι δύσκολο να κατηγοριοποιηθούν. Για αυτό τον λόγο είναι χρήσιμο να δημιουργηθεί μία ταξινομία με σκοπό να βοηθήσει τα εμπλεκόμενα μέρη ενός οργανισμού να κατανοήσουν καλύτερα τους κινδύνους και τα μέτρα που χρειάζονται ώστε να αντιμετωπίσουν τέτοια συμβάντα. Μία τέτοια ταξινομία είναι ένα πολύ χρήσιμο εργαλείο για την κατηγοριοποίηση και καλύτερη κατανόηση των επιθέσεων με αποτέλεσμα την καλύτερη προετοιμασία και γενικότερα αντιμετώπιση των απειλών από έναν οργανισμό. Υπάρχουν διάφορες προσεγγίσεις για την κατηγοριοποίηση κάποιες από τις οποίες θα παρουσιάσουμε εδώ. Όπως αναφέρεται στο**[83]** ορίζουν κάποιες απαιτήσεις που θα πρέπει

να ικανοποιεί η μέθοδος και κατηγοριοποιούν τις επιθέσεις με βάση 5 κριτήρια. Οι απαιτήσεις όπως αναφέρονται πρέπει να είναι οι εξής.

- Πρώτον η ταξινομία πρέπει να είναι Αποδεκτή(Accepted) δηλαδή να βασίζεται σε υπάρχουσες μεθόδους που είναι δοκιμασμένες και αποδεκτές στον χώρο της κυβερνοασφάλειας.
- Δεύτερον πρέπει να είναι Mutually Exclusive. Αυτό σημαίνει ότι κάθε επίθεση θα πρέπει να κατηγοριοποιείται μόνο σε μία κατηγορία ώστε να μην υπάρχει επικάλυψη ανάμεσα στις κατηγορίες.
- Τρίτον πρέπει να είναι Κατανοητή(Comprehensible) .Με αυτόν τον όρο εννοούν ότι θα πρέπει να είναι κατανοητή και σαφής και από ειδικούς αλλά και από εμπλεκόμενα μέρη χωρίς προχωρημένες τεχνικές γνώσεις.
- Τέταρτον θα πρέπει να είναι Πλήρης/Εξαντλητική(Complete/Exhaustive). Αυτό σημαίνει ότι οι κατηγορίες καλύπτουν όλες τις πιθανές επιθέσεις.
- Πέμπτον πρέπει να είναι Ξεκάθαρη(Unambiguous). Αυτό σημαίνει ότι δεν θα πρέπει να μην υπάρχει αμφιβολία η σύγχυση για την κατηγορία που ανήκει μία επίθεση.
- Έκτον πρέπει οι όροι να είναι καλά ορισμένοι. Αυτό σημαίνει ότι οι κατηγορίες θα πρέπει να είναι καλά ορισμένες και οι όροι να είναι συμβατοί με την κοινότητα της ασφάλειας.

Αυτές οι απαιτήσεις αναφέρονται γενικά για τις ταξινομίες κυβερνοασφάλειας και δεν αφορούν μόνο στην περίπτωση που αναφέραμε. Πρέπει να λαμβάνονται υπόψη σε κάθε προσπάθεια δημιουργίας μιας νέας μεθοδολογίας.

Τα κριτήρια με βάση τα οποία πραγματοποιείται η κατηγοριοποίηση όπως αναφέρθηκε είναι πέντε και τα παρουσιάζουμε παρακάτω**[83]**.

- Κατηγοριοποίηση με βάση τον τρόπο επίθεσης(Classification by attack vector)

Εδώ η κατηγοριοποίηση πραγματοποιείται με βάση το μονοπάτι το οποίο ο επιτιθέμενος ακολουθεί ώστε να αποκτήσει πρόσβαση σε ένα σύστημα. Τον όρο Attack vector τον ορίσαμε σε προηγούμενη ενότητα. Αυτή η κατηγορία περιλαμβάνει πολλές περιπτώσεις κάποιες από τις οποίες είναι Misconfiguration, kernel flaws, buffer overflows,Insufficient Input Validation, Symbolic Links, File Descriptor, Race Condition, Incorrect File/Directory Permission,Social Engineering.

- Κατηγοριοποίηση με βάση την επιχειρησιακή επίπτωση(Classification by Operational Impact)

Εδώ η κατηγοριοποίηση γίνεται με βάση την επιχειρησιακή επίπτωση που θα έχει μία επίθεση στον οργανισμό και στόχος είναι να δώσει μία γενική ιδέα στα εμπλεκόμενα μέρη για την επίπτωση της επίθεσης. Μερικά παραδείγματα είναι τα εξής. Misuse of Resources,User Compromise,Root Compromise,Web Compromise,Installed Malware,Denial of Service.

- Κατηγοριοποίηση με βάση την Άμυνα(Classification by Defense)

Εδώ η κατηγοριοποίηση γίνεται με βάση τα μέτρα που παίρνει ο αμυνόμενος για να αντιμετωπίσει μία πιθανή επίθεση. Εδώ παρουσιάζονται διάφορες τεχνικές τις οποίες μπορεί να αξιοποιήσει ο αμυνόμενος και για την περίπτωση πριν από την επίθεση αλλά και μετά από την επίθεση. Μερικά παραδείγματα είναι τα ακόλουθα. Mitigation, Remediation. Η περίπτωση του Mitigation αφορά τα μέτρα που μπορεί να λάβει ο αμυνόμενος ώστε να μετριάσει τα αποτελέσματα και τις επιπτώσεις μίας επίθεσης. Αυτή η μέθοδος αφορά μέτρα που λαμβάνονται πριν γίνει μία επίθεση. Σαν παράδειγμα αναφέρεται η περίπτωση ενός worm που μεταδίδεται μέσω του δικτύου. Σε αυτήν την περίπτωση μέτρα όπως το white listing μπορούν να μπλοκάρουν ή να καθυστερήσουν την εξάπλωση μέχρι να παρθούν άλλα μέτρα. Η περίπτωση του Remediation αφορά μέτρα που λαμβάνονται επίσης πριν την διαδικασία του exploitation που στόχο έχουν να εμποδίσουν μία γνωστή ευπάθεια. Μέτρα αυτής της κατηγορίας είναι η πρόσθεση των διορθώσεων που παρέχει ο κατασκευαστής του λειτουργικού συστήματος ή η δημιουργία μίας διόρθωσης όπως ενός εκτελέσιμου ή μίας web εφαρμογής για μία αστοχία/λάθος που ανακαλύφθηκε κατα την διάρκεια ενός προγραμματισμένου audit.

- Classification by Informational Impact

Σε αυτήν την περίπτωση η κατηγοριοποίηση πραγματοποιείται με βάση τις επιπτωσεις που έχει μία πιθανή επίθεση στις προγραμματισμένες δραστηριότητες του συστήματος. Αυτή η κατηγορία περιλαμβάνει τις ακόλουθες περιπτώσεις, Distort, Disrupt, Destruct, Disclosure, Discovery.

- Κατηγοριοποίηση με βάση τον στόχο(Classification by Attack Target)

Εδώ η κατηγοριοποίηση πραγματοποιείται με βάση το υπο σύστημα που στοχεύει ο επιτιθέμενος. Εδώ κάποια παραδείγματα είναι τα εξής. Operating System, Network, Local, User, Application.

Επίσης άλλη μία δουλειά που στην ουσία επεκτείνει αυτή που αναφέραμε προηγουμένως είναι η εξής**[84]**. Αυτή η μεθοδολογία προσπαθεί να εστιάσει στην αδυναμία άλλων να εστιάσουν στην πτυχή της άμυνας σε επιθέσεις. Αυτή η ταξινομία προσθέτει τα παρακάτω features πέρα από αυτα που αφορούν για την προηγούμενη.

- Ημερομηνία Επίθεσης(Date of Attack)

  Ποιά μέρα πραγματοποιήθηκε η επίθεση
- Ημερομηνία Ανακάλυψης(Date of Discovery)

  Ποιά μέρα ο οργανισμός ανακάλυψε ότι κάποιο σύστημα παραβιάστηκε.
- Ημερομηνία Ανακοίνωσης(Date of Announcement)

Ποιά μέρα ανακοίνωσε ο οργανισμός σε ενδιαφερόμενα μέρη ότι η ασφάλεια τους παραβιάστηκε.

- Breached Before

Αυτό το feature αφορά το ενδεχόμενο η επίθεση που καταγράφηκε να είναι συνέχεια μιας άλλης επίθεσης.Είναι ιδιαίτερα σημαντικό σε σενάρια Advanced Persistent Attacks.

- Τομέας(Sector)

Αφορά το αν ο οργανισμός είναι ιδιωτικός ή κυβερνητικός.

- Στρατηγική Άμυνας(Defense Strategy)

Αφορά ποιά μέτρα θα μπορούσαν να έχουν ληφθεί ώστε να είχε αποφευχθεί η επίθεση. Αναφέρονται οι εξής επιλογές. Deception, Frustration, Resistance, Recognition and Recovery.

- Άνθρωπος εκ των έσω(Insider Job)

Αφορά το αν υπήρξε άνθρωπος εκ των έσω. Αυτός θα μπορούσε να έχει δράσει μόνος ή να βοήθησε με την θέληση του τον επιτιθέμενο

- Χώρες που Επηρεάστηκαν(Countries Affected)

Αν η επίθεση έλαβε χώρα σε ένα κράτος ή πολλά.

- Απώλεια Χρημάτων(Money Loss)

Πόσα χρήματα κόστισε άμεσα και έμμεσα η επίθεση.

- Αριθμός Ανθρώπων που επηρεάστηκαν(People Affected)

Πόσους ανθρώπους επηρέασε η επίθεση. Για παράδειγμα πόσοι λογαριασμοί χρηστών παραβιάστηκαν.

Αυτή η μεθοδολογία**[84]** αποτελεί μία πολύ χρήσιμη προσέγγιση για την σωστή οργάνωση και ανάλυση των πιθανών επιθέσεων παρόλο που υπάρχουν και άλλες. Πολλές φορές σε έναν οργανισμό δημιουργούνται ταξινομίες που ταιριάζουν καλύτερα σε ένα σενάριο χρήσης. Για παράδειγμα υπάρχει μία ταξινομία που αφορά συστήματα SCADA. Αυτά τα συστήματα ελέγχουν φυσικές συσκευές όπως για παράδειγμα εργοστάσια ηλεκτροπαραγωγής. Η ταξινομία αυτή**[85]** αναγνωρίζει τις ιδιαιτερότητες και διαφορές τέτοιων συστημάτων και προσφέρει μία πιο συγκεκριμένη λύση για αυτήν την περίπτωση. Επίσης άλλο μία ακόμη πολύ γνωστή ταξινομία που ακολουθεί μία εντελώς διαφορετική προσέγγιση από την πρώτη περίπτωση και κατηγοριοποιεί τις επιθέσεις με βάση το αν παραβιάζουν το confidentiality availability integrity είναι αυτή**[86].**Στην συγκεκριμένη εργασία δεν θα υιοθετήσουμε την συγκεκριμένη μεθοδολογία αλλά αντί για αυτό θα ακολουθήσουμε μία πιο απλή προσέγγιση η οποία κατηγοριοποιεί τις επιθέσεις με βάση τις επιπτώσεις. Αυτό το κάνουμε διότι στην περίπτωση του σκοπού της εργασίας δεν έχει ιδιαίτερη σημασία πως κατηγοριοποιούνται οι επιθέσεις. Επίσης είναι πιο εύκολο να καταλάβει ο αναγνώστης το dataset που θα χρησιμοποιήσουμε για την υλοποίηση της εφαρμογής. Η ταξινομία που χρησιμοποιούμε είναι αυτό που ορίζεται εδω**[87]**. Σε αυτήν την προσέγγιση οι επιθέσεις κατηγοριοποιούνται με βάση τις επιπτώσεις που έχουν. Οι επιπτώσεις αυτές χωρίζονται σε τέσσερις

κατηγορίες αυτές είναι οι Denial of Service, Remote to Local, User to Root και Probe. Ακολουθεί η επεξήγηση τους.

- Denial of Service(DoS)

Μία επίθεση τύπου Denial of Service**[79]** έχει στόχο να θέσει εκτός λειτουργίας ένα κόμβο ή ένα δίκτυο με συνέπεια οι νόμιμοι χρήστες του δικτύου ή της υπηρεσίας να μην έχουν πρόσβαση. Αυτού του είδους οι επιθέσεις παρόλο που δεν έχουν συνήθως σαν συνέπεια την απόκτηση παράνομης πρόσβασης σε ένα σύστημα ή αποκάλυψη δεδομένων που θα έπρεπε να παραμείνουν κρυφά ή άλλων δεδομένων αποτελούν μεγάλο κίνδυνο για έναν οργανισμό με συνέπεια απώλεια χρόνου και χρημάτων. Για παράδειγμα μία επίθεση τέτοιου είδους σε έναν εμπορικό οργανισμό στοιχίζει σε πωλήσεις και ανθρωποώρες καθώς αποτελεί και πλήγμα στην φήμη του. Επίσης μία τέτοια επίθεση μπορεί να αποτελέσει μόνο ένα υπο μέρος μιας σοβαρότερης επίθεσης όπως αυτή του evil twin. Σε αυτήν την επίθεση αρχικά γίνεται επίθεση Denial of Service σε ένα νόμιμο wireless access point ώστε όλοι οι χρήστες να αποσυνδεθούν και στην συνέχεια να συνδεθούν σε ένα άλλο access point που βρίσκεται υπο τον έλεγχο του επιτιθέμενου γεγονός που δίνει στον επιτιθέμενο να συνεχίσει με μια επίθεση Man in the Middle ώστε να υποκλέψει την επικοινωνία. Αυτου΄του είδους οι επιθέσεις συνήθως γίνονται με δύο τρόπους. Ο πρώτος αφορά την εξάντληση των δικτυακών ή επεξεργαστικών πόρων με αποτέλεσμα την αδυναμία παροχής της υπηρεσίας. Για παράδειγμα μία υπηρεσία που εκτελείται σε ένα δίκτυο με σύνδεση ADSL μπορεί να κατακλυστεί από συνδέσεις που προέρχονται από έναν επιτιθέμενο με σύνδεση Fiber. Αυτή η κατηγορία συνήθως ονομάζεται flooding. Ο δεύτερος τρόπος αφορά την διακοπή της υπηρεσίας μέσω παρεμπόδισης λειτουργίας του λογισμικού. Αυτό επιτυγχάνεται συνήθως με τον επιτιθέμενο να εκμεταλλεύεται κάποια ευπάθεια όπως ένα buffer overflow που οδηγεί το λογισμικό της υπηρεσίας ή ολόκληρο το λειτουργικό σύστημα στο οποίο εκτελείται η υπηρεσία να αποτύχει και να βγει εκτός λειτουργίας. Μία άλλη πολύ γνωστή περίπτωση είναι αυτή όπου ο ίδιος κόμβος που ελέγχει ο επιτιθέμενος εκκινεί συνδέσεις χωρίς ποτέ να τις ολοκληρώσει με αποτέλεσμα να εξαντληθούν οι πόροι του server. Μία ευρέως γνωστή υπο περίπτωση της πρώτης μεθόδου είναι η επίθεση Distributed Denial of Service. Εδώ ο επιτιθέμενος αξιοποιεί τον έλεγχο που έχει αποκτήσει σε botnets και τα αξιοποιεί με σκοπό να πετύχει το ίδιο αποτέλεσμα με την πρώτη περίπτωση. Πρόκειται για δύσκολη επίθεση από την σκοπιά της αντιμετώπισης διότι δεν αρκει ένα απλό blacklisting μίας IP και η ανίχνευση του επιτιθέμενου είναι αδύνατη. Θύματα τέτοιων επιθέσεων είναι συνήθως κυβερνητικοί οργανισμοί, μεγάλοι εμπορικοί οργανισμοί , τραπεζικοί οργανισμοί και άλλα.

- Remote to Local(R2L)

Αυτή η κατηγορία**[79]** περιλαμβάνει όλες τις περιπτώσεις επιθέσεων όπου ο επιτιθέμενος προσπαθεί να αποκτήσει πρόσβαση σε ένα κόμβο ενός δικτύου από απομακρυσμένο σημείο. Για να μπορεί να επιτευχθεί αυτό ο οργανισμός στόχος θα πρέπει να έχει σε λειτουργία κάποια υπηρεσία

προσβάσιμη από το internet. Το είδος της υπηρεσίας δεν παίζει ιδιαίτερο ρόλο αλλά γενικά όσο περισσότερες και όσο πιο πολύπλοκες είναι οι υπηρεσίες τόσο περισσότερο πιθανό είναι ο επιτιθέμενος να βρει μία ευπαθεια ή μία αστοχία στη διαμόρφωση(configuration). Αυτή η ανακάλυψη πληροφορίας από τον επιτιθέμενο επιτυγχάνεται με την μέθοδο banner grabbing όπου αλληλεπιδρώντας με την εφαρμογή ανακαλύπτει λεπτομέρειες όπως ποιά εφαρμογή είναι, ποιά έκδοση είναι ή αν έχει τα πιο πρόσφατα patches εγκατεστημένα. Παραδείγματα υπηρεσιών είναι mail servers, web applications, VPNs,Remote Desktop, ssh και άλλα. Ο επιτιθέμενος στόχο έχει συνήθως να βρει μία υπάρχουσα ευπάθεια ώστε να πετύχει τον στόχο. Αν δέν υπάρχει ευπάθεια μπορεί να εξαπολύσει μία επίθεση brute force ώστε να προσπαθήσει να αποκτήσει διαπιστευτήρια(credentials) νόμιμων χρηστών οι οποίοι δεν έχουν επιλέξει δυνατούς κωδικούς. Το πιο σοβαρό είδος επίθεσης τέτοιου ονομάζεται pre authentication και αφορά ευπάθεια που υπάρχει πρίν αναγκαστεί ο επιτιθέμενος να κάνει log in στην εφαρμογή.

- User to Root(U2R)

Σε αυτό το είδος**[79]** επιθέσεων γίνεται η υπόθεση ότι ή ο επιτιθέμενος έχει αποκτήσει πρόσβαση με απομακρυσμένη επίθεση ή έχει φυσική πρόσβαση σε ένα κόμβο. Αυτή η κατηγορία ονομάζεται User to Root επειδή σε λειτουργικά συστήματα UNIX ο υπερχρήστης έχει πάντα το όνομα χρήστη root. Ένας πιο σωστός όρος που περιγράφει είναι privilege escalation. Σε αυτήν την κατηγορία ο επιτιθέμενος προσπαθεί να αποκτήσει δικαιώματα υπερχρήστη ξεκινώντας με διαπιστευτήρια χρήστη που έχει λιγότερα δικαιώματα με σκοπό να θέσει το σύστημα υπό τον πλήρη έλεγχο του. Αυτό μπορεί να επιτευχθεί με πολλούς τρόπους. Ένας από τους πιο συνηθισμενους είναι η εκμετάλλευση μιας υπάρχουσας ευπάθειας σε ένα εκτελέσιμο που έχει δικαιώματα υπερχρήστη. Η ύπαρξη τέτοιων εκτελέσιμων είναι πολύ συνηθισμένη για τις περιπτώσεις που πρέπει μία νόμιμη ενέργεια να εκτελεστεί με δικαιώματα υπερχρήστη. Αποτελεί το πιο συνηθισμένο ενδεχόμενο στην διαδικασία που είναι γνωστή ως rooting σε smartphones. Ένας άλλος τρόπος είναι η αλλαγή παραμέτρων σε configuration files του λειτουργικού συστήματος ή μίας εφαρμογής. Σε αυτήν την κατηγορία επιθέσεων εκμεταλλευόμαστε την λάθος ρύθμιση των δικαιωμάτων ενός αρχείου που περιέχει σημαντικές παραμέτρους για την εκτέλεση ενός προγράμματος που έχει δικαιώματα υπερχρήστη. Αλλάζοντας τις παραμέτρους αναγκάζουμε το πρόγραμμα να εκτελεσει ενέργειες που κανονικά δεν θα έπρεπε να κάνει.

Τέλος μία τρίτη περίπτωση αφορά την περίπτωση όπου όταν ένα εκτελέσιμο εκκινεί και φορτώνει βιβλιοθήκες πχ DLLs ψάχνει για αυτά τα αρχεία ιεραρχικά σε πολλούς φακέλους μέχρι να βρεί το αρχείο. Αν ο επιτιθέμενος καταφέρει να τοποθετήσει ένα δικό του DLL που περιέχει κακόβουλο κώδικα σε έναν φάκελο πιο πάνω στην ιεραρχία από τον φάκελο που βρίσκεται το νόμιμο τότε η εφαρμογή θα φορτώσει και θα εκτελέσει το κακόβουλο λογισμικό με δικαιώματα υπερχρήστη. Φυσικά υπάρχουν και άλλες τεχνικές αλλά αυτές που αναφέραμε είναι οι πιο γνωστές.

- Probe

Αυτή η κατηγορία**[79]** αναφέρεται ως επίθεση αλλά στην πράξη αποτελεί την προ εργασία που πρέπει να γίνει πριν εξαπολύσει μία επίθεση ο επιτιθέμενος. Γενικός στόχος αυτών των επιθέσεων είναι η συλλογή πληροφοριών για τον στόχο. Πιο συγκεκριμένα προσπαθεί να συλλέξει πληροφορίες για το λογισμικό που χρησιμοποιείται και τις γνωστές ευπαθειες που υπάρχουν σε αυτό. Το πρώτο βήμα αυτής της διαδικασίας είναι η συλλογή πληροφοριών για τους ανθρώπους που εμπλέκονται στον οργανισμό και το ρόλο που έχουν σε αυτό. Αυτό το βήμα μπορεί να οδηγήσει σε μία πολύ επιτυχημένη επίθεση κοινωνικής μηχανικής. Κύριοι στόχοι αποτελούν τεχνικό προσωπικό και κυρίως administrators καθώς και υψηλόβαθμο διοικητικό προσωπικό. Το δεύτερο βήμα αποτελεί την ανακάλυψη όλων των κόμβων ενός δικτύου. Αυτό το βήμα είναι γνωστό και ως mapping. Στην συνέχεια και για κάθε κόμβο που έχει αναγνωριστεί λαμβάνει χώρα το port scanning. Εδώ και για κάθε μηχάνημα και με passive ή active να αναγνωρίσουμε ποιά ports είναι ανοιχτά. Στην συνέχεια ακολουθεί η διαδικασία του banner grabbing την οποία αναφέραμε και προηγουμένως. Οι υπηρεσίες που τρεχουν σε ένα σύστημα και όταν ο επιτιθέμενος αλληλεπιδρά με αυτές αποκαλύπτουν στοιχεία. Αυτά τα στοιχεία αποκαλύπτουν πληροφορίες όπως πιο λογισμικό τρέχει, για παράδειγμα web server. Ποιά έκδοση του λογισμικού τρέχει, αν έχουν εγκατασταθεί οι πιο πρόσφατες ενημερώσεις ασφαλείας. Γενικότερα η συλλογή τέτοιας πληροφορίας δεν γίνεται με αυτόν τον τρόπο. Θα μπορούσε να συλλεχθεί με οποιαδήποτε αλληλεπίδραση όπως για παράδειγμα η εσκεμμένη δημιουργία λάθους στην εφαρμογή όπως η χρήση λάθος ονόματος χρήστη. Αυτή η διαδικασία που περιγράφηκε συνήθως είναι πολύ θορυβώδης. Αυτό σημαίνει πως είναι πολύ εύκολη η ανίχνευση και αντιμετώπιση της. Για παράδειγμα σε ένα μεγάλο δίκτυο με πολλά hosts η διαδικασία του port scanning θα παρήγαγε κίνηση εκατοντάδων megabytes. Στην περίπτωση του penetration testing όπου ο επιτιθέμενος δεν ενδιαφέρεται να μην γίνει αντιληπτός αυτό δεν αποτελεί πρόβλημα. Σε μία πραγματική επίθεση όμως και ειδικά στις περιπτώσεις Advanced Persistent Attacks πρέπει να ακολουθηθεί άλλη μεθοδολογία και με πιο προσεκτικό και μη θορυβώδη τρόπο να συλλεχθούν οι πληροφορίες. Μέθοδοι για αυτή την προσέγγιση είναι η διαδικασία να απλωθεί στον χρόνο, να χρησιμοποιηθεί host το οποίο παράγει παρόμοια κίνηση και σε είδος αλλά και σε όγκο ή το scanning να πραγματοποιηθεί κατανεμημένα δηλαδή να χρησιμοποιηθούν πολλοί κόμβοι του δικτύου.

Στον παρακάτω πίνακα παρουσιάζονται οι επιθέσεις που υπάρχουν και η κατηγορία που ανήκει η κάθε μία. Δεν θα αναλύσουμε το πως δουλεύει η κάθε μία από αυτές τις επιθέσεις.

TABLE VII : MAPPING OF ATTACK CLASS WITH ATTACK TYPE

Attack Class	Attack Type
DoS	Back, Land, Neptune, Pod, Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm (10)
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint (6)
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named (16)
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps (7)

2.4.5 Παρακολούθηση Πόρων και γεγονότων(Resource and event monitoring)

Όλα τα συστήματα που βρίσκονται σε λειτουργία έχουν τρόπους να καταγράφουν γεγονότα που μπορεί να έχουν σημασία στο μέλλον. Η σημασία αυτή μπορεί να καλύπτει πολλές περιπτώσεις και λόγους. Για παράδειγμα οι τραπεζικοί οργανισμοί καταγράφουν όλες τις συναλλαγές ώστε να μπορεί να ανιχνευθεί η πορεία του χρήματος κάτι τέτοιο θα μπορούσε να γίνει για νομικούς λόγους. Μία δεύτερη περίπτωση είναι η καταγραφή της δραστηριότητας ενός base station κινητής τηλεφωνίας ώστε αν πραγματοποιηθεί ένα έγκλημα η αστυνομία να μπορεί έχει στοιχεία για το ποιός βρισκόταν στην συγκεκριμένη γεωγραφική περιοχή τον συγκεκριμένο χρόνο. Άλλη μία περίπτωση είναι η διατήρηση καταγραφών(logs) με σκοπό να πραγματοποιούνται τακτικοί έλεγχοι(audits). Αυτή η πληροφορία που συσσωρεύεται δημιουργεί προκλήσεις σχετικά με την αποτελεσματική αξιοποίηση της για τους σκοπούς της ανίχνευσης των επιθέσεων. Για τους σκοπούς της εργασίας και το σενάριο χρήσης μας ενδιαφέρει το πως δίκτυα, λειτουργικά συστήματα και εφαρμογές που παρέχουν υπηρεσίες και εφαρμογές που παρακολουθούν την δραστηριότητα που εκτελεί η κάθε υπηρεσία όπως τα SIEMs**[88,89,90]** στα οποια αναφερθήκαμε προηγουμένως και θα αναφερθούμε και στην συνέχεια πιο αναλυτικά, παράγουν και αποθηκεύουν δεδομένα. Γενικά μας ενδιαφέρουν τρεις διαδικασίες. Αυτές είναι το logging το monitoring και το alerting**[91]**. Με βάση αυτές τις 3 έννοιες και δίνοντας τον ορισμό για κάποιες βασικές έννοιες θα εξηγήσουμε τη διαδικασία που ακολουθείται από την στιγμή που ένα γεγονός συμβαίνει μέχρι την κατηγοριοποίηση του ώς κυβερνο επίθεση ή νόμιμη δραστηριότητα. Αυτή η διαδικασία αποτελεί κορμό της όλης διαδικασίας ανίχνευσης επιθέσεων είτε αυτή γίνεται με αυτόματο ημι αυτόματο ή τελείως manual τρόπο.

- Καταγραφή(Logging)

Αυτή η διαδικασία αποτελεί το πρώτο μέρος. Εδώ απλά καταγράφεται πληροφορία που αφορά ένα γεγονός. Εδώ εστιάζουμε στην περίπτωση όπου συλλέγουμε πληροφορία για γεγονότα που αφορούν την ασφάλεια. Ο υπεύθυνος, αυτός που σχεδιάσε ή αυτός που χρησιμοποιεί τον μηχανισμό αυτό πρέπει να δώσει έμφαση σε πολλές σχεδιαστικές παραμέτρους. Η πρώτη είναι πόσο αναλυτική πληροφορία θα διατηρείται για ένα γεγονός Αυτό σημαίνει ότι για παράδειγμα αν μία εφαρμογή τερματιστει με μη κανονικό τρόπο θα καταγραφεί απλώς το γεγονός ή και άλλες παράμετροι όπως το stack trace και ποιος χρήστης χρησιμοποιεί την εφαρμογή την στιγμή του γεγονότος. Αυτή η σχεδιαστική παράμετρος έχει να κάνει με τους σκοπούς σε κάθε σενάριο χρήσης. Για το σενάριο χρήσης αυτής της εργασίας ιδανικά θα θέλαμε να καταγράφονται εκείνα τα δεδομένα τα οποία χρειάζεται το μοντέλο Μηχανικής Μάθησης που θα σχεδιάσουμε ώστε να πραγματοποιήσει την προβλεψη. Η δεύτερη σχεδιαστική παράμετρος αφορά το πόσο θα πρέπει να διατηρούνται οι πληροφορίες που καταγράφονται. Αυτή η παράμετρος πολλές φορές δεν αποτελεί επιλογή αλλά υποχρέωση του οργανισμού που απορρέει από νομικές υποχρεώσεις. Γενικά τα logs από κρίσιμες σε σχέση με την ασφάλεια εφαρμογές όπως η δραστηριότητα γραμμής εντολών ενός υπερχρήστη ή υλικό όπως firewalls πρέπει να διατηρούνται περισσότερο χρόνο. Άλλη μια σχεδιαστική παράμετρος είναι τα logs που διατηρούνται να βρίσκονται αποθηκευμένα σε άλλους κόμβους συσκευές που δεν έχει πρόσβαση η εφαρμογή που παρακολουθείται ώστε στην περίπτωση μίας επιτυχημένης επίθεσης να μην μπορούν να καταστραφούν.

- Παρακολούθηση(Monitoring)

Το να καταγράφεται κάθε δραστηριότητα σε ένα σύστημα δεν είναι αρκετό για να αξιοποιηθεί η πληροφορία που υπάρχει στα logs. Ένα γεγονός που καταγράφηκε από μόνο του μπορεί να μην έχει κάποια σημασία ή νόημα αλλά σε συνδυασμό με κάποια άλλα μπορεί να γίνει ξεκάθαρο ότι πρόκειται για μία σοβαρή επίθεση που βρίσκεται σε εξέλιξη. Ο τρόπος αξιοποίησης αυτών των συσχετίσεων είναι βασικός στην ανίχνευση κυβερνοεπιθέσεων. Τρία είδη λογισμικού χρησιμοποιούνται για αυτή την διαδικασία αλλά τις περισσότερες φορές οι δυνατότητες τους και η λειτουργικότητα που προσφέρουν δεν σταματάει σε αυτό. Αυτά είναι το Complex Event Processing, τα SIEMs και τα Log Management Systems.

Το Complex Event Processing**[92,93]** στόχο έχει να αναγνωρίσει γεγονότα τα οποία οποία αναφέρονται σαν complex events από άλλα γεγονότα που αναφέρονται σαν απλά γεγονότα(simple events). Σε αυτό το λογισμικό δημιουργείται μία ροή από απλά γεγονότα και το CEP προσπαθεί να ανιχνεύσει ένα σύνθετο γεγονός(complex event) όπως αυτό έχει οριστεί.

Τα SIEM**[88,89]** παρέχουν μία αντίστοιχη λειτουργικότητα αλλα δεν περιορίζονται σε αυτή. Ένα SIEM μπορεί να χρησιμοποιηθεί για πολλές λειτουργίες και θα αφιερώσουμε ξεχωριστή ενότητα. Εδώ απλά αναφέρουμε ότι μπορεί να αναλύσει και να αποθηκεύσει μεγάλο όγκο

πληροφοριών και με βάση υπογραφές να ανιχνεύσει πολύπλοκα γεγονότα. Τέλος τα Log Management Systems είναι λογισμικό που στόχο έχει την πιο εύκολη ανάλυση μεγάλου όγκου δεδομένων από καταγραφές. Όταν λέμε πιο εύκολη εννοούμε πιο εύκολη συγκριτικά με απλά εργαλεία γραμμής εντολών**[94]** όπως το grep,awk,tail,cut και regular expressions τύπου Perl. Επίσης είναι πιο αποδοτικά από άποψη χρησιμοποίησης πόρων σε σχέση με τα εργαλεία γραμμής εντολών.

- Ενημέρωση(Alerting)

Το τελευταίο στάδιο στην διαδικασία που περιγράφεται είναι η ενημέρωση μιας ομάδας ή ενός διαχειριστή για ένα γεγονός ή η τροφοδότηση του γεγονότος σε ένα ακόμη λογισμικό ή η περαιτέρω επεξεργασία του. Για παράδειγμα για την περίπτωση των Log Management Systems όταν μία πολύπλοκη αναζήτηση επιστρέψει ένα ή παραπάνω αποτελέσματα μπορεί αυτόματα να ενημερωθεί ένας διαχειριστής. Για την περίπτωση των SIEMs μπορεί να σταλεί στην ομάδα των αναλυτών ώστε να αξιολογηθεί περαιτέρω. Για την περίπτωση του CEP μπορεί να προωθηθεί σε ένα μοντέλο μηχανικής μάθησης για να γίνει το inference.

Για την πρώτη φάση της διαδικασίας δηλαδή το logging πρέπει να σημειωθεί ότι το τι καταγράφεται by default από ένα λειτουργικό σύστημα ή μία εφαρμογή/διεργασία διαφέρει σε κάθε περίπτωση. Η αναλυτική παρουσίαση των μηχανισμών για κάθε περίπτωση είναι πέρα από τους σκοπούς της εργασίας. Εδώ θα ασχοληθούμε με την περίπτωση του Linux.

- Linux Logging Mechanisms**[95]**

Το Linux έχει έναν ξεχωριστό φάκελο όπου αποθηκεύει τα logs τον /var/log. Σε αυτόν τον φάκελο αποθηκεύονται κάθε είδους logs. Γενικά τα logs που παράγονται και αποθηκεύονται μπορούν να χωριστούν σε τέσσερις κατηγορίες. Αυτές είναι οι εξής. Application Logs, Event Logs, System Logs και Kernel Logs. Στην συνέχεια παρουσιάζουμε αναλυτικά τον κάθε υποφάκελο και τι καταγράφει.

/var/log/auth.log

Αυτός ο φάκελος καταγράφει όλα τα γεγονότα που σχετίζονται με την αυθεντικοποίηση(authentication). Αυτό περιλαμβάνει αποτυχημένες ή επιτυχημένες προσπάθειες αυθεντικοποίησης, την δραστηριότητα του υπερχρήστη καθώς και όλα τα γεγονότα που παράγει το Pluggable Authentication Module(PAM). Το PAM σκοπό έχει να επιβεβαιώνει την ταυτότητα και να αναλαμβάνει την αυθεντικοποίηση για εφαρμογές που το χρειάζονται με κεντρικοποιημένο τρόπο.

/var/log/dmesg

Σε αυτόν τον φάκελο καταγράφεται πληροφορία που αφορά μηνύματα από οδηγούς συσκευών. Τα μηνύματα διαγράφονται σε κάθε τερματισμό του συστήματος.

/var/log/boot.log

Σε αυτόν τον φάκελο καταγράφονται γεγονότα που αφορούν την εκκίνηση ή τον τερματισμό του συστήματος.

/var/log/kern.log

Σε αυτόν τον φάκελο καταγράφονται γεγονότα που προέρχονται από το kernel.

/var/log/syslog

Αυτός ο φάκελος καταγράφει όλα τα πιθανά γεγονότα του συστήματος που δεν σχετίζονται με τις προηγούμενες περιπτώσεις που αναφέραμε. Περισσότερα για το syslog θα αναφέρουμε στην συνέχεια.

/var/log/cron

Σε αυτόν τον φάκελο καταγράφονται γεγονότα που σχετίζονται με προγραμματισμένες εργασίες.

/var/log/apt

Σε αυτόν το φάκελο καταγράφονται γεγονότα που σχετίζονται με εγκατάσταση λογισμικού με χρήση του εργαλείου apt που είναι διαθέσιμο στο Ubuntu.

/var/log/lastlog

Εδώ καταγράφονται λεπτομέρειες που αφορούν το τελευταίο login του κάθε χρήστη.

Παρουσιάσαμε τα βασικά που που καταγράφονται από το Linux αλλά υπάρχουν και άλλα. Για παράδειγμα πολλές εφαρμογές δημιουργούν δικό τους φάκελο μέσα στον φάκελο /var/log. Παράδειγμα είναι ο Apache web server και το MySQL.

Στην συνέχεια παρουσιάζουμε τον τρόπο που μπορεί να επεκταθεί αυτός ο μηχανισμός καταγραφής. Αυτός ο τρόπος είναι το syslog στο οποίο κάναμε μία αναφορά προηγουμένως.

SysLog

Πρόκειται για ένα πρότυπο που σκοπό έχει να δημιουργεί και να αποθηκεύει ή να μεταδίδει logs. Όταν αναφερόμαστε στο syslog αναφερόμαστε σε τρία διαφορετικά πράγματα. Το πρώτο είναι το syslog service. Αυτό το service δέχεται και επεξεργάζεται μηνύματα. Παρακολουθεί τα γεγονότα δημιουργώντας ένα socket στον φάκελο /var/log όπου όπως αναφέραμε και προηγουμένως μπορούν να γράφουν διάφορες εφαρμογές. Στην συνέχεια μπορεί να αποθηκεύει τα logs σε αρχείο για να τα επεξεργαστεί ή να τα στείλει σε κάποιον sever. Το δεύτερο είναι το πρωτόκολλο syslog(RFC 5424) το οποίο είναι ένα πρωτόκολλο μεταφοράς που προσδιορίζει το πως θα μεταδίδονται τα logs στο δίκτυο. Είναι επίσης ένα format που προσδιορίζει το πως είναι δομημένα τα μηνύματα. Το τρίτο είναι ένα μήνυμα syslog. Το μήνυμα αυτό όπως είπαμε έχει συγκεκριμένο format και αποτελείται από κεφαλίδα και περιεχόμενο. Κάποια από τα πιο σημαντικά πεδία του μηνύματος είναι τα εξής. Timestamp, Hostname, AppName, Priority. Μία εναλλακτική του syslog αποτελεί το Systemd.

Να σημειωθεί ότι αυτά που αναφέραμε αφορούν τον λειτουργικό σύστημα Ubuntu. Σε άλλα συστήματα τύπου UNIX όλα αυτά μπορούν να διαφέρουν. Επίσης ειδαμε ότι το λειτουργικό σύστημα ναι μεν καταγράφει πληροφορία αλλά στο σενάριο χρήσης και το dataset με το οποίο θα

ασχοληθούμε χρειαζόμαστε συγκεκριμένη πληροφορία για κάθε process που τρέχει σε τακτά χρονικά διαστήματα. Για παράδειγμα χρειαζόμαστε τον αριθμό των προσπαθειών για root πρόσβαση καθώς και τον αριθμό των bytes που έστειλε και δέχτηκε μία διεργασία μεταξύ άλλων. Ακόμα και αν καταγράφεται και είναι διαθέσιμη αυτή η πληροφορία για μία διεργασία φαίνεται να είναι αντιπαραγωγικό να επεξεργαζόμαστε όλα τα logs για να την αποκτησουμε. Αντί για αυτού θα ήταν καλύτερα να έχουμε διεργασίες οι οποίες παράγουν και μεταδίδουν όλη αυτή την πληροφορία που χρειαζόμαστε για να τροφοδοτούμε το μοντέλο Μηχανικής Μάθησης. Περισσότερα θα αναφέρουμε σε επόμενη ενότητα.

2.5 Complex Event Processing

Ο όρος CEP έγινε δημοφιλής μετά από αυτή την δουλειά**[96]**. Πριν από αυτό όπως αναφέρεται**[97]** δεν υπήρχε σαν όρος από μόνος του αλλά είχε βάσεις σε πολλά ανεξάρτητα ερευνητικά πεδία. Κάποια από αυτά είναι discrete event simulation, active databases, network management, temporal reasoning.

Επεξεργασία Γεγονότων(Event Processing) είναι η διαδικασία επεξεργασίας και εξαγωγής γνώσης από την πραγματικού χρόνου επεξεργασία και ανάλυση ροών δεδομένων που εξάγονται από ένα πλήθος πηγών. Αυτά τα γεγονότα αναφέρονται στην βιβλιογραφία σαν απλά γεγονότα(simple events). Παραδείγματα simple events θα μπορούσε να είναι μία συναλλαγή ή ένα αποτυχημένο login σε έναν server. Στην συνέχεια σκοπός είναι να συνδυαστούν και συσχετιστούν μεταξύ τους ή/και σε σχέση με τον χρόνο τέτοια απλά γεγονότα ώστε να αναγνωριστούν άλλα γεγονότα που μπορεί να έχουν ενδιαφέρον για τον οργανισμό όπως μία κυβερνοεπίθεση. Αυτά τα γεγονότα αναφέρονται ως complex events. Εδώ πρέπει να τονίσουμε ότι όλη αυτή αυτή η διαδικασία μπορεί να λειτουργήσει ιεραρχικα. Αυτό σημαίνει ότι όταν ανιχνευθεί ένα complex event αυτό μπορεί να χρησιμοποιηθεί ώς simple event στην ανίχνευση ενός πιο πολύπλοκου γεγονότος. Αυτή η τεχνολογία υπάρχει πολλά χρόνια και βρίσκει χρήση σε πολλούς τομείς. Παραδείγματα είναι credit card fraud, business activity monitoring, security monitoring καθώς και την ανίχνευση αόριστων(high level) γεγονότων στην παρακολούθηση πλοίων. Βασικός στόχος είναι η επεξεργασία και ο συσχετισμός γεγονότων σε πραγματικό χρόνο και η έγκαιρη ενημέρωση των ενδιαφερόμενων ώστε η κατάσταση που ανιχνεύθηκε να αντιμετωπιστεί το συντομότερο.

Μία μεγάλη διαφορά ανάμεσα στο CEP και την περίπτωση όπου μία αποθήκη δεδομένων(repository) χρησιμοποιείται από μία γλώσσα ώστε να απαντάει σε ερωτήματα(queries) είναι ότι στην δεύτερη περίπτωση παρουσιάζεται η ανάγκη να αποθηκευτεί μεγάλος όγκος πληροφορίας σε κάποια βάση δεδομένων. Αντίθετα το CEP στόχο έχει να επεξεργάζεται τις ροές

δεδομένων σε πραγματικό χρόνο χωρίς την ανάγκη να αποθηκεύει όλη την εισερχόμενη πληροφορία. Αυτό επιτυγχάνεται με το να ορίζει σαφώς γεγονότα ενδιαφέροντος και να τα τροφοδοτεί με απλά γεγονότα όταν αυτά καταυθάνουν. Όταν και και αν ικανοποιηθούν εκτελείται μία ενέργεια. Αυτή η ενέργεια θα μπορούσε να είναι μία ειδοποίηση σε έναν administrator. Αυτή η προσέγγιση μειώνει δραστικά την καθυστέρηση(latency) και κάνει αυτήν την τεχνολογία ιδανική για εφαρμογές πραγματικού χρόνου. Με βάση αυτά πρέπει να τονίσουμε όπως θα δούμε και στην υλοποίηση ότι τα λογισμικά CEP μπορούν να συνδυάζουν real time πληροφορία με ιστορική πληροφορία αν αυτή είναι διαθέσιμη. Στην περίπτωση όπου ένα complex event ανιχνευθεί καλό είναι και συνήθως γίνεται να αποθηκεύονται τα δεδομένα ώστε να μπορεί να εξηγηθεί το πως προέκυψε αυτό το σύνθετο γεγονός(complex event). Η αρχιτεκτονική του CEP ορίζεται ως event driven**[97]** και βασικο ρόλο παίζουν ο όρος γεγονός(event) τον οποίο ορίσαμε προηγουμένως. Επίσης ο όρος event engine αναφέρεται στον τρόπο με τον οποίο τα απλά γεγονότα χρησιμοποιούνται. Τέλος ο όρος δράση(action) αναφέρεται στην ενέργεια που πραγματοποιείται όταν προκύψει ένα σύνθετο γεγονός.

Στην συνέχεια παρουσιάζουμε το μοντέλο 4D**[98]** το οποίο αποτελεί ένα framework που έχει δημιουργηθεί με αρχιτεκτονική πραγματικού χρόνου για τον σκοπό της αυτόματης λήψης αποφάσεων με τρόπο proactive. Ο όρος proactive αναφέρεται στην ικανότητα να αποφύγουμε ή να μετριαστούν οι επιπτώσεις ενός αρνητικού γεγονότος που λαμβάνει χώρα στο παρόν ή θα λάβει χώρα στο άμεσο μέλλον ή αν πρόκειται για θετικό γεγονός να μπορέσουμε εγκαίρως να εκμεταλλευτούμε μία ευκαιρία. Στην περίπτωση χρήσης της εργασίας αυτής που αφορά την αντιμετώπιση των συνεπειών μιας επίθεσης. Το μοντέλο 4D αποτελείται από τέσσερις έννοιες και βοηθάει στο να κατανοήσουμε την χρησιμότητα του CEP. Τα τέσσερα Ds είναι Detect, Derive, Decide και Do.

Detect

Αυτή η φάση αφορά την συλλογή ανάκτηση και αποθήκευση πληροφορίας για ένα σύστημα.

Derive

Αυτή η φάση αφορά την εξαγωγή νέας γνώσης ή πολύπλοκων γεγονότων τα οποία συμπαιρένονται με βάση πληροφορία που συλλέχθηκε και συσχετίστηκε στην προηγούμενη φάση.

Decide

Αυτή η φάση αφορά την λήψη μιας απόφασης proactivly με βάση την γνώση που προέκυψε από την προηγούμενη φάση. Εδώ υποθέτουμε την ύπαρξη μιας υπηρεσίας που έχει ως σκοπό την πρόταση πιθανών ενεργειών. Αυτόν το ρόλο μπορεί να τον αναλάβει μία ομάδα ανθρώπων ή ένα υπολογιστικό μοντέλο.

Do

Αυτή η φάση αφορά την υλοποίηση και πραγματοποίηση της ενέργειας καθώς και την παρατηρηση των αποτελεσμάτων που προέκυψαν από την εκτέλεση της.

Με βάση αυτό το μοντέλο το CEP ασχολείται με τις φάσεις Detect kai Derive.

Το CEP επίσης μοιράζεται έννοιες και λειτουργικότητα με άλλα συστήματα που είναι γνωστά σαν Event Stream Processing**[99]**. Αυτά τα συστήματα έχουν στόχο όπως και το CEP να αναλύσουν μία ροή από γεγονότα και να εφαρμόσουν μία επεξεργασία σε αυτά. Το CEP θα μπορούσε να οριστεί ότι είναι ένα ένα ESP μαζί με ένα event correlation engine**[97]**.

Γενικά οι απαιτήσεις**[100]** που προκύπτουν για ένα CEP engine είναι τρεις. Αυτές είναι το high throughput το low latency και complex computations. Οι έννοιες αυτές περιγράφονται στην συνέχεια.

High throughput

Αυτό σημαίνει ότι η εφαρμογή θα πρέπει να μπορεί να αναλύσει μεγάλο όγκο δεδομένων σε πραγματικό ή σε σχεδόν πραγματικό χρόνο.

Low Latency

Αυτό σημαίνει ότι η εφαρμογή θα πρέπει να ολοκληρώνει την επεξεργασία σε σχεδόν πραγματικό χρόνο.

Complex Computations

Αυτό σημαίνει ότι η εφαρμογή θα πρέπει να υποστηρίζει πολύπλοκους υπολογισμούς στα δεδομένα που επεξεργάζεται. Αυτό περιλαμβάνει υπολογισμούς όπως πολύπλοκα patterns(event correlation), φιλτράρισμα γεγονότων, συνάθροιση γεγονότων με βάση τον χρόνο η την αλληλουχία συνεχόμενων γεγονότων και άλλα.

Διαδικασία Επεξεργασίας(Processing Pipeline)

Ένα σύστημα CEP**[92,97]** πρέπει να καθορίσει τα παρακάτω. Πρώτον πρέπει να καθορίσει τις ροές δεδομένων(streams) που θα δέχεται. Αυτά μπορούν να είναι τα logs που παράγουν διάφορες διεργασίες που τρέχουν και θέλουμε να παρακολουθούμε τις ενέργειες τους. Αυτά τα logs μπορούν να παράγονται με τους τρόπους που παρουσιάσαμε σε προηγούμενη ενότητα. Δεύτερον πρέπει να ορίσει τα είδη των events που θα επεξεργάζεται. Αυτά τα events είναι αυτά που αναφέρονται ως απλά γεγονότα(simple events) και για κάθε πληροφορία που λαμβάνουμε από τα logs θα δημιουργούμε και ένα αντίστοιχο γεγονός. Αφού καθοριστούν αυτά τα γεγονότα δημιουργούνται τα statements ή αλλιώς EPL Queries. Αυτά είναι τα σύνθετα γεγονότα(complex events) που αναφέραμε προηγουμένως. Στις γλώσσες EPL θα αναφερθούμε στην συνέχεια. Αφού δημιουργήσουμε όλα τα complex events που μας ενδιαφέρουν πάνω στις ροές δεδομένων(datastreams) δημιουργούμε listeners ένα για κάθε statement. Οι listeners ενεργοποιούνται όταν ικανοποιηθούν τα statements και πραγματοποιούν μία ενέργεια. Αυτή η ενέργεια μπορεί να είναι οτιδήποτε. Από την αποστολή ενός notification, ή την προώθηση της πληροφορίας σε ένα επόμενο υπολογιστικό μέρος όπως το μοντέλο μηχανικής μάθησης στην περίπτωση αυτής της εργασίας μέχρι την αυτόματη ενεργοποίηση ενός μηχανισμού. Επίσης όταν ανιχνευτεί ενα complex event μπορεί να εισαχθεί σε ένα άλλο stream όπου

το event που ανιχνεύθηκε παίζει το ρόλο ενός απλού γεγονότος. Αυτό βοηθάει στην ιεραρχική δημιουργία άλλων σύνθετων γεγονότων που δεν αποτελούνται μόνο από simple events αλλά αποτελούνται και από complex events.

Γλώσσα EPL(EPL Language)

Πρόκειται για μία γλώσσα η οποία μοιάζει με την SQL αλλά αποτελεί μία επέκταση της. Είναι η γλώσσα στην οποία γράφονται τα queries ή αλλιώς τα statements όπως τα ορίσαμε πρίν. Αυτό το συστατικό μέρος είναι υπεύθυνο και προσφέρει πολύπλοκη λειτουργικότητα ώστε να υποστηρίζονται οι απαιτήσεις που ορίζονται από το CEP δηλαδή τους σύνθετους υπολογισμούς . Σε αντίθεση με γλώσσες που χρησιμοποιούνται σε βάσεις δεδομένων και δεν παρέχουν λειτουργικότητα που να επιτρέπει χρονική και συναθροιστική λογική και αποτελεί δουλειά του προγραμματιστή να την υλοποιήσει, η EPL την υποστηρίζει. Γενικά η διαφορά ανάμεσα στις δύο αυτές προσεγγισεις εξαγωγής γνώσης από δεδομένα είναι ότι στις κλασικές βάσεις δεδομένων σώζουμε τα δεδομένα και δημιουργούμε δυναμικά queries ενώ στο CEP έχουμε στατικά queries, τα σύνθετα γεγονότα που θέλουμε να ανιχνεύσουμε και τα δεδομένα αλλάζουν δυναμικά μέσω των ροών(streams) που αναλύονται.

Η κύριες λειτουργίες που παρέχει η EPL**[92,97]** είναι οι εξής.

Filter event Εδώ κρατάμε το event αν ικανοποιείται ένα ή παραπάνω κριτήρια.

Compute Percentages and Ratios Εδώ υπολογίζουμε τιμές και τις ενημερώνουμε όσο έρχονται νέα δεδομένα ή για κάποιο προκαθορισμένο διάστημα.

Correlate events Αποτελεί την βασική λειτουργικότητα ενός CEP Engine και προσπαθεί να συνδέσει δεδομένα από πολλά streams ώστε να ανιχνευσει complex events.

Regular expression Βασικό συστατικό της γλώσσας είναι να υποστηρίζει την εφαρμογή πολύπλοκων κανονικών εκφράσεων πάνω στα data από τα input streams.

Αντί για τον όρο EPL πολλές φορές στην βιβλιογραφία αναφέρεται ως Event Query Language(EQL)**[97]**. Αυτοί οι όροι είναι ταυτόσημοι. Υπάρχουν πολλές γλώσσες που μπορούν να χρησημοποιηθούν**[92,97]** και συνήθως η κάθε υλοποίηση CEP έχει την δική της. Γενικά πάντως οι γλώσσες αυτές μπορούν να κατηγοριοποιηθούν με διάφορα κριτήρια σε είδη. Σε αυτήν την ενότητα δεν θα παρουσιάσουμε μία συγκεκριμένη γλώσσα. Επιλέγουμε να παρουσιάσουμε ένα taxonomy**[97]** το οποίο κατηγοριοποιεί τις γλώσσες με βάση το στυλ της γλώσσας και συγκρίνει τις κατηγορίες με βάση κριτήρια όπως expressivity, ease of use, readability, formal semantics και άλλα. Οι κατηγορίες που προκύπτουν είναι πέντε. Αυτές είναι composition operators, data stream query languages, production rules, timed state machines και logic languages.

Composition operators

Αυτή η κατηγορία δημιουργεί ερωτήσεις(queries) με βάση simple events και ένα σύνολο τελεστών(operators). Οι πιο συνηθισμένοι τελεστές είναι οι ακόλουθοι. Conjunction, Negation,

sequence. Ο πρώτος σημαίνει ότι όλα τα γεγονότα που αναφέρονται πρέπει να συμβούν. Ο δεύτερος μπορεί να συνδυαστεί με τους άλλους δύο και σημαίνει ότι ένα γεγονός δεν θα πρέπει να συμβεί. Τέλος ο τρίτος σημαίνει ότι τα γεγονότα πρέπει να συμβούν με συγκεκριμένη σειρά.

Data Stream Query Languages

Αυτή η κατηγορία περιλαμβάνει γλώσσες που έχουν φτιαχτεί με βάση και αποτελούν επέκταση γλωσσών που χρησιμοποιούνται σε βάσεις δεδομένων. Για τις προσθήκες που πραγματοποιούνται ώστε να είναι κατάλληλες για σενάρια όπου η επεξεργασία πρέπει να γίνεται σε σχεδόν πραγματικό χρόνο αναφερθήκαμε και προηγουμένως. Η κυριότερη προσθήκη είναι η λειτουργικότητα για χρονική συσχέτιση των γεγονότων.

Production Rules

Αυτή η κατηγορία δεν αποτελεί γλώσσα EQL από μόνη της αλλά αποτελεί ένα βολικό και ευέλικτο τρόπο για την δημιουργία υλοποίησης event queries. Αυτά τα rules είναι στενά συνδεδεμένα με μία γλώσσα προγραμματισμού και προσδιορίζουν τις ενέργειες που θα εκτελεστούν όταν το πρόγραμμα φτάσει σε ένα συγκεκριμένο state. Ο αναγνώστης μπορεί να βρεί περισσότερα για αυτή τη μέθοδο εδώ**[101]**.

Timed State Machine

Αυτή η μέθοδος βασίζεται στα automata. Οι δύο περιπτώσεις που χρησιμοποιούνται είναι τα Deterministic Finite State Automata(DFAs) και τα Non Deterministic Finite State Automata(NFAs). Το σύστημα μπορεί να μοντελοποιηθεί σαν ένας κατευθυνόμενος γράφος. Οι κόμβοι αυτού του γράφου είναι οι καταστάσεις που μπορεί να βρεθεί το σύστημα και οι ακμές αντιπροσωπεύουν τις ενέργειες που μπορούν να οδηγήσουν στην μετάβαση σε άλλη κατάσταση ή την παραμονή στην τρέχουσα. Οι ακμές μπορούν να περιλαμβάνουν και την χρονική διάσταση. Μοντελοποιώντας τα σύνθετα γεγονότα με αυτόν τον τρόπο και όταν οδηγηθεί το σύστημα στην τελική κατάσταση τότε έχει ανιχνευτεί ένα σύνθετο γεγονός. Μία περίπτωση που αξίζει αναφορά είναι αυτή των Time Buchi Automata(TBA)**[102]**. Αυτή η περίπτωση είναι η πρώτη που προσπάθησε να επεκτείνει τα automata με την χρονική διάσταση για την μοντελοποίηση συστημάτων.

Logic Languages

Αυτή η κατηγορία αποτελεί μία πολύ φυσική και προφανή επιλογή για τον συγκεκριμένο σκοπό και βασίζεται στο event calculus στο οποίο βασίζονται γλώσσες σαν την prolog με όλες τις δυνατότητες που προσφέρουν αυτές οι γλώσσες και η σύνταξη τους.

Τέλος κλείνοντας αυτήν την ενότητα πρέπει να αναφέρουμε το γεγονός ότι δεδομένου ότι όλες οι γλώσσες και οι εφαρμογές CEP σχεδόν πάντα λαμβάνουν υπόψη τον χρονικό παράγοντα προκύπτει το πρόβλημα της απόκλισης και του σφαλματος των ρολογιών των συσκευών που παράγουν τα γεγονότα. Δεδομένου ότι τις περισσότερες φορές τα events παράγονται από διαφορετικές συσκευές και καταλήγουν σε ένα σύστημα που τα επεξεργάζεται, ο απόλυτος χρόνος

μιας συσκευής διαφέρει από τις υπόλοιπες. Αυτό δημιουργεί το πρόβλημα της αβεβαιότητας για το ποιο γεγονός προηγείται και με ποια σειρά πραγματοποιήθηκαν. Δεν θα παρουσιάσουμε λύσεις για αυτό το πρόβλημα καθώς είναι πέρα από τους σκοπούς της εργασίας και στο use case που μας απασχολεί θεωρούμε ότι όλα τα processes θα τρέχουν στην ίδια συσκευή.

2.6 Διαδικασία Ανίχνευσης Απειλών και σχετικό λογισμικό

2.6.1 Ανίχνευση Απειλών(Threat Detection)

Παρόλο του ότι σε αυτήν την εργασία θα ασχοληθούμε με την ανίχνευση επιθέσεων σε πραγματικό χρόνο σε μεμονωμένα host μηχανήματα αξίζει να αναφερθούμε στην αντίστοιχη διαδικασία που δεν περιορίζεται σε μία μόνο συσκευή. Η διαδικασία της ανίχνευσης απειλών αφορά όλες τις ενέργειες που πρέπει να γίνουν ώστε να ανιχνευθεί κακόβουλη δραστηριότητα και να αντιμετωπιστεί εγκαίρως**[103]** ώστε να μετριαστούν η συνέπειες και να αποφευχθεί η εξάπλωση της ώστε να περιοριστεί ο αριθμός των συστατικών μερών και των χρηστών που επηρεάζονται από αυτήν. Αυτό προϋποθέτει ότι το σύστημα που είναι υπεύθυνο για αυτήν την διαδικασία έχει την δυνατότητα να παρακολουθεί, να αναλύει και να συσχετίζει γεγονότα από όλο το σύστημα και να φιλτράρει αυτά τα δεδομένα ώστε να παράγει αναφορές και ειδοποιήσεις που συνήθως αποτελούν το δεδομένα για για περεταίρω επεξεργασία και ανάλυση είτε από ανθρώπους(αναλυτές), είτε από λογισμικό. Αυτή η διαδικασία σχεδόν πάντα περιλαμβάνει και ανθρώπους αλλά και λογισμικό για να επιτευχθεί. Η τάση βέβαια είναι να αυτοματοποιουνται όσο το δυνατόν περισσότερο οι διαδικασίες ώστε η ανάλυση που πρέπει να γίνει από ανθρώπους να είναι ελάχιστη, εύκολη και σύντομη. Αυτή η προσέγγιση βέβαια δεν αφορά μόνο τον οικονομικό παράγοντα που είναι σχεδόν πάντα ο κύριος παράγοντας για την αυτοματοποίηση ενεργειών μειώνοντας το κόστος από το ανθρώπινο δυναμικό που χρειάζεται . Στον τομέα της ανίχνευσης επιθέσεων είναι σημαντικό γιατί όσο πιο γρήγορα ανιχνευθεί μία επίθεση τόσο πιο γρήγορα μπορεί να περιοριστεί σε αρχικό στάδιο και στην συνέχεια να αντιμετωπιστεί πλήρως. Επίσης ο όγκος των δεδομένων που παράγονται από συστήματα που παρακολουθούνται για λόγους ασφαλείας είναι τεράστιος και η πλήρης καταγραφή και αποθήκευση καθώς και η ανάλυση τους από ανθρώπους είναι εξαιρετικά δύσκολη έως απαγορευτική. Για αυτούς του λόγους έχουν αναπτυχθεί συστήματα**[88,89]** για την διευκόλυνση των αναλυτών και την αποδοτικότητα της όλης διαδικασίας. Αυτά τα συστήματα όμως παρόλες τις εξελίξεις τους δεν είναι σε θέση να λειτουργήσουν αυτόνομα ακόμα. Η όλη διαδικασία περιλαμβάνει και λογισμικό που πραγματοποιει το μεγαλύτερο μέρος της ανάλυσης αλλά και ανθρώπινο δυναμικό

που είναι σε θέση να κατανοήσει καλυτερα τις περιπτώσεις γεγονότων που προέκυψαν από την προηγούμενη επεξεργασία. Γενικά ένα σύστημα που πραγματοποιεί αυτήν την λειτουργία θα πρέπει να έχει την δυνατότητα να αναγνωρίζει και τις γνωστές ευπάθειες αλλά και τις άγνωστες. Η ανίχνευση άγνωστων ευπαθειών είναι ιδιαίτερα δύσκολη διότι η διαδικασία αυτή βασίζεται σε στατικές υπογραφές επιθέσεων. Επίσης πολλές φορές ακόμα και η ανίχνευση γνωστών ευπαθειών είναι δύσκολη και δεν ανιχνεύεται λόγω της πολυπλοκότητας των συστημάτων.

Υπάρχουν πολλές μέθοδοι που μπορεί να επιτευχθεί η ανίχνευση απειλών. ΟΙ βασικές**[103]** είναι threat intelligence, User and entity analytics behaviour, Intruder traps, threat hunting.

Threat Intelligence

Πρόκειται για μία επαναληπτική διαδικασία παραγωγής μοτίβων επιθέσεων με βάση γνώση που έχει εξαχθεί στο παρελθον. Αυτά τα μοτίβα γνωστά και ως υπογραφές έχουν προκύψει από δεδομένα που συλλέχθηκαν και είναι χαρακτηριστικά της συμπεριφοράς μίας επίθεσης. Με βάση αυτά τα χαρακτηριστικά παράγονται οι υπογραφές. Αυτή η τεχνική είναι ιδιαίτερα αποδοτική για την ανίχνευση γνωστών απειλών αλλά όχι αγνώστων. Η επαναληπτικότητα της διαδικασίας αφορά την ενημέρωση της λίστας υπογραφών όταν ανιχνευτεί μία νέα άγνωστη εως τότε επίθεση ή κάποια παραλαγή υπάρχουσας.

User and Entity Behavior Analytics

Αυτή η μέθοδος έχει σκοπό να μοντελοποιήσει την συμπεριφορά χρηστών του συστήματος με αποτέλεσμα να κατανοήσει τον τρόπο που συνηθως συμπεριφέρονται και να δημιουργήσει μία γραμμή βάσης από την συμπεριφορά. Στην συνέχεια και παρακολουθώντας την δραστηριότητα μπορεί να ανιχνεύσει ασυνήθιστες ενέργειες. Αυτή η ανίχνευση μπορεί να οδηγήσει στο συμπέρασμα ότι τα διαπιστευτήρια(credentials) του συγκεκριμένου χρήστη βρίσκονται στην κατοχή ενός επιτιθέμενου ή ότι ο χρήστης αυτός εκτελεί δραστηριότητα για κακόβουλους λόγους και σαν συνέπεια να ανακληθεί η πρόσβαση του χρήστη στο σύστημα ώστε να αποφευχθούν περαιτέρω επιθέσεις ή να τεθεί υπό επιτήρηση ώστε να συλλεχθούν δεδομένα αποδείξεις στην περίπτωση νομικών συνεπειών. Παραδείγματα ασυνήθιστης συμπεριφοράς μπορούν να είναι logins σε ύποπτο χρόνο ή τόπο ή η αντιγραφή μεγάλου όγκου πληροφορίας από μία βάση δεδομένων. Ο όρος entity αναφέρεται στην επέκταση της τεχνικής αυτής από χρήστες σε συσκευές. Σε αυτήν την περίπτωση δημιουργείται γραμμή βάσης(baseline) συμπεριφοράς για δικτυακές συσκευές, σταθμούς εργασίας, servers, εφαρμογές ή οποιαδήποτε άλλη συσκευή υπάρχει συνδεδεμένη στο δίκτυο. Παράδειγμα τέτοιας αποτελεί η επικοινωνία ενός server με μία βάση δεδομένων που συνήθως δεν επικοινωνεί. Η τεχνική αυτή αποτελεί ένα πολύ ισχυρό εργαλείο.

Intruder Traps(Honeypots)

Αυτή η τεχνική ανήκει στην κατηγορία των deceptive τεχνολογιών. Εδώ δημιουργείται ένα σύστημα όπως ένας server ή μία βάση δεδομένων με χαρακτηριστικά τα οποία είναι ιδιαίτερα

ελκυστικά για έναν επιτιθέμενο όπως για παράδειγμα μία βάση δεδομένων με το όνομα passwords. Το σύστημα που θα δημιουργηθεί πρέπει να μοιάζει όσο το δυνατό περισσότερο με πραγματικό πράγμα που σημαίνει ότι θα πρέπει να υπάρχουν χρήστες και περιεχόμενο. Όταν ο επιτιθέμενος αρχίσει να αλληλεπιδρά με το ψεύτικο σύστημα τότε μπορεί να συμπεραθεί ότι υπάρχει κακόβουλος χρήστης στο σύστημα. Σε τέτοιες περιπτώσεις ακολουθεί έρευνα για το πως συνέβει αυτό καθώς και περιορισμός των δραστηριοτήτων του κακόβουλου χρήστη.

Threat Hunting(Penetration Testing)

Εδώ αντί οι αναλυτές να περιμένουν να πραγματοποιηθεί μία επίθεση που θα οδηγήσει στην ανίχνευση μία απειλής κάνουν ενέργειες για τον εντοπισμό τέτοιων απειλών. Πρόκειται για γνωστή τεχνική και αφορά τον τακτικό και ενδελεχή έλεγχο κάθε συστήματος με τις ίδιες τεχνικές που ακολουθούν και οι επιτιθέμενοι ώστε να εντοπιστούν ευπάθειες σε αυτά τα συστήματα. Αυτή η διαδικασία συνήθως εκτελείται από εξειδικευμένους τεχνικούς και παρόλο που δεν έχει την δυσκολία του να μην αποκαλυφθούν που αντιμετωπίζουν οι επιτιθέμενοι, απαιτείται μεγάλη προσοχή στο να μην διαταραχθεί η λειτουργία του συστήματος και να επηρεαστούν οι νόμιμοι χρήστες.

Στην συνέχεια παρουσιάζονται κατηγορίες λογισμικού που είτε σχετίζονται μόνο με την ανίχνευση απειλών ή την περιλαμβάνουν σαν ένα υποσύνολο των συνολικών λειτουργιών που προσφέρουν. Όπως αναφέρθηκε και προηγουμένως, σε αυτήν την ενότητα θα εστιάσουμε μόνο στο στάδιο της ανίχνευσης απειλών παρόλο που στη συνολική διεργασία μίας ολοκληρωμένης διαδικασίας αντιμετώπισης περιλαμβάνει και άλλα βήματα.

2.6.2 Λογισμικό που σχετίζεται με Ανίχνευση Απειλών

- Security Information and Event Management(SIEM)**[88,104]**

Το λογισμικό αυτό συνδυάζει την διαδικασία Security Information Management(SIM) και τη διαδικασία Security Event Management(SEM). Πρόκειται για λογισμικό το οποίο χρησιμοποιείται για πολλούς λόγους πέρα από την διαδικασία της ανίχνευσης απειλών που μας ενδιαφέρει σε αυτήν την ενότητα. Το λογισμικό αυτό είναι κεντρικοποιημένο και συσσωρεύει πληροφορίες από κάθε είδους συσκευή ή εφαρμογή ολόκληρου του δικτύου. Παραδείγματα τέτοιων συσκευών και εφαρμογών αποτελούν routers, σταθμοί εργασίας, firewalls, λογισμικό antivirus, IDSs, web applications και άλλα. Με αυτή την αρχιτεκτονική η ομάδα που ασχολείται με την ασφάλεια έχει συγκεντρωμένη όλη την πληροφορία σε ένα σύστημα και μπορεί να αναλύσει γεγονότα σε όλο το δίκτυο και πληροφοριακό σύστημα σε πραγματικό χρόνο. Με τα χρόνια τα συστήματα SIEM εξελίχθηκαν σε ένα εργαλείο που προσφέρει πολλά περισσότερα από απλή εφαρμογή διαχείρισης καταγραφών(logs). Οι δύο κυριότερες χρήσεις που έχουν αυτά τα συστήματα είναι η διαχείριση απειλών που βρίσκονται σε εξέλιξη και η χρήση τους για την διευκόλυνση διαδικασιών και ελέγχου

που σχετίζονται με συμμόρφωση σε κανονισμούς που μπορεί να υπόκειται ένας οργανισμός. Γενικά τα συστήματα αυτά προσφέρουν την δυνατότητα να συλλέγουν, να συσσωρεύουν, να αποθηκεύουν και να συσχετίζουν γεγονότα από διαφορετικές πηγές με πολύπλοκο τρόπο που θα ήταν πολύ δύσκολο να γίνει χειροκίνητα από μία ομάδα ανθρώπων. Παρόλο που το κάθε λογισμικό τέτοιου τύπου προσφέρει πιθανόν διαφορετική λειτουργικότητα και χρησιμοποιεί άλλους όρους για να την περιγράψει γενικά όλες οι υλοποιήσεις προσφέρουν τουλάχιστον τα παρακάτω.

Διαχείριση Καταγραφών(Log Management)

Αυτό αποτελεί την βασική λειτουργικότητα όπως αναφέραμε και προηγουμένως. Το πλεονέκτημα είναι ότι όλη η πληροφορία βρίσκεται στην διάθεση της ομάδας ασφάλειας και της δίνει το πλεονέκτημα της αυτόματης διαχείρισης όλης αυτής της πληροφορίας.

Συσχέτιση Γεγονότων(Event Correlation)

Αυτή η λειτουργικότητα γενικά στα συστήματα αυτά βασίζεται συνήθως σε στατικές υπογραφές πολύπλοκων γεγονότων που αποτελούν ένδειξη μίας επίθεσης που λαμβάνει χώρα. Η έγκαιρη ανίχνευση τέτοιων σύνθετων γεγονότων δίνει το πλεονέκτημα στην ομάδα ασφάλειας να λάβει άμεσα ενέργειες για την αποτροπή της επίθεσης ή τον μετριασμό των επιπτώσεων της. Η χρήση SIEMs βελτιώνει δύο πολύ σημαντικά μετρικά. Το πρώτο είναι το Mean Time To Detect(MTTD) το οποίο είναι ο χρόνος από την εκκίνηση μίας επίθεσης μέχρι και την ανίχνευση της. Είναι πολύ σημαντικό μετρικό δεδομένου ότι στο παρελθόν υπήρξαν περιπτώσεις όπου η επίθεση έγινε αντιληπτή μήνες μετά την πραγματοποίηση της. Το δεύτερο είναι το Mean Time To Respond το οποίο είναι ο χρόνος από την ανίχνευση μέχρι και την αντιμετώπιση της απειλής. Αυτό βέβαια εξαρτάται και από άλλους παράγοντες όπως το αν η ομάδα υπεύθυνη για την ασφάλεια του οργανισμού έχει έτοιμα αντίμετρα για το ενδεχόμενο μίας επίθεσης. Η νέες γενιές SIEMs προσφέρουν λειτουργικότητα και προς αυτή την κατεύθυνση με αυτοματοποιημένους τρόπους επιλογής και επιβολής αντιμέτρων για την επίθεση όπως για παράδειγμα να απομονώσουν μία συσκευή του συστήματος ή να απενεργοποιήσουν έναν λογαριασμό χρήστη. Αυτό προσφέρει πλεονεκτήματα έναντι των συνηθισμένων τρόπων καθώς αυτοί επιλέγουν αντίμετρα χωρίς να έχει προηγηθεί ανάλυση επιπτώσεων της επίθεσης αλλά και των αντιμέτρων συνολικά στο σύστημα.

Διαχείριση Συμμόρφωσης(Compliance Management)

Πολλοί οργανισμοί είναι αντικείμενα κανονιστικών πλαισίων σχετικά με τα δεδομένα και έχουν υποχρέωση να επιβεβαιώνουν και να επιδεικνύουν την συμμόρφωσή τους με διάφορους νόμους και κανονισμούς. Επειδή τα SIEMs παρέχουν μία συνολική εικόνα όλου του δικτυου και του πληροφοριακού συστήματος του οργανισμού η διαδικασία των εσωτερικών ελέγχων που πραγματοποιεί ο ίδιος ο οργανισμός αλλά και των ελέγχων από τις αρμόδιες ρυθμιστικές αρχές είναι πιο εύκολη και λιγότερο χρονοβόρα. Οι νέες γενιές SIEMs είναι σε θέση να παράγουν αυτόματα αναφορές σχετικές για παράδειγμα με PCI-DSS ή GDPR και να αναγνωρίζουν παραβιάσεις της

συμμόρφωσης σε σχεδόν πραγματικό χρόνο. Όλη αυτή η διαδικασία μειώνει την ανάγκη για ανθρώπινο δυναμικό και την ανάγκη τακτικών ελέγχων από εξωτερικούς ελεγκτές(auditors). Γενικά τα οφέλη των SIEMs μπορούν να συνοψιστούν στα παρακάτω. Ανίχνευση απειλών σε πραγματικό χρόνο, αυτόματη συμμόρφωση με κανονισμούς, βελτίωση της αποδοτικότητας του οργανισμού.

- Endpoint Detection and Response(EDR)**[105,106]**

Το λογισμικό αυτό όπως και τα SIEMs παρέχει περισσότερη λειτουργικότητα από την απλή ανίχνευση απειλών. Το λογισμικό αυτό όπως λέει και το όνομα του εκτελείται σε endpoints. Αυτά μπορεί να είναι σταθμοί εργασίας, servers, routers, IoT συσκευές και άλλα. Μοιάζει πολύ με την περίπτωση που ασχολείται αυτή η εργασία αλλα γενικά τα EDRs προσφέρουν παραπάνω λειτουργικότητα από την ανίχνευση επιθέσεων. Αυτή η λειτουργικοτητα είναι περιορισμός των επιθέσεων στο endpoint, ανάλυση των επιθέσεων και καθοδήγηση για μέτρα αποκατάστασης. Η πρώτη λειτουργικότητα αφορά μέτρα που θα ληφθούν με αυτόματο τρόπο ώστε η επίθεση που πραγματοποιείται στο συγκεκριμένο endpoint να περιοριστεί εκεί. Η δεύτερη λειτουργικότητα αφορά την άμεση συλλογή και αποθήκευση ή μετάδοση δεδομένων για την διερεύνηση της επίθεσης ώστε η ομάδα που είναι υπεύθυνη για την ασφάλεια να έχει ψηφιακά στοιχεία ώστε να παρουσιάσει σε άλλα εμπλεκόμενα μέρη τον τρόπο που πραγματοποιηθηκε αυτή η ενέργεια. Η τελευταία λειτουργικότητα αφορά την πρόταση τρόπων από το λογισμικό, αφού έχει πραγματοποιηθεί ανάλυση επιπτώσεων ώστε το endpoint που δέχθηκε την επίθεση να επανέλθει σε λειτουργία. Αυτό θα μπορούσε να σημαίνει επαναφορά συστήματος ή αποκατάσταση κάποιας βάσης δεδομένων που καταστράφηκε.

- Network Detection and Response(NDR)**[105,106]**

Πρόκειται για την ίδια λειτουργικότητα με το προηγούμενο αλλά με την διαφορά ότι εδώ τα δεδομένα με τα οποία τροφοδοτείται το λογισμικό δεν είναι από ένα μεμονωμένο σύστημα αλλά από διάφορα συστατικά μέρη ενός δικτύου. Εδώ η διαδικασία αφορά την ανίχνευση του τι και ποιός κινείται στο δίκτυο που παρακολουθείται. Με αυτοματοποιημένο και κεντρικοποιημένο τρόπο γίνεται η ανάλυση των γεγονότων και η αντιμετώπιση των απειλών που ανιχνεύονται. Αυτό επιτυγχάνεται με μεθόδους analytics και behavioral τεχνικές όπως αυτές που αναφέρθηκαν προηγουμένως.

- Extended Detection and Response(XDR)

Πρόκειται για σχετικά καινούργια τεχνολογία και όρο[**89,105,106**]. Πολλές φορές ορίζεται ώς ο συνδυασμός EDR και NDR. Σε αυτή την μέθοδο παρακολουθείται κάθε component του δικτύου και του πληροφοριακού συστήματος. Αυτό περιλαμβάνει κάθε συσκευή καθώς και συστατικά μέρη που εκτελούνται στο cloud.Αυτό μπορεί να ερμηνευτεί σαν να υπάρχουν πολλά EDRs ένα για κάθε endpoint σε συνδυασμό με την λειτουργικότητα NDR. Αυτό περιλαμβάνει και δυνατότητες user and entity behavioral analytics. Οι λειτουργίες που παρέχονται είναι η ανίχνευση, ανάλυση όπως και στις

προηγούμενες περιπτώσεις αλλά εδώ προστίθενται άλλες δύο. Η πρώτη είναι η threat hunting που σημαίνει ότι με ενεργό τρόπο προσπαθεί να εντοπισει απειλές. Η δεύτερη είναι η διαδικασία της διόρθωσης(patch). Η βασική διαφορά είναι το ότι συγκεντρώνει και συσχετίζει γεγονότα από όλο το δίκτυο και γεγονότα που υπο κανονικές συνθήκες δεν θα είχαν αντιμετωπιστεί από ένα EDR πλέον μπορούν να ανιχνευθούν. Ένα τέτοιο απλό παράδειγμα αποτελεί ένα κατανεμημένο probe attack. Σε αυτήν την περίπτωση εντοπίζεται παρόλο που τα υπο γεγονότα προέρχονται από διαφορετικά endpoints. Αυτό αποτελεί μεγάλο πλεονέκτημα για τις περιπτώσεις Advanced Persistent Attacks όπως περιγράφηκαν σε προηγούμενη ενότητα. Επίσης σε πολλά προϊόντα XDR χρησιμοποιούνται τεχνικές μηχανικής μάθησης για την αυτόματη βελτίωση των μεθόδων που χρησιμοποιούνται

Με βάση αυτά βλεπουμε ότι η περίπτωση του XDR μοιάζει πολύ με την περίπτωση των SIEMs αλλά με λειτουργικότητα που περιορίζεται στην ανίχνευση και αντιμετώπιση των απειλών. Γενικά η τάση είναι να αυτοματοποιουνται οι διαδικασίες και η χρήση μηχανικής μάθησης έχει βοηθήσει πολύ σε αυτό όπως θα δούμε στην συνέχεια αυξάνοντας το detection rate και μειώνοντας τα false positives . Αυτό ναι μεν είναι θεμιτό αλλά ο ρόλος της ομάδας ασφάλειας και των αναλυτών σε έναν οργανισμό παραμένει σημαντικός. Ειδικά για τις εφαρμογές όπου λαμβάνουν μέτρα για την αντιμετώπιση της απειλής πρέπει να δίνεται μεγάλη προσοχή όταν αυτοματοποιουνται καθώς μία καλά οργανωμένη επίθεση μπορεί να καταφέρει να πετύχει μία πολύ σύνθετη επίθεση Denial of Service σε έναν οργανισμό χρησιμοποιώντας τους ίδιους τους μηχανισμούς της άμυνας του.

Όλες οι μέθοδοι που παρουσιάσαμε χρησιμοποιούνται σε μεγάλης κλίμακας εφαρμογές και αποτελούν πλέον την νόρμα. Ειδικά με την περίπτωση του Security as a Service(SECaaS) πλέον οι οργανισμοί μπορούν να αναθέσουν(outsource) όλη την διαδικασία του security σε εταιρείες που ειδικεύονται σε αυτό και έχουν τις υποδομές και το ανθρώπινο δυναμικό και την τεχνογνωσία να αναλάβουν την εφαρμογή και διαχείριση αυτής της διαδικασίας. Αυτό περιλαμβάνει λύσεις για authentication, antivirus, anti malware, intrusion detection, penetration testing και φυσικά security event management. Αυτό είναι καλό και θεμιτό καθώς προσφέρει ένα υψηλές προδιαγραφές και ποιότητα υπηρεσίας σε όλους τους πελάτες. Επίσης οι πάροχοι τέτοιων υπηρεσιών τις περισσότερες φορές είναι πιστοποιημένοι πράγμα που διασφαλίζει γενικά την ποιότητα. Αυτό επίσης μπορεί να αποτελεί αδυναμία σε κάποιες περιπτώσεις καθώς οι ιδιαιτερότητες που υπάρχουν σε κάθε οργανισμό μπορεί να μην ληφθούν υπόψη όταν σχεδιάζεται η υπηρεσία. Επίσης υπάρχουν και μικροί οργανισμοί για τους οποίους ίσως να είναι πιο εύκολο και αποδοτικό να υλοποιήσουν τις δικές τους μεθόδους και να διατηρούν την δική τους ομάδα ή άτομο υπεύθυνο για την ασφάλεια. Τέλος υπάρχουν και οι περιπτώσεις μεμονωμένων χρηστών που για τους δικούς τους λόγους μπορεί να χρειάζονται τεχνικές λύσεις ασφάλειας. Σε αυτές τις περιπτώσεις παραδοσιακές και δοκιμασμένες λύσεις μπορούν να εφαρμοστούν με μικρό κόστος όπως έχει γίνει με εφαρμογές antivirus οι οποίες πλέον αποτελούν νόρμα και απαραίτητο συστατικό μέρος για κάθε οικιακό χρήστη. Άλλη μια τέτοια

λύση η οποία είναι και ο σκοπός της εργασίας είναι τα Intrusion Detection Systems. Αυτό το λογισμικό σε αντίθεση με αυτά που παρουσιάστηκαν σε αυτήν την ενότητα σκοπό έχει μόνο την ανίχνευση απειλών και την έγκαιρη ενημέρωση για αυτήν και μπορεί να αποτελεί ικανοποιητική λύση για μικρής και μεσαίας κλίμακας υποδομές. Δεν παρουσιάσαμε αυτή την κατηγορία διότι η υλοποίηση της αποτελεί σκοπό της εργασίας και θα την παρουσιάσουμε αναλυτικά στην επόμενη ενότητα.

2.7 Συστήματα Ανίχνευσης Εισβολής(IDS)

Όπως αναφέραμε σύντομα και σε προηγούμενες ενότητες τα IDS**[107]** αποτελούν μία πολύ διαδεδομένη λύση για την ανίχνευση απειλών σε πραγματικό χρόνο. Επίσης αναφέραμε ότι αυτή η τεχνολογία αποτελεί υποπερίπτωση της προηγούμενης ενότητας. Η διαφορά είναι ότι η δράση της είναι στοχευμένη κυρίως στην ανίχνευση ενώ οι προηγούμενες λύσεις περιλαμβάνουν γενικά περισσότερη λειτουργικότητα. Οι προηγούμενες τεχνολογίες στοχεύουν σε κεντρικοποιημένες λύσεις που αναλύουν όλο το πληροφοριακό σύστημα και το δίκτυο και συσχετίζουν γεγονότα από όλα τα endpoints του συστήματος και πολλές φορές η συλλογή πληροφοριών, γεγονότων ή ειδοποιήσεων περιλαμβάνει και δεδομένα που προέρχονται από IDSs. Με βάση αυτά μπορούμε να πούμε ότι οι state of the art λύσεις για πολύπλοκα και μεγάλης κλίμακας πληροφοριακά συστήματα ενσωματώνουν τα IDSs . Σε αντίθεση με αυτήν την αρχιτεκτονική τα IDSs στοχεύουν στο να παρακολουθούν μεμονωμένα σημεία του δικτύου ή συγκεκριμένα work stations ή servers.

Γενικά τα IDSs είναι μία συσκευή ή ένα λογισμικό το οποίο παρακολουθεί όλα τα resources και τα γεγονότα που προκύπτουν σε ένα σύστημα με σκοπό να τα αναλύσει και να κατηγοριοποιήσει την δραστηριότητα ώς νόμιμη ή κακόβουλη. Στην συνέχεια αν μιλάμε για IDSs συνήθως αν ανιχνευθεί μία κακόβουλη ενέργεια ακολουθεί μία ειδοποίηση προς ένα administrator ή προωθεί την ειδοποίηση και τα δεδομένα από τα οποία προέκυψε σε ένα άλλο σύστημα όπως αυτά που παρουσιάσαμε στην προηγούμενη ενότητα**[105,106]**. Άν μιλάμε για IPS τότε επιπλέον πραγματοποιούνται ενέργειες για την αντιμετώπιση του συμβάντος. Παραδείγματα τέτοιων ενεργειών μπορεί να είναι η απενεργοποίηση ενός λογαριασμού χρήστη, η προσθήκη σε black list μίας IP ή ακόμα και ο τερματισμός και η επανεκκίνηση μίας υπηρεσίας. Το τι ακριβώς θα παρακολουθεί ένα τέτοιο σύστημα είναι σχεδιαστική επιλογή. Γενικά η πιο συνηθισμένη κατηγοριοποίηση των IDSs είναι αυτή των host based(HIDS) και network based(NIDS). Για την περίπτωση του host based αυτό τοποθετείται σε κάθε endpoint και παρακολουθεί μέρη του συστήματος όπως για παράδειγμα το σύστημα αρχείων, τα logs του λειτουργικού συστήματος και

την δραστηριότητα κάθε χρήστη. Στην περίπτωση του NIDS αυτό τοποθετείται κοντά στο μέρος του δικτύου το οποίο παρακολουθεί όπως για παράδειγμα σε ένα switch το οποίο βρίσκεται σε ένα τμήμα ενός μεγάλου δικτύου. Και οι δύο προσεγγίσεις έχουν τα οφέλη και τα προβλήματα τους. Ένα NIDS για παράδειγμα δεν έχει πρόσβαση σε όλη την πληροφορία. Ένα δεύτερο χαρακτηριστικό το οποίο χρησιμοποιείται συχνα για την κατηγοριοποίηση είναι αυτό της μεθόδου ο τρόπος με τον οποίο πραγματοποιείται η ανίχνευση. Οι δύο κατηγορίες που αναφέρονται είναι η ανίχνευση με βάση threat intelligence όπως ορίστηκε προηγουμένως. Τα IDSs έχουν εξελιχθεί**[108]** όπως και οι λύσεις που παρουσιάσαμε προηγουμένως. Αλλά πριν από την χρήση μεθόδων μηχανικής μάθησης**[108]** για τον σκοπό αυτό τα συστήματα αυτά βασίζονταν στην χρήση στατικών υπογραφών(signature based detection) για την ανίχνευση μιας απειλής. Θα αναφέρουμε περισσότερα**[109,110]** για τα πλεονεκτήματα και τα μειονεκτήματα της χρήσης μεθόδων μηχανικής μάθησης για αυτό το σκοπό έναντι των παραδοσιακών μεθόδων σε ξεχωριστή ενότητα. Η δεύτερη κατηγορία είναι η anomaly detection. Σε αυτή την περίπτωση η οποία αναφέρεται ως anomaly based detection το σύστημα δημιουργεί μία γραμμή βάσης συμπεριφοράς ενός χρήστη ενός συστήματος ή μίας συσκευής και παράγει ειδοποιήσεις όταν παρατηρεί δραστηριότητα που αποκλίνει από το συνηθισμένο. Οι μέθοδοι μηχανικής μάθησης βρίσκουν εφαρμογή και στις δύο περιπτώσεις όπως θα δούμε στην συνέχεια. Η κατηγοριοποίηση που αναφέραμε είναι η πιο γνωστή και γενικά περιγράφει με σωστό τρόπο τις περιπτώσεις και κατηγορίες των IDSs. Παρόλα αυτά εδώ θα παρουσιάσουμε μία πιο αναλυτική ταξινομία**[111]** η οποία περιγράφει καλύτερα τον τρόπο που λειτουργεί η κάθε προσέγγιση και λαμβάνει υπόψη τον παράγοντα χρήσης μηχανικής μάθησης. Πριν προχωρήσουμε στην παρουσίαση της ταξινομίας θα παρουσιάσουμε τις απαιτήσεις που πρέπει να ικανοποιεί ένα σύγχρονο IDS.

2.7.1 Απαιτήσεις

Ένα σύγχρονο σύστημα IDS θα πρέπει να ικανοποιεί συγκεκριμένες απαιτήσεις**[107,111]** ώστε να είναι η απόδοση τους σε αποδεκτά πλαίσια. Οι απαιτήσεις αυτές με βάση αυτό**[111]** είναι οι εξής.

- Αξιοπιστία(Reliability)

Αυτό αφορά την αξιοπιστία του συστήματος. Το σύστημα θα πρέπει να λειτουργεί συνεχώς σε ένα host χωρίς να χρειάζεται κάποια ανθρώπινη επίβλεψη.

- Διαφάνεια(Transparency)

Αυτό αφορά την ιδιότητα που πρέπει να έχει το σύστημα να μην είναι black box. Αυτό σημαίνει ότι όλες οι λειτουργίες πρέπει να είναι γνωστές και κατανοητές από τα εμπλεκόμενα μέρη και να μην υπάρχει κάποιο μη εξηγήσιμο σημείο. Αυτή η απαίτηση είναι σημαντική καθώς πρέπει να γνωρίζουν τους λόγους που προέκυψε ένα γεγονός. Η περίπτωση των μεθόδων Βαθιάς Μηχανικής

Μάθησης είναι ένα παράδειγμα όπου παραβιάζεται αυτό το προαπαιτούμενο. Αυτό το πρόβλημα στην βιβλιογραφία της μηχανικής μάθησης αναφέρεται ως explainability.

- Ανοχή σε Σφάλματα(Fault Tolerance)

Αυτό σημαίνει ότι το σύστημα πρέπει να έχει την δυνατότητα να ανακάμπτει χωρίς ανθρώπινη παρέμβαση σε περίπτωση κάποιου σφάλματος. Σαν παράδειγμα αναφέρουμε την πιθανότητα βλάβης σε κάποιο virtual machine και την άμεση εκκίνηση ενός νέου ώστε να μην διακοπεί η υπηρεσία.

- Αντοχή σε Καταστροφή(Destruction Resistance)

Το σύστημα πρέπει να έχει την δυνατότητα να παρακολουθεί τον εαυτό του ώστε να ξέρει ότι η λειτουργία του συνεχίζεται απρόσκοπτα και δεν έχει παρακαμφθεί.

- Καταγραφή Γεγονότων(Incident Recording)

Το σύστημα πρέπει να μπορεί να παρατηρεί και να καταγράφει γεγονότα ενδιαφέροντος.

- Προσαρμοστικότητα(Adaptability)

Ο κάθε κόμβος που παρακολουθείται για να προστατευτεί έχει ιδιαιτερότητες. Το IDS πρέπει να μπορεί να προσαρμόζεται στις ανάγκες του κάθε συστήματος.

- Επεκτασιμότητα(Scalability)

Το σύστημα πρέπει να μπορεί να επεκτείνεται και να καλύπτει τις ανάγκες συστημάτων πάνω στα οποία λειτουργίες αφαιρουνται και προστίθενται συνεχώς.

- Επίδοση(Performance)

Το σύστημα πρέπει να έχει την δυνατότητα να ανταπεξέρχεται στις σχεδιαστικές απαιτήσεις. Αυτή η ικανότητα έχει δύο κύριες διαστάσεις. Η πρώτη είναι το throughput δηλαδή το σύστημα πρέπει να είναι σε θέση να μπορεί να καλύψει τον φόρτο εργασίας. Η δεύτερη διάσταση είναι να δίνει μικρό ποσοστό false positives και false negatives.

2.7.2 Ταξινομία

Στην αρχή της ενότητας αναφέραμε μία πολύ βασική και γνωστή προσέγγιση για την κατηγοριοποίηση των IDSs. Σε αυτή την υποενότητα περιγράφουμε μία άλλη μεθοδολογία**[111]**. Εκτός από την μεθοδολογία που θα παρουσιάσουμε υπάρχουν και άλλες οι οποίες θεωρούμε ότι δεν είναι αρκετά επεξηγηματικές δεδομένου του ότι κατηγοριοποιούν τα IDSs με βάση τον αλγόριθμο**[112]** που χρησιμοποιούν κατα την διάρκεια της εκπαίδευσης. Η μεθοδολογία που παρουσιάζουμε επιχειρεί να κατηγοριοποιήσει τα IDSs με βάση την πηγή των δεδομένων που χρησιμοποιείται καθώς και τον τρόπο που πραγματοποιείται η διαδικασία της ανίχνευσης. Επειδή αυτή η πηγή**[111]** δεν παρουσιάζει αναλυτικά την κατηγοριοποίηση με βάση την μέθοδο ανίχνευσης θα παρουσιάσουμε την ταξινομία για την περίπτωση αυτή που αναφέρεται σε αυτό το survey**[113]**.

Επίσης σε άλλες εργασίες παρουσιάζεται και η μέθοδος κατηγοριοποίησης των IDSs με βάση τον τρόπο που αντιδρούν στην ανίχνευση ενός γεγονότος. Ο διαχωρισμός αφορα τις κατηγορίες passive/(re)active. Η passive κατηγορία**[114]** σημαίνει ότι το IDS απλά παρακολουθεί το σύστημα και όταν εντοπίσει ένα γεγονός ενδιαφέροντος απλά ενημερώνει έναν administrator ή προωθεί την πληροφορία σε ένα άλλο λογισμικό. Στην (re)active περίπτωση**[114]** το IDS πραγματοποιεί άμεσα μία ενέργεια για να σταματήσει την επίθεση όπως πχ η απενεργοποίηση ενός λογαριασμού χρήστη. Αυτή η περίπτωση αποτελεί υπο κατηγορία των IDSs και αναφέρεται στην βιβλιογραφία ώς Intrusion Prevention System(IPS).

2.7.2.1 Κατηγοριοποίηση με βάση την μέθοδο Ανίχνευσης

Σε αυτή την μέθοδο με βάση αυτό**[113]** έχουμε τις δύο κύριες κατηγορίες οι οποίες είναι οι misuse detection και anomaly detection. Αυτές οι δύο κατηγορίες στην συνέχεια χωρίζονται σε υπο κατηγορίες.

- Ανίχνευση λάθος Χρήσης(Misuse Detection)

Σε αυτήν την κατηγορία σκοπός είναι να συγκριθεί η τρέχουσα δραστηριότητα σε ένα σύστημα με την αναμενόμενη δραστηριότητα που έχει παρατηρηθεί στο παρελθόν από επιτιθέμενους. Αυτή η κατηγορία πολλές φορές αναφέρεται και ως signature based detection. Σε αυτή την περίπτωση το σύστημα βασίζεται σε μοτίβα(patterns) από γνωστές επιθέσεις. Αυτά τα μοτίβα είναι γνωστά και ως υπογραφές(signatures). Αυτά διατηρούνται σε μία βάση δεδομένων και όταν ανιχνευτεί κίνηση που ταιριάζει με την υπογραφή ενεργοποιείται μία ειδοποίηση ή μία ενέργεια. Αυτή η μέθοδος έχει το πλεονέκτημα του ότι παρέχει πολύ υψηλό accuracy και χαμηλό false positive rate και επίσης είναι απλή και γρήγορη. Τα μειονεκτήματα είναι ότι αυτές οι υπογραφές πολλές φορές δεν μπορούν να ανιχνεύσουν νέες επιθέσεις. Ακόμα και αν αντιμετωπίσουν μία παραλλαγή μίας επίθεσης για την οποία υπάρχει υπογραφή στη βάση δεδομένων είναι πιθανό αυτή να μην ανιχνευθεί. Επίσης όταν αναγνωρίζεται μία νέα επίθεση ή παραλλαγή μιας προηγούμενης πρέπει ενημερώνεται η βάση δεδομένων. Η ενημέρωση αυτή γίνεται συνήθως από ανθρώπους με τεχνικές γνώσεις και είναι χρονοβόρα διαδικασία με μεγάλο κόστος. Ακόμα και για την περίπτωση που αυτό είναι εφικτό ένας επιτιθέμενος μπορεί να αξιοποιήσει την πολυπλοκότητα των δεδομένων ώστε να εκτελέσει μία επίθεση με προχωρημένες τεχνικές ώστε να αποφύγει την ανίχνευση ακόμα και από τις πιο αναλυτικές υπογραφές. Αυτές οι τεχνικές περιλαμβάνουν αλλά δεν περιορίζονται σε payload encoders και κρυπτογράφηση. Για αυτούς τους λόγους όπως θα δούμε και παρακάτω χρησιμοποιούνται υβριδικές λύσεις που κάνουν χρήση μεθόδων μηχανικής μάθησης. Κατηγορίες τεχνικών detection είναι οι χρήση statistical μοντέλων και μηχανικής μάθησης.

- Ανίχνευση Ανωμαλιών(Anomaly Detection)

Η θεωρία πίσω από αυτή την μέθοδο είναι ότι παρακολουθώντας ένα σύστημα ορίζουμε μία γραμμή βάσης συμπεριφοράς για αυτό και στην συνέχεια παρατηρώντας την δραστηριότητα κατηγοριοποιείται κάθε συμπεριφορά που αποκλίνει από αυτό το baseline σαν κακόβουλη δραστηριότητα. Αυτή η μέθοδος μπορεί να δουλέψει και στις δύο κύριες κατηγορίες των IDSs δηλαδή Network based και Host based. Τα πλεονεκτήματα της μεθόδου αυτής είναι ότι παρέχει πολύ καλή γενίκευση της γνώσης με αποτέλεσμα να παρουσιάζει καλή επίδοση στην ανίχνευση άγνωστων επιθέσεων. Τα μειονεκτήματα της μεθόδου είναι ότι δίνει πολύ υψηλό false alarm rate και δεν παρέχει την δυνατότητα της μετρήσιμης αιτιολόγησης του γιατί θεώρησε ότι μία συμπεριφορά δεν είναι συνηθισμένη. Φυσικά η επιτυχία ή αποτυχία της μεθόδου εξαρτάται από την ποιότητα του προφίλ που δημιουργήθηκε. Παραδείγματα τέτοιων μεθόδων ανίχνευσης είναι μοντέλα μηχανικής μάθησης, statistical methods και knowledge based τα οποία περιγραφουμε στην συνέχεια.

Μοντέλα Μηχανικής Μάθησης

Αυτήν την μέθοδο την έχουμε συναντήσει πολλές φορές έως τώρα και δεν χρειάζεται ιδιαίτερη επεξήγηση. Απλά εκπαιδεύουμε ένα μοντέλο και στην συνέχεια πραγματοποιούμε το inference. Να σημειωθεί ότι εδώ μπορούν να χρησιμοποιηθούν και μέθοδοι supervised learning αλλά και unsupervised.

Στατιστικά Μοντέλα

Σε αυτήν την περίπτωση βασικό ρόλο παίζουν τα στατιστικά tests. Εδώ αξιολογούμε την συμπεριφορά που παρατηρείται και με βάση ένα η περισσότερα στατιστικά test υπολογίζουμε την απόκλιση. Τα πλεονεκτήματα αυτής της μεθόδου είναι ότι δεν βασίζονται σε στατικές υπογραφές και είναι πιο ευέλικτα. Τα αρνητικά είναι ότι δίνει μεγάλο false alarm rate. Παραδείγματα τέτοιων μοντέλων είναι markov method, deviation method, multivariate method, and time series method.

Μοντέλα βασισμένα στην Γνώση(Knowledge Based)

Αυτή η μέθοδος μοιάζει πολύ με το signature based misuse detection. Εδώ πάλι έχουμε ένα μεγάλο database με γνώση που έχουμε αποκτήσει από το παρελθόν και προσπαθούμε συγκρίνοντας την παρατηρήσιμη συμπεριφορά με αυτή την γνώση να εντοπίσουμε αποκλίνουσα συμπεριφορά. Αυτή η μέθοδος κληρονομεί όλα τα αρνητικά από την περίπτωση misuse όπως το ότι το knowledge base πρέπει να ενημερώνεται συχνά. Παραδείγματα τέτοιων μεθόδων ανίχνευσης είναι τα expert systems, petri nets και άλλα.

2.7.2.2 Κατηγοριοποίηση με βάση την πηγή των δεδομένων

Σε αυτήν την μέθοδο**[111]** έχουμε τις δύο μεγάλες κατηγορίες που αναφέραμε και προηγουμένως. Network Intrusion Detection και Host Intrusion Detection System. Αυτές οι

κατηγορίες προκύπτουν με βάση του ότι η πρώτη παρακολουθεί και αναλύει τα δεδομένα από τα πακέτα που μεταφέρονται από το δίκτυο ενώ η δεύτερη παρακολουθεί και αναλύει δεδομένα από logs άρα διαφοροποιούνται με βάση το πρώτο κριτήριο της μεθοδολογίας.

- Network Intrusion Detection

Εδώ υπάρχουν τρεις προσεγγίσεις με βάση του ότι τα δεδομένα που αναλύει το IDS προέρχονται από δεδομένα που βρίσκονται σε κίνηση στο δίκτυο. Αυτές είναι οι packet based, flow based και session based. Την διαδικασία και επεξήγηση της κάθε μεθόδου παρουσιάζουμε στην συνέχεια.

Packet Based.

Σε αυτή την μέθοδο αναλύονται πακέτα.Τα πακέτα είναι η βασική μονάδα πληροφορίας που διέρχεται από το δίκτυο. Σε αυτήν την μέθοδο αναλύονται πακέτα απομονωμένα χωρίς να λαμβάνεται υπόψη κάποια άλλη πληροφορία. Τα πακέτα αποτελούνται από δύο συστατικά. Το πρώτο είναι οι κεφαλίδες(headers) οι οποίες περιέχουν πληροφορία σχετικά με το πρωτόκολλο που χρησιμοποιείται, τις διευθύνσεις των μερών που επικοινωνούν καθώς και άλλη πληροφορία που σχετίζεται συνήθως με το πρωτόκολλο όπως flags. Επίσης περιέχουν πληροφορία σχετικά με την δρομολόγηση του πακέτου όπως το πεδίο TTL. Το φορτίο(payload) αποτελεί την πληροφορία που μεταδίδει το πακέτο. Να σημειωθεί ότι το φορτίο μπορεί να περιέχει εκτός από το application payload και κεφαλίδες από ενθυλακωμένα πρωτόκολλα. Τα πλεονεκτήματα αυτής της μεθόδου είναι ότι τα πακέτα συνοδεύονται από διευθύνσεις IP οπότε είναι εύκολο να εντοπιστεί η προέλευση μίας επίθεσης καθώς και ο χρόνος που πραγματοποιήθηκε. Επίσης το φορτίο μπορεί να δώσει αξιόπιστη πληροφορία για τις ενέργειες του επιτιθέμενου αν το σύστημα επιτρέπει την ανάλυση κρυπτογραφημένων δεδομένων. Σε αυτή την περίπτωση χρησιμοποιείται ένας proxy server ο οποίος λειτουργεί σαν man in the middle και διασφαλίζει ότι υπάρχει απο άκρο σε άκρο κρυπτογράφηση αλλά ταυτόχρονα επιτρέπει την ανάλυση του φορτίου κάθε πρωτοκόλλου από μία συσκευή. Τέλος τα πακέτα μπορούν να αναλυθούν και να πραγματοποιηθεί πρόβλεψη σε πραγματικό χρόνο σε αντίθεση με τις μεθόδους flow based και session based. Με βάση αυτά έχουμε δύο υποκατηγορίες. Αυτές είναι οι packet parsing και payload analysis.

Στην μέθοδο **packet parsing** αναλύονται μόνο οι κεφαλίδες των πρωτοκόλλων. Αυτές οι κεφαλίδες περιέχουν πληροφορία. Όλα τα υποπεδία μίας κεφαλίδας που σχετίζεται με ένα πρωτόκολλο μπορούν να συναθροιστούν και να χρησιμοποιηθούν από έναν αλγόριθμο μηχανικής μάθησης σαν χαρακτηριστικά. Δίνοντας αυτά τα χαρακτηριστικά στον αλγόριθμο μπορούμε να τον εκπαιδεύσουμε να εντοπίζει επιθέσεις. Σε αυτήν την μέθοδο συνήθως χρησιμοποιούνται shallow μοντέλα διότι τα δεδομένα έχουν πολύ συγκεκριμένη δομή. Σε αυτή την μέθοδο μπορούν να αναλυθούν ταυτόχρονα όλες οι κεφαλίδες όλου του σωρού πρωτοκόλλων που μεταδίδεται αλλά όχι το application layer φορτιο.

Στην μέθοδο **payload analysis** δίνεται έμφαση στο application level φορτίο. Αυτή η τεχνική δεν έχει τόσο καλά δομημένα χαρακτηριστικά όπως η προηγούμενη κάτι που οδηγεί στην υιοθέτηση τεχνικών Βαθιάς Μηχανικής Μάθησης για την υλοποίηση αυτής της μεθόδου. Πρέπει να σημειωθεί ότι υπάρχουν περιορισμοί. Στην περίπτωση όπου τα δεδομένα δηλαδή το application layer φορτίο είναι κρυπτογραφημένα αυτή η μέθοδος δεν μπορεί να λειτουργήσει. Ακόμα μία δυσκολία που προκύπτει είναι το ότι σε αυτήν την περίπτωση είναι ότι εξετάζουμε την περίπτωση των Network IDSs. Στην προηγούμενη περίπτωση αυτή η δυσκολία μπορεί να ξεπεραστεί με εναν proxy και την αποδοχή από όλους τους κόμβους ενός πιστοποιητικού ασφαλείας ενω στην περίπτωση της ανάλυσης φορτίου το NIDS πρέπει να έχει στην κατοχή του το κλειδί κρυπτογράφησης καθώς και την γνώση του τύπου της εφαρμογής ανάμεσα σε δύο peers που επικοινωνούν. Αυτοί οι περιορισμοί καθιστούν την μέθοδο αυτή δύσκολη στην υλοποίηση.

Flow Based.

Σε αυτήν την μέθοδο έχουμε την ίδια πληροφορία που έχουμε στην περίπτωση του packet parsing. Η διαφορά εδώ είναι ότι δεν αναλύουμε συγκεκριμένα πακέτα αλλά ένα σύνολο πακέτων που μεταδόθηκαν σε ένα προκαθορισμένο χρονικό διάστημα. Αυτό έχει το αρνητικό ότι τα πακέτα πρέπει να αποθηκεύονται και ο χρόνος πρόβλεψης δεν είναι πραγματικού χρόνου. Ο πρώτος περιορισμός είναι σημαντικός μόνο στην περίπτωση που υπάρχει πολύ κίνηση στο δίκτυο γεγονός που κάνει την προσωρινή αποθήκευση των πακέτων σημείο αποτυχίας(bottleneck) της διαδικασίας. Ο δεύτερος περιορισμός δεν είναι ιδιαίτερα σημαντικός καθώς είναι αρκετά κοντά στο πραγματικό χρόνο συνήθως με καθυστέρηση λίγων δευτερολέπτων. Οι προσεγγίσεις για αυτή την μέθοδο είναι τρείς. Αυτές είναι οι feature engineering, deep models και traffic grouping based detection.

Στην μέθοδο **feature engineering** με manual τρόπο εξάγουμε χαρακτηριστικά από το σύνολο των πακέτων που έχουν αποθηκευτεί προσωρινά. Αυτά τα χαρακτηριστικά είναι τις περισσότερες φορές στατιστικά της δραστηριότητας που παρατηρήθηκε. Παραδείγματα είναι όλα τα χαρακτηριστικά από τα πρωτόκολλα και τις κεφαλίδες, ο αριθμός των bytes που εστάλησαν και άλλα. Στην συνέχεια όλα αυτά χρησιμοποιούνται σαν είσοδος σε ένα shallow μοντέλο. Αυτή η μέθοδος είναι πολύ συνηθισμένη και όπως θα δούμε στο πειραματικό μέρος της εργασίας τα περισσότερα features του dataset αφορούν τέτοια δεδομένα. Αυτή η μέθοδος τις περισσότερες φορές επιτυνχάνει τον περιορισμό του πραγματικού χρόνου.

Deep Models. Παρόλο που η προηγούμενη μέθοδος είναι η πιο συνηθισμένη για την περίπτωση του flow based detection υπάρχουν και λύσεις που χρησιμοποιούν deep learning και η τάση είναι να αντικαταστήσουν την προηγούμενη μέθοδο καθώς στην περίπτωση των shallow μοντέλων η ποιότητα των features αποτελεί σημείο περιορισμού(bottleneck) της διαδικασίας. Εδώ η διαδικασία εξαρτάται από την κάθε μέθοδο αλλά γενικά εκμεταλλευόμαστε την ιδιότητα των deep

μοντέλων να λειτουργούν απο άκρο σε άκρο. Αυτό σημαίνει ότι δίνουμε κατευθείαν τα μη προ επεξεργασμένα δεδομένα στον αλγόριθμο και αυτός μαθαίνει ιεραρχικά χαρακτηριστικά και πραγματοποιεί πρόβλεψη στον ίδιο χρόνο. Αυτό είναι πλεονέκτημα καθώς τα χαρακτηριστικά που μαθαίνονται παρέχουν περισσότερη πληροφορία από αυτά που κατασκευάζονται με χειροκίνητο τρόπο. Για την προηγούμενη περίπτωση αναφερόμαστε στην περίπτωση επιβλεπόμενης μάθησης. Στην βιβλιογραφία αναφέρονται περιπτώσεις όπου χρησιμοποιούνται τεχνικές μη επιβλεπόμενης Βαθιάς Μηχανικής Μάθησης για εξαγωγή χαρακτηριστικών και στην συνέχεια η χρήση shallow μοντέλων για την διαδικασία της κατηγοριοποίησης(classification). Τα μειονεκτήματα των deep τεχνικών είναι ότι δεν δίνουν καλά αποτελέσματα σε μικρά datasets και στις περιπτώσεις datasets με μεγάλη διαφορά μεταξύ των διαθέσιμων δειγμάτων για κάθε κλάση(imbalance).

Traffic Grouping. Επειδή η μέθοδος flow based detection που αναλύουμε περιλαμβάνει όλη την κίνηση μέσα σε ένα χρονικό περιθώριο περιλαμβάνει και πληροφορία που δεν έχει καμία αξία για την διαδικασία δηλαδή αποτελεί θόρυβο. Αυτό σε συνδυασμό με τον περιορισμό σε διαθέσιμα δεδομένα μπορεί να οδηγήσει σε overfitting ένα πρόβλημα που αναφέραμε σε προηγούμενη ενότητα. Για να αντιμετωπιστεί αυτό στη μέθοδο που αναλύουμε εφαρμόστηκαν λύσεις που ομαδοποιούν τα δεδομένα με βάση τα δεδομένα ή το πρωτόκολλο. Στην περίπτωση της ομαδοποίησης με βάση το πρωτόκολλο δημιούργησαν υποσύνολα δεδομένων για κάθε πρωτόκολλο και εκπαίδευσαν ξεχωριστούς κατηγοριοποιητές για κάθε ομάδα.

Session Based Detection

Σε αυτήν την μέθοδο βασικό ρόλο παίζει η έννοια της συνόδου. Η σύνοδος περιγράφεται και εντοπίζεται με βάση τα χαρακτηριστικά των δύο άκρων που επικοινωνούν και του πρωτοκόλλου που χρησιμοποιείται. Η μέθοδος αυτή έχει το πλεονέκτημα του ότι μπορεί να ανιχνεύσει πολλές γνωστές επιθέσεις όπως tunneling και trojan αλλα και το μειονέκτημα ότι μία σύνοδος παραμένει ενεργή για πολύ ώρα με σκοπό να πρέπει να αποθηκευτούν πολλά πακέτα για κάθε σύνοδο κάτι που αυξάνει σημαντικά το overhead της διαδικασίας επίσης πλέον η απαίτηση για πρόβλεψη σε πραγματικό χρόνο δεν μπορεί να επιτευχθεί. Οι μέθοδοι ανίχνευσης με βάση την σύνοδο είναι οι statistic based features και sequence based μέθοδοι.

Στην περίπτωση **statistic based features** χρησιμοποιούνται στατιστικά από όλη την διάρκεια της συνόδου τα οποία όπως και στις προηγούμενες ανάλογες περιπτώσεις αφορούν πεδία από κεφαλίδες, τον αριθμό των πακέτων, την αναλογία εισερχόμενων και εξερχόμενων πακέτων και πολλά άλλα. Αυτά τα χαρακτηριστικά χρησιμοποιούνται στην συνέχεια από shallow μοντέλα με τον ίδιο τρόπο όπως και σε προηγούμενες περιπτώσεις. Η κατηγοριοποίηση εδώ αφορά κανονικές και μή κανονικές συνόδους. Επίσης αυτή η μέθοδος όπως και όλες όσες βασίζονται σε shallow μοντέλα εξαρτώνται από expert knowledge κατα την διαδικασία του της δημιουργίας

χαρακτηριστικών(feature engineering) κάτι που αποτελεί μειονέκτημα όσον αφορά την γενίκευση της γνώσης σε νέα σενάρια δηλαδή νέες επιθέσεις. Το μειονέκτημα αυτής της μεθόδου είναι ότι αγνοεί την ακολουθία των γεγονότων με αποτέλεσμα να γίνεται μη αποδοτικό και αξιόπιστο διότι χάνει πληροφορία. Για αυτό τον λόγο στην δεύτερη μέθοδο γίνεται προσπάθεια να εξαχθεί και να χρησιμοποιηθεί αυτή η πληροφορία.

Στην περίπτωση **sequence based features** και αντίθετα με την περίπτωση του flow based detection τα δεδομένα έχουν μία χρονολογική σειρά που κρύβει πληροφορία που αξίζει να αξιοποιηθεί. Οι περισσότεροι αλγόριθμοι μηχανικής μάθησης αδυνατούν να εκμεταλλευτούν τέτοια δεδομένα. Η προφανής επιλογή για αυτό είναι η χρήση deep μοντέλων τύπου RNN. Παρόλο που είναι ελάχιστες οι χρήσεις τέτοιων μοντέλων για αυτό τον σκοπό υπάρχουν κάποιες αναφορές στην βιβλιογραφία.

- Host Based Intrusion Detection

Σε αυτή την περίπτωση η οποία είναι και το αντικείμενο της εργασίας τα δεδομένα που χρησιμοποιούνται για την ανίχνευση προέρχονται από τις καταγραφές(logs) Αυτό περιλαμβάνει διάφορες ετερογενής πηγές πληροφορίας και το τι ακριβώς είναι διαθέσιμο στα logs εξαρτάται από πολλούς παράγοντες. Αυτοί μπορεί να περιλαμβάνουν γεγονότα που προέρχονται από ξεχωριστές διεργασίες, το λειτουργικό σύστημα, καποιο firewall που τρέχει τοπικά στο host, λογισμικό antivirus και άλλα. Όταν έχουμε ένα μοντέλο μηχανικής μάθησης το οποίο εκπαιδεύτηκε με συγκεκριμένα features αποτελεί πρόκληση το να καταφέρουμε να αντλήσουμε σε πραγματικό χρόνο την απαραίτητη πληροφορία από διάφορες ετερογενής πηγές και να την αξιοποιήσουμε στον κατάλληλο χρόνο ώστε να τροφοδοτούμε το μοντέλο. Στην περίπτωση των λειτουργικών συστημάτων συνήθως έχουμε αρκετά αναλυτική πληροφορία για διάφορα γεγονότα που συμβαίνουν όπως login προσπαθειες και άλλα. Αλλά όταν ο στόχος είναι να παρακολουθούμε ξεχωριστά processes και χρειαζόμαστε συγκεκριμένη πληροφορία από αυτά πολλές φορές δεν την έχουμε.

Ανίχνευση βασισμένη σε καταγραφές(Log Based Detection)

Για τα logs και το τι πληροφορία μπορεί να έχουμε διαθέσιμη αναφερθήκαμε στην εισαγωγή. Επίσης εκτός από αυτά που αναφέραμε στην περίπτωση ενός HIDS έχουμε επίσης και όλη την πληροφορία για όλα τα sessions που αφορούν το host στο οποίο εκτελείται το HIDS. Τελος δεν έχουμε τον περιορισμό του ότι δεν μπορούμε να αξιοποιήσουμε κρυπτογραφημένα δεδομένα καθώς με τη σωστή διαμόρφωση των εφαρμογών μπορούμε να έχουμε πρόσβαση σε αυτή την πληροφορία. Τα πλεονεκτήματα αυτής της μεθόδου είναι ότι στα logs μπορούμε να έχουμε συγκεκριμένη πληροφορία ώστε να ανιχνεύσουμε με ακρίβεια διάφορους τύπους επιθέσεων όπως επιθέσεις injection. Δεύτερον τα γεγονότα που καταγράφονται περιέχουν αναλυτικά στοιχεία για τους λογαριασμούς χρηστών που πιθανώς χρησιμοποιήθηκαν σε μία επίθεση καθώς και χρονο σφραγίδες

κάτι που μπορεί να βοηθήσει στην γρήγορη ανταπόκριση του συστήματος σε μία επίθεση. Τρίτον τα logs μπορούν να καταγράψουν λεπτομερώς την πορεία της επιθεσης κάτι που κάνει τις αποφάσεις του IDS εξηγήσιμες. Το αρνητικό αυτής της μεθόδου είναι ότι τα logs είναι γενικά ετερογενή. Αυτό σημαίνει ότι η μορφή, οι συχνότητες καταγραφής το είδος της πληροφορίας και το μέρος αποθήκευσης διαφέρουν δραματικά από εφαρμογή σε εφαρμογή. Αυτά τα προβλήματα πρέπει να αντιμετωπιστούν για μία λύση. Σε αυτό θα ξανα αναφερθούμε στην ενότητα της μεθοδολογίας που θα ακολουθήσουμε. Γενικά και με βάση την ταξινομία που αποφασίσαμε να ακολουθησουμε στην παρουσίαση οι τεχνικές που βασίζονται στα logs είναι rule και ML based, log feature extraction και text analysis.

Hybrid Rules και ML

Αυτές οι μέθοδοι χρησιμοποιούν έναν συνδυασμό από κανόνες(υπογραφές) και μοντέλα μηχανικής μάθησης. Επειδή από μόνοι τους οι κανόνες τις περισσότερες φορές παράγουν πολλά αποτελέσματα και έχουν πολλά false positives συνδυάζονται με μοντέλα ML ώστε να φιλτράρουν όλα αυτά τα αποτελέσματα.

Log Feature Extraction

Αυτή η μέθοδος όπως και όλες όσες χρησιμοποιούν features που εξάγονται με manual τρόπο προϋποθέτουν domain expert knowledge. Εδώ συνδυάζοντας πληροφορία από πολλές πηγές εξάγονται features τα οποία στην συνέχεια τροφοδοτούν αλγορίθμους ώστε να ανιχνευθεί μη συνηθισμένη συμπεριφορά(anomaly detection). Τις περισσότερες φορές χρησιμοποιείται ένα sliding window το οποίο περιλαμβάνει πληροφορία(contextual information) και με βάση τα γεγονότα που βρίσκονται μέσα σε αυτό το χρονικό διάστημα εξάγονται features τα οποία τροφοδοτούν τα μοντέλα μηχανικής μάθησης. Επίσης το sliding window αποτελεί μία μέθοδο streaming που εξασφαλίζει χαμηλό delay στην διαδικασία.

Text Analysis

Αυτή η μέθοδος αντιμετωπίζει όλα τα logs σαν απλό κείμενο. Αυτές οι μέθοδοι χρησιμοποιούν δοκιμασμένες και συνηθισμένες τεχνικές ανάλυσης κειμένου για να αναλύσουν τα logs. Σε σύγκριση με τη προηγούμενη τεχνική που αφορά Log feature extraction αυτή η μέθοδος ερμηνεύει το περιεχόμενο των logs σε σημασιολογικό επίπεδο και αυτό έχει το πλεονέκτημα του ότι τα αποτελέσματα είναι ερμηνεύσιμα. Όταν αναλύονται κείμενα μερικές λέξεις παίζουν σημαντικό ρόλο ενώ άλλες όχι έτσι ένα λεξικό από έννοιες που χρησιμοποιούνται στον τομέα της ασφάλειας συστημάτων μπορεί να βοηθήσει να βελτιωθεί η απόδοση της ανίχνευσης.

3.Σχετικές Εργασίες

Σε αυτή την ενότητα θα παρουσιάσουμε εργασίες και αποτελέσματα πάνω στο τομέα που μας ενδιαφέρει δηλαδή IDSs βασισμένα σε αλγορίθμους Μηχανικής Μάθησης. Όπως θα δούμε οι λύσεις που υπάρχουν μένουν στην δημιουργία ενός μοντέλου ML ή DL. Γενικά υπάρχουν πολλές τέτοιες δουλειές στην βιβλιογραφία και έχει χρησιμοποιηθεί μεγάλο εύρος τεχνολογιών για την υλοποίηση τους. Σκοπός αυτής της ενότητας είναι να αναδείξει αυτό. Με βάση αυτά θα παρουσιάσουμε λύσεις με βάση την τεχνολογία που χρησιμοποιείται και όχι κάποια ταξινομία σαν αυτά που παρουσιάσαμε στην προηγούμενη ενότητα. Θα χωρίσουμε την ενότητα σε δύο υπο ενότητες. Η πρώτη αφορά λύσεις βασισμένες σε Deep Learning και η δεύτερη σε shallow μοντέλα.

3.1.Βαθιά Μηχανική Μάθηση

3.1.1 A Deep Auto-Encoder based Approach for Intrusion Detection System

Η εργασία αυτή**[115]** χρησιμοποιεί την τεχνολογία Autoencoders και ασχολείται με anomaly based detection. Επίσης για την εκπαίδευση και την αξιολόγηση χρησιμοποιείται το γνωστό KDD99 dataset. Η μεθοδολογία που ακολουθείται είναι η εξης. Χρησιμοποιώντας τον Autoencoder με είσοδο όλα τα χαρακτηριστικά υπάρχουν στο dataset και χρησιμοποιώντας κρυμμένα επίπεδα παράγει αναπαραστάσεις σε χαμηλή διάσταση. Στην συνέχεια συνδυάζει τον autoencoder ένα ακόμα κρυμμένο επίπεδο το οποίο είναι ο classifier και χρησιμοποιείται για επιβλεπόμενη μάθηση με έξοδο το σύνολο των κλάσεων του dataset. Η επίδοση του μοντέλου αξιολογήθηκε σε δύο σενάρια. Το πρώτο είναι το binary classification όπου γίνεται κατηγοριοποίηση επίθεση ή όχι. Στο δεύτερο έχουμε multiclass classification με όλες τις διαθέσιμες κλάσεις. Το accuracy στο πρώτο σενάριο είναι 96% και στο δεύτερο 95%. Το πλεονέκτημα είναι ότι χρειάζεται πολύ μικρή προεπεξεργασία και χρησιμοποιεί όλη την διαθέσιμη πληροφορία.

3.1.2 Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection

Εδώ**[116]** χρησιμοποιείται ο μηχανισμός LSTM. Τα χαρακτηριστικά περνάνε από ένα LSTM δίκτυο το οποίο εκπαιδευεται end to end με έναν classifier. Η μέθοδος δεν περιγράφεται με μεγάλη λεπτομέρεια. Το dataset που χρησιμοποιείται είναι και πάλι το KDD99 για εκπαίδευση και αξιολόγηση. Επίσης πραγματοποιούν πολλά πειράματα ώστε να βρουν τις βέλτιστες υπερ

παραμέτρους. Οι δημιουργοί αξιολογούν το μοντέλο με βάση το Detection Rate και το False Alarm Rate και το accuracy. Για το πρώτο έχουν 98% για το δεύτερο 10% και για το τρίτο 96%.

3.1.3 A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks

Εδώ**[117]** χρησιμοποιούνται και πάλι RNNs. Τα χαρακτηριστικά είναι είσοδος στο δίκτυο και συνδυάζεται με έναν classifier. Το dataset που χρησιμοποιείται είναι και πάλι το KDD99. Οι δημιουργοί πραγματοποιούν δύο πειράματα όπως και στο πρώτο. Ένα για binary classification και ένα για multiclass. Η αξιολόγηση του μοντέλου γίνεται με βάση το detection rate το οποίο είναι 97%.

3.1.4 A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks

Σε αυτή**[118]** την δουλεία χρησιμοποιούνται τα CNNs. Αρχικά πραγματοποιείται μία προ επεξεργασία και κανονικοποίηση στα χαρακτηριστικά. Στην συνέχεια το διάνυσμα αυτό μετατρέπεται σε 2D δομή δηλαδή εικόνα. Στην συνέχεια εκπαιδεύεται το δίκτυο με βάση τις ετικέτες(labels). Οι δημιουργοί αξιολογούν την επίδοση του μοντέλου με βάση το accuracy και το False Alarm Rate. Για το πρώτο έχουν 99% και για το δεύτερο 27%.

3.1.5 Deep Learning Approach for Network Intrusion Detection in Software Defined Networking

Σε αυτή**[119]** την δουλειά οι δημιουργοί χρησιμοποίησαν ένα Deep Neural Network. Το δίκτυο αυτό αποτελείται από τρία κρυμμένα επίπεδα. Το πρόβλημα που λύνουν είναι το binary classification δηλαδή επίθεση ή όχι. Το dataset που χρησιμοποίησαν είναι το KDD99 και εδώ. Οι δημιουργοί εδώ ακολούθησαν την τακτική του να χρησιμοποιήσουν όσο λιγότερα χαρακτηριστικά γίνεται και να είναι εύκολο να τα συλλέξουν. Με βάση αυτό χρησιμοποιούν μόνο έξι που μπορούν να βρεθούν από ένα δίκτυο εύκολα. Το αποτέλεσμα ήταν να έχουν accuracy 76%.

3.1.6 Deep Learning-Based Intrusion Detection System for Advanced Metering Infrastructure

Και εδώ**[120]** οι δημιουργοί χρησιμοποιούν ένα Deep Neural Network. Επίσης και εδώ χρησιμοποιείται το KDD99 dataset. Το πρόβλημα που λύνουν είναι το binary classification δηλαδή επίθεση ή νόμιμη κίνηση. Επίσης πραγματοποιούν πειράματα με τις υπερ παραμέτρους όπως activation functions. Η αξιολόγηση πραγματοποιήθηκε με το μετρικό του accuracy το οποίο ισχυρίζονται ότι είναι 99% με χρήση όλων των χαρακτηριστικών.

3.2 Μηχανική Μάθηση

3.2.1 Building an Efficient Intrusion Detection System Based on Feature Selection and Ensemble Classifier

Σε αυτή**[121]** την εργασία οι δημιουργοί χρησιμοποιούν shallow ensemble μοντέλα. Και εδώ χρησιμοποιείται το dataset KDD99. Όπως συνηθίζεται για τα shallow μοντέλα προηγείται της εκπαίδευσης του μοντέλου η διαδικασία της επιλογής χαρακτηριστικών(feature selection). Εδώ μετά από αυτό το βήμα εφαρμόζονται vote classifiers οι οποίοι αποτελούν μέθοδο ensemble. Η διαδικασία του feature selection πραγματοποιείται με την τεχνική CFS(Correlation based Feature Selection) . Στόχος αυτής της μεθόδου είναι να βρει τα χαρακτηριστικά εκείνα που σχετίζονται περισσότερο με την ετικέτα(label) ή που σχετίζονται λιγότερο με άλλα features άρα φέρουν περισσότερη πληροφορία. Πιο συγκεκριμένα χρησιμοποιείται ένας αλγόριθμος με το όνομα CFS-BA. Στην συνέχεια εκπαιδεύουν τρία ensemble μοντέλα, Αυτά είναι τα C4.5, Random Forest και Forest PA. Η αξιολόγηση του συνδυασμού των τεχνικών για την διαδικασία της επιλογής χαρακτηριστικών και των classifiers έγινε με όλα τα γνωστά metrics όπως Accuracy, F-score, Precision Detection Rate. Με χρήση όλων των χαρακτηριστικών κατάφεραν αποτελέσματα με όλα τα metrics της τάξης του 94%. Με χρήση της μεθόδου επιλογής χαρακτηριστικών και με μόνο 10 features κατάφεραν αποτελέσματα της τάξης του 98%. Βλέπουμε ότι η μέθοδος feature selection που ακολούθησαν παράγει αξιοσημείωτα αποτελέσματα.

3.2.2 Hybrid Model For Intrusion Detection Systems

Σε αυτή**[122]** την εργασία οι δημιουργοί προτείνουν πάλι ένα ensemble μοντέλο το οποίο συνδυάζει Decision Tree και Random Forest με την ensemble μέθοδο stacking. Να αναφέρουμε εδώ ότι η μέθοδος stacking αφορά την παράλληλη εκπαίδευση μοντέλων και τον συνδυασμό τους στην

εκπαίδευση ενός μετα μοντέλου. Αυτό επιτυγχάνεται με την εκπαίδευση πολλών μοντέλων παράλληλα και στην συνέχεια την χρήση των εξόδων αυτών για την εκπαίδευση ενός μεταμοντέλου. Το μοντέλο αυτό μπορεί να είναι και νευρωνικό δίκτυο. Το πλεονέκτημα με την μέθοδο stacking είναι ότι αν ένας από τους classifiers δεν έχει καλή απόδοση για οποιουσδήποτε λόγους είναι σχεδόν σίγουρο ότι ο meta classifier θα το αντιμετωπίσει.Οι έξοδοι των μοντέλων βάσης και συνεπώς η είσοδος του meta classifier αναφέρονται ώς μετά χαρακτηριστικά(meta features). Οι δημιουργοί εδώ συνδυάζουν δύο base models Random Forest και Decision Tree. Χρησιμοποιήθηκε και εδώ το KDD99 dataset και η αξιολόγηση του μοντέλου έγινε με βάση τα True Positive, False Positive, Recall, Precision και Fscore. Τα αποτελέσματα ήταν 0.85, 0.13, 0.86, 0.85, 0.85.

3.2.3 Intrusion Detection System Using Data Mining Technique: Support Vector Machine

Σε αυτή**[123]** την δουλειά οι δημιουργοί χρησιμοποιούν classifier SVC. Και εδώ χρησιμοποιείται το KDD99 dataset. Πρόκειται για απλό μοντέλο το οποίο μετά από όλα τα βήματα προεπεξεργασίας εκπαιδεύει τον SVC. Επίσης δοκιμάζουν υπερ παραμέτρους για το kernel. Η αξιολόγηση της επίδοσης έγινε με βάση το metric accuracy και το καλύτερο αποτέλεσμα που πετυχαν ήταν 98%.

3.2.4 Evaluation of Machine Learning Algorithms for Intrusion Detection System

Εδώ**[124]** οι δημιουργοί χωρίς να προτείνετε κάποιο καινούριο μοντέλο, αρχιτεκτονική ή κάποια διαδικασία feature selection όπως έκαναν κάποιοι από τους προηγούμενους συγκρίνουν την επίδοση των πιο γνωστών shallow μοντέλων καθώς και ενός MLP. Το dataset που χρησιμοποιείται είναι το KDD99. Οι αλγόριθμοι που δοκιμάζονται είναι J.48, Random Forest, Random Tree, Decision Table, Naive Bayes, Bayes Network και MLP. Η αξιολόγηση γίνεται με βάση. Μετά από κάποια βασική και συνηθισμένη προ επεξεργασία δοκίμασαν τους classifiers και τους αξιολόγησαν με βάση το True Positive Rate, Precision, Accuracy και ROC. Δεν διευκρινίζουν αν προχώρησαν σε επιλογή χαρακτηριστικών. Γενικά έχουν καλά αποτελέσματα σε όλα τα metrics και για τους περισσότερους classifiers.

Βλέπουμε με βάση αυτά ότι υπάρχουν πολλές τεχνολογίες που μπορούν να πετύχουν την εκπαίδευση ενός μοντέλου που στην συνέχεια χρησιμοποιούνται για ένα ML IDS. Σαν συμπέρασμα και σαν σύγκριση μεταξύ των deep και shallow μοντέλα μπορούμε να πούμε ότι με το είδος και την ποσότητα των δεδομένων που είναι διαθέσιμα είτε σε δημόσια διαθέσιμα datasets είτε σε μή δημόσια

και την δομή που αυτά έχουν είναι συνήθως καλύτερα να χρησιμοποιούνται shallow μοντέλα όπως αναφέραμε και σε προηγούμενη ενότητα. Επίσης με βάση αυτές τις δουλειές βλέπουμε ότι μια τακτική επιλογής χαρακτηριστικών είναι απαραίτητη καθώς αν και στην διάρκεια της εκπαίδευσης ίσως να μην έχει ιδιαίτερη σημασία το πόσα και τι είδους features έχουμε, όταν η λύση θα βγει σε λειτουργία ίσως να μην υπάρχουν τα μέσα για να έχουμε όλη την πληροφορία διαθέσιμη. Επίσης ο χρόνος πρόβλεψης και γενικά το overhead σε πόρους μπορεί να μην είναι αποδεκτό. Περισσότερα για αυτό θα πούμε σε επόμενη ενότητα.

4.Μεθοδολογία

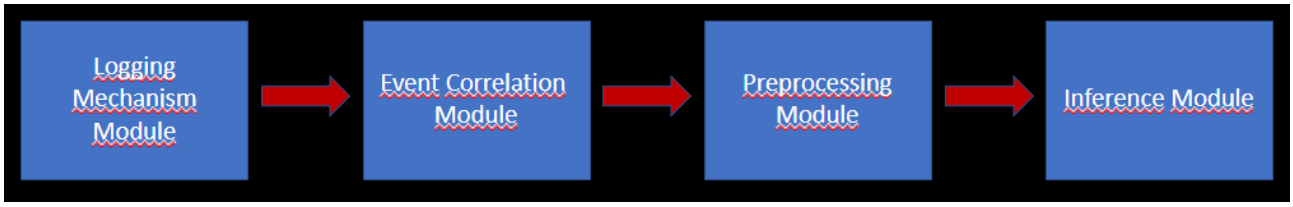
Όπως είδαμε στην προηγούμενη ενότητα υπάρχουν πολλές εργασίες που με διάφορες μεθόδους είτε shallow είτε deep πετυχαίνουν την δημιουργία ενός μοντέλου. Γενικά όλες αυτές οι προσεγγίσεις παράγουν καλά αποτελέσματα στο test dataset. Ειδικά στην περίπτωση πολύ εύκολων σεναρίων όπως η περίπτωση της επίθεσης DoS η απόδοση αγγίζει το 100%. Το πρόβλημα είναι ότι η διαδικασία της δημιουργίας ενός τέτοιου συστήματος δεν τελειώνει στην εκπαίδευση του μοντέλου Μηχανικής Μάθησης. Αυτό σημαίνει ότι πρέπει να εξεταστεί ο συνολικός τρόπος που μπορεί να τρέξει αυτό το σύστημα ώστε να πραγματοποιεί το σκοπό του.

Σκοπός αυτής της εργασίας είναι με συνδυασμό των κατάλληλων τεχνολογιών να παρουσιάσει όλες τις πτυχές και τα προβλήματα που προκύπτουν ώστε να δημιουργηθεί μία λειτουργική λύση και να προτείνει λύσεις στα κυριότερα προβλήματα που προκύπτουν στην πράξη.

Ξεκινώντας από την αρχή τα βήματα που θα πρέπει να πραγματοποιηθούν είναι πέντε. Το πρώτο είναι να βρούμε ποιες διεργασίες θα παρακολουθεί το σύστημα. Το δεύτερο βήμα είναι να βρούμε τρόπο αυτές οι διεργασίες να καταγράφουν την απαραίτητη πληροφορία που χρειάζεται το μοντέλο μηχανικής μάθησης για να λειτουργήσει. Το τρίτο βήμα είναι να βρούμε έναν τρόπο ώστε για κάθε διεργασία που παρακολουθείται να συσχετίζεται πληροφορία όταν αυτή είναι διαθέσιμη από ένα συστατικό μέρος και όταν η απαραίτητη πληροφορία είναι διαθέσιμη να τροφοδοτεί το μοντέλο για να γίνει η πρόβλεψη. Το τέταρτο βήμα είναι η δημιουργία του μοντέλου με τον πιο αποδοτικό τρόπο με βάση τις απαιτήσεις. Τέλος το πέμπτο βήμα είναι ο τρόπος που θα βγεί ένα ολοκληρωμένο σύστημα στην παραγωγή ώστε να ικανοποιούνται τα κριτήρια που παρουσιάσαμε σε προηγούμενη ενότητα για την σωστή λειτουργία ενός IDS. Τα πέντε αυτά βήματα θα παρουσιάσουμε αναλυτικά σε αυτήν την ενότητα.

4.1 Αρχιτεκτονική

Γενικά όλες οι εργασίες μένουν στην δημιουργία του μοντέλου μηχανικής μάθησης. Σκοπός της δική μας προσέγγισης είναι να συνδυάσουμε τις πιο κατάλληλες τεχνολογίες που χρειάζονται ώστε το μοντέλο που εκπαιδεύεται να μπορεί να λειτουργεί κανονικά. Υπάρχουν πολλές παράμετροι για αυτό και είναι δύσκολο να καλύψουμε όλες τις πτυχές του προβλήματος ώστε η τελική λύση να καλύπτει όλα τα πιθανά σενάρια χρήσης για την σωστή λειτουργία ενός τέτοιου συστήματος καθώς και την ικανοποίηση όλων των απαιτήσεων, όπως ορίστηκαν σε προηγούμενη ενότητα. Τους λόγους για αυτό θα τους εξηγήσουμε αναλυτικά σε επόμενη ενότητα. Εδώ απλά να αναφέρουμε ότι οδηγός για το συστατικό μέρος που θα πραγματοποιεί την πρόβλεψη είναι τα δεδομένα τα οποία θα συλλέγουμε για να τροφοδοτούμε το μοντέλο δεδομένα τα οποία θα εξαρτηθούν από το dataset που έχουμε διαθέσιμο και από την διαδικασία της επιλογής χαρακτηριστικών(feature selection). Είναι προφανές ότι θέλουμε να πετύχουμε την βέλτιστη απόδοση όπως οι εργασίες που αναφέρθηκαν. Εκτός από αυτό θέλουμε τον ελάχιστο αριθμό από χαρακτηριστικά ώστε το συνολικό σύστημα να μπορεί να καλύπτει με μικρό κόστος(resources) μεγάλο φόρτο κάτι που αποτελεί βασική απαίτηση. Για τον αριθμό των features θα ακολουθήσουμε την στρατηγική να κρατήσουμε τα βασικότερα με βάση το πόσο σημαντικά είναι για το μοντέλο. Πιο συγκεκριμένα στην διάρκεια της εκπαίδευσης θα αποκλείουμε χαρακτηριστικά ιεραρχικά με βάση την αξία τους για την διαδικασία της πρόβλεψης. Εκτός από αυτή την προσέγγιση θα μπορούσαμε να ακολουθήσουμε και άλλες. Για παράδειγμα θα μπορούσαμε να χωρίσουμε τα χαρακτηριστικά με βάση την πηγή που τα κάνει διαθέσιμα με σκοπό να τα περιορίσουμε και να κρατήσουμε εκείνα μόνο ώστε το IDS που θα δημιουργηθεί να είναι Host Based IDS ή Network Based IDS και όχι hybrid. Αυτή η στρατηγική είναι εύκολο να γίνει κατανοητή ιδιαίτερα στην περίπτωση ενός HIDS. Περισσότερα για αυτό σε επόμενη ενότητα. Στην συνέχεια παρουσιάζουμε τα βασικά μέρη από τα οποία αποτελείται το IDS που δημιουργούμε και παρουσιάζουμε τα βασικά χαρακτηριστικά τους. Πρέπει να τονίσουμε ότι πέρα από την βασική λειτουργικότητα που παρουσιάζουμε υπάρχουν και άλλα πράγματα που πρέπει να προστεθούν και να βελτιωθούν ώστε η λύση αυτή να είναι πλήρως λειτουργική και να μπορεί να βγει σε παραγωγή. Γενικά οι τεχνολογίες που θα χρησιμοποιήσουμε είναι δύο. Η πρώτη είναι αλγόριθμοι μηχανικής μάθησης ώστε να δημιουργήσουμε το μοντέλο πρόβλεψης. Εδώ επιλέγουμε την λύση των αλγορίθμων ML shallow για λόγους που θα εξηγήσουμε στην συνέχεια. Η δεύτερη τεχνολογία που θα χρησιμοποιήσουμε είναι λογισμικό Complex Event Processing για την πραγματικού χρόνου επεξεργασία που απαιτείται όταν παρακολουθείται ένας μεγάλος αριθμός από διεργασίες ή συνδέσεις στην περίπτωση που μία διεργασία δημιουργεί παραπάνω από μία σύνδεση. Η αρχιτεκτονική παρουσιάζεται παρακάτω και το κάθε συστατικό μέρος παρουσιάζεται αναλυτικά στην συνέχεια.



4.1.1 Μηχανισμός Καταγραφής

Το μέρος αυτό θα είναι υπεύθυνο για την συλλογή όλων των δεδομένων που σχετίζονται με τη διεργασία/σύνδεση που παρακολουθούμε. Υπάρχουν αρκετοί τρόποι για να γίνει αυτό αλλά εξαρτώνται από την περίπτωση. Διαφορές μπορούν να υπάρχουν ανάμεσα σε λειτουργικά συστήματα και εφαρμογές για το τι πληροφορία καταγράφουν και αυτό πρέπει να ληφθεί υπόψη για την σχεδίαση αυτού του συστατικού μέρους. Ιδανικά θα θέλαμε να καταγράφεται μόνο η απαραίτητη πληροφορία ώστε το overhead να είναι μικρό και επίσης να συγκεντρώνεται όλη η πληροφορία για όλες τις διεργασίες σε ένα μέρος αν και αυτό δεν αποτελεί σοβαρό πρόβλημα. Περισσότερα για το πως θα αντιμετωπίσουμε αυτό το βήμα στην ενότητα της Υλοποίησης.

4.1.2 Μηχανισμός Συσχέτισης

Το συστατικό μέρος αυτό θα είναι υπεύθυνο να συσχετίζει όλη την πληροφορία που γίνεται διαθέσιμη από το Μηχανισμό Καταγραφής και θα ανιχνεύει τα σύνθετα γεγονότα που μας ενδιαφέρουν για να τα κάνει διαθέσιμα στο συστατικό μέρος της Προ επεξεργασίας. Το λογισμικό που θα χρησιμοποιηθεί θα πρέπει να ικανοποιεί τις απαιτήσεις όπως τις αναφέραμε σε προηγούμενη ενότητα. Ειδικά αν ο αριθμός των διεργασιών/συνδέσεων που παρακολουθούνται είναι μεγάλος και έχουμε και μεγάλο αριθμό χαρακτηριστικών η χρήση τέτοιου λογισμικού προσφέρει πλεονεκτήματα σχετικά με την επίδοση καθώς εφαρμόζει βελτιστοποιήσεις που βελτιώνουν σημαντικά τις απαιτήσεις σε απόδοση και προσφέρει και λειτουργικότητα σχεδιασμένη ειδικά για τέτοια σενάρια. Περισσότερα για τον τρόπο που θα αντιμετωπίσουμε την δημιουργία αυτού του module στην συνέχεια.

4.1.3 Μηχανισμός προ επεξεργασίας

Αυτό το συστατικό μέρος θα είναι υπεύθυνο να εφαρμόζει την απαραίτητη προ επεξεργασία στα δεδομένα που παράγει ο Μηχανισμός Συσχέτισης. Η βασική του λειτουργικότητα είναι να παράγει κανονικοποιημένες τιμές για κάποια χαρακτηριστικά καθώς και κάθε άλλη απαραίτητη

προεπεξεργασία χρειάζεται, όπως feature hashing ή encoding categorical features. Με βάση αυτά θα πρέπει να έχει διαθέσιμες όλες τις απαραίτητες παραμέτρους από το dataset με το οποίο εκπαιδεύτηκε το μοντέλο.

4.1.4 Μηχανισμός Πρόβλεψης

Αυτός ο μηχανισμός θα έχει ως είσοδο την έξοδο του Μηχανισμού προ επεξεργασίας και θα έχει αποθηκευμένο το μοντέλο. Για κάθε σύνθετο γεγονός που λαμβάνει θα πραγματοποιεί πρόβλεψη και σε περίπτωση που ανιχνευτεί μία επίθεση θα συγκεντρώνει τα δεδομένα, δηλαδή κυρίως την διεργασία/σύνδεση που αφορά αυτή η επίθεση και τον χρόνο που ανιχνεύτηκε και θα παράγει μία ενημέρωση. Η ενημέρωση θα μπορεί να αφορά την προώθηση της πληροφορίας σε έναν μηχανισμό καταγραφής, την ενημέρωση ενός administrator με email ή την αναφορά ότι ανιχνευθηκε επίθεση στη γραμμή εντολών. Επιπλέον θα μπορούσαμε δίνοντας δικαιώματα υπερχρήστη σε αυτό το μηχανισμό να λαμβάνουμε και ενέργειες όπως απομόνωση και τερματισμός μίας διεργασίας.

4.2 Βήματα για την Υλοποίηση

4.2.1 Επιλογή για τις διεργασίες που θα παρακολουθούνται

Αυτό το βήμα αφορά στο να αποφασίσουμε ποιές διεργασίες θα παρακολουθεί το σύστημα που θα αναπτύξουμε. Θεωρητικά το βέλτιστο θα ήταν να παρακολουθούνται όλες καθώς υπάρχει κίνδυνος από όλα επειδή αποτελούν attack surface. Στην πράξη όμως ένα κριτήριο για επιλογή είναι το πόσο εκτεθειμένη είναι μία διεργασία στον επιτιθέμενο και ένα δεύτερο κριτήριο είναι το πόσο σημαντικό είναι αυτό για την εφαρμογή που εκτελείται στο host. Για παράδειγμα ένας server ο οποίος είναι προσβάσιμος από το internet και πραγματοποιεί μία σημαντική λειτουργία ικανοποιεί και τα δύο κριτήρια και θα πρέπει να παρακολουθείται από το IDS. Γενικά αυτό το βήμα εξαρτάται από κριτήρια και πολιτικές που εφαρμόζει ο χρήστης και η ίδια προσέγγιση ακολουθείται και από τις μεγάλης κλίμακας λύσεις που αντιμετωπίζουν την ανίχνευση απειλών όπως τα SIEMs. Επίσης όπως είδαμε και στην ενότητα με τις απαιτήσεις το σύστημα πρέπει να εκτελείται χωρίς να παρεμποδίζει την σωστή λειτουργία του συστήματος που παρακολουθεί οπότε η εκτέλεση του IDS δεν πρέπει να έχει μεγάλο overhead και με αποτέλεσμα να επιβαρύνει το σύστημα.

4.2.2 Τρόπος απόκτησης πληροφορίας

Αυτό το βήμα είναι ίσως το πιο απαιτητικό της διαδικασίας καθώς χρειαζόμαστε πληροφορία που προέρχεται από πολλές πηγές με αποτέλεσμα να υπάρχει η δυσκολία του ότι πρέπει να παρακολουθούμε ένα μεγάλο πλήθος από ροές πληροφορίας ώστε να έχουμε την απαραίτητη πληροφορία κάτι που αυξάνει δραματικά το overhead δεδομένου του ότι πρέπει να επεξεργαζόμαστε μεγάλο όγκο δεδομένων. Παράδειγμα αποτελούν όλα τα logs που καταγράφει για τις διεργασίες το Linux όπως αναφέραμε σε προηγούμενη ενότητα. Άλλος περιορισμός είναι ότι μπορεί να χρειαστεί να τρέχουμε άλλα εργαλεία όπως το wireshark για να έχουμε την απαραίτητη πληροφορία δεδομένου του ότι δεν υπάρχει στα στις καταγραφές. Το τελευταίο αποτελεί γενικότερο πρόβλημα διότι μπορεί να υπάρχει πληροφορία που απλά δεν είναι διαθέσιμη από καμία πηγή. Η πιο απλή λύση την οποία και θα ακολουθήσουμε αφορά διεργασίες που γνωρίζουν ότι θα παρακολουθούνται από το IDS. Σε αυτήν την προσέγγιση η κάθε εφαρμογή που θα παρακολουθείται θα έχει την υποχρέωση να παρέχει την απαραίτητη πληροφορία στο IDS μέσω ενός μηχανισμού. Για να είναι η εφαρμογή πιο εύκολη στην διαχείριση θα χρησιμοποιήσουμε τους μηχανισμούς καταγραφής(logging) που παρέχει το Linux. Τέλος θα πρέπει να αναφέρουμε ότι αυτό το βήμα εξαρτάται πλήρως από το μοντέλο που θα εκπαιδεύσουμε. Αυτό συμβαίνει γιατί από αυτό θα εξαρτηθεί τί πληροφορία θα πρέπει να έχουμε διαθέσιμη. Αρνητικό για αυτήν την προσέγγιση αποτελεί το γεγονός ότι τις περισσότερες φορές θα πρέπει να παρακολουθούνται και διεργασίες όπου δεν είναι ανοιχτού κώδικα με αποτέλεσμα να μην γίνεται να παρέμβουμε ώστε να την τροποποιήσουμε και να πετύχουμε το επιθυμητό αποτέλεσμα. Σε αυτήν την περίπτωση θα πρέπει να αρκεστούμε στα logs που υπάρχουν διαθέσιμα. Οδηγός σε αυτό το βήμα θα είναι το ίδιο το dataset που έχουμε διαθέσιμο και πιο συγκεκριμένα τα features στα οποία θα βασίσουμε το μοντέλο που θα εκπαιδεύσουμε.

Με βάση αυτά στην διαδικασία της επιλογής χαρακτηριστικών(feature selection) μπορούμε να αποκλείσουμε χαρακτηριστικά με κριτήριο όχι πόση πληροφορία μας δίνουν για την ανίχνευση μιας επίθεσης αλλά πόσο εύκολο είναι να τα έχουμε διαθέσιμα.

4.2.3 Συσχέτιση Πληροφορίας

Σε αυτό το βήμα θεωρούμε ότι έχουμε διαθέσιμη την πληροφορία από το προηγούμενο. Συνεπώς εδώ στόχος είναι να συσχετίσουμε τα γεγονότα ώστε να εξάγουμε την απαραίτητη πληροφορία που θα προωθηθεί στο μοντέλο μηχανικής μάθησης. Αυτή η λειτουργία θα μπορούσε να πραγματοποιηθεί με πολλούς τρόπους. Ένας από αυτούς και για την περίπτωση που αναφέραμε προηγουμένως, δηλαδή των διεργασιών που γνωρίζουν ότι παρακολουθούνται και όταν όλη η πληροφορία είναι διαθέσιμη να την προωθούν στο μοντέλο. Αυτή η λύση δεν είναι λάθος αλλά εδώ θα χρησιμοποιήσουμε ένα πιο γενικό μηχανισμό. Αυτός είναι το Complex Event Processing. Όπως

τον περιγράψαμε προηγουμένως αυτός ο μηχανισμός θα παρακολουθεί όλες τις ροές δεδομένων δηλαδή όλη την πληροφορία από τα processes και όταν ένα σύνθετο γεγονός ανιχνευθεί θα το προωθεί στο μοντέλο. Για το βήμα αυτό θα χρησιμοποιήσουμε το CEP Engine Esper.

4.2.4 Δημιουργία Μοντέλου ML

Αυτό το βήμα αποτελεί τον πυρήνα της εφαρμογής. Εδώ σκοπός είναι να εκπαιδεύσουμε ένα μοντέλο ώστε να κατηγοριοποιεί την παρατηρούμενη δραστηριότητα και να γίνεται η κατηγοριοποίηση σε πραγματικό χρόνο. Σκοπός του μοντέλου είναι να έχει καλή απόδοση με βάση ένα μετρικό από αυτά που αναφέραμε στην αντίστοιχη ενότητα το οποίο θα είναι αντιπροσωπευτικό της επίδοσης. Δεδομένου του ότι όπως θα δούμε στην επόμενη ενότητα το dataset είναι imbalanced θα πρέπει να επιλέξουμε το κατάλληλο μετρικό. Επίσης η επιλογή του αλγορίθμου αποτελεί πρόκληση και θα πρέπει να επιλεχθεί ένας αλγόριθμος ο οποίος να είναι γρήγορος στην διαδικασία της πρόβλεψης και αν είναι εφικτό να παρέχει σε κάποιο βαθμό επεξηγηματικότητα(explainability). Αυτό σημαίνει ότι ιδανικά θα πρέπει να παρέχει τις πληροφορίες με βάση τις οποίες κατέληξε στο συμπέρασμα καθώς και ποσοστό εμπιστοσύνης ως προς την πρόβλεψη για το αποτέλεσμα που προέκυψε. Επίσης ένα άλλο κριτήριο είναι η ταχύτητα του αλγορίθμου το οποίο είναι σημαντικό διότι θα πρέπει να αναλυει μεγάλο όγκο δεδομένων σε πραγματικό χρόνο. Στο σενάριο χρήσης που αναλύουμε δεν μας ενδιαφέρει ο χρόνος εκπαίδευσης αλλά μόνο ο χρόνος πρόβλεψης.

4.2.5 Ανάπτυξη Συστήματος

Εδώ όπως είδαμε στις απαιτήσεις που πρέπει να ικανοποιούν τα IDSs πρέπει να γίνει η ανάπτυξη με τρόπο που να ικανοποιεί τις απαιτήσεις. Ιδανικά το σύστημα θα πρέπει να εκτελείται σε προστατευμένο περιβάλλον με δικαιώματα που θα του επιτρέπουν να έχει πρόσβαση στην πληροφορία που είναι απαραίτητη για να λειτουργήσει. Επίσης θέλουμε να εκτελείται σε προστατευμένο περιβάλλον ώστε να μην είναι εύκολο να αλληλεπιδράσει μαζί του ένα κακόβουλο λογισμικό που κατάφερε να αποκτήσει πρόσβαση στο σύστημα και να το θέσει εκτός λειτουργίας. Ο καλύτερος τρόπος για να επιτευχθεί αυτό είναι ο πυρήνας του συστήματος δηλαδή ο μηχανισμός πρόβλεψης και συσχέτισης να τρέχουν σε ξεχωριστό εικονικό μηχάνημα και η μόνη αλληλεπίδραση να αφορά την ανάκτηση των καταγραφών που παράγονται.

Με βάση αυτούς τους περιορισμούς και τις απαιτήσεις θα προχωρήσουμε στην υλοποίηση στην επόμενη ενότητα.

5.Υλοποίηση

Σε αυτήν την ενότητα παρουσιάζουμε την διαδικασία της υλοποίησης η οποία πραγματοποιείται με βάση όσα αναφέρθηκαν στην προηγούμενη ενότητα. Εδώ για τους λόγους που εξηγήσαμε προηγουμένως, δηλαδή το ότι η υλοποίηση θα εξαρτηθεί από το μοντέλο που θα προκύψει από την εκπαίδευση, δεν θα παρουσιάσουμε την διαδικασία με την ίδια σειρά της προηγούμενης ενότητας. Θα ξεκινήσουμε με την διαδικασία της εκπαίδευσης του μοντέλου. Επίσης πριν από αυτό θα παρουσιάσουμε το dataset.

5.1 Σύνολο Δεδομένων

Το dataset που θα χρησιμοποιήσουμε είναι το NSL-KDD**[125]**. Το dataset αυτό έχει προκύψει από το KDD99 dataset. Προκειται για βελτιωμένη έκδοση του, όπου οι βασικές αλλαγές είναι οι εξής. Πρώτον ότι έχουν διαγραφεί όλες οι επαναλαμβανόμενες εγγραφές ώστε ο αλγόριθμος που θα εκπαιδευτεί με βάση το dataset να μήν είναι biased. Δεύτερον υπάρχουν επαρκή δεδομένα για να εκπαιδευτεί ένα μοντέλο. Τρίτον ο αριθμός των επιλεγμένων εγγραφών από κάθε επίπεδο δυσκολίας είναι αντιστρόφως ανάλογος του ποσοστού στο KDD99.

Με βάση αυτά συνεχίζουμε παρουσιάζοντας τα χαρακτηριστικά(features) από τα οποία αποτελείται το dataset. Αυτά τα χαρακτηριστικά μπορούν να χωριστούν σε τέσσερις κατηγορίες**[125]**. Αυτή η κατηγοριοποίηση θα μας βοηθήσει στην συνέχεια στο πρόβλημα που προκύπτει για το πως θα αποκτήσουμε την απαραίτητη πληροφορία για να τροφοδοτούμε το μοντέλο με δεδομένα. Οι κατηγορίες είναι οι Intrinsic, Content, Time based, Host based. Για κάθε κατηγορία περιγράφουμε το νόημα των features.

- Intrinsic

Σε αυτή την κατηγορία ανήκουν χαρακτηριστικά τα οποία μπορούμε να εξάγουμε από τις κεφαλίδες του πακέτου χωρίς να χρειαστεί να αναλύσουμε το φορτίο του πακέτου.

Duration	Η διάρκεια της κάθε σύνδεσης	Dst_bytes	Ο αριθμός των bytes που εστάλησαν από destination σε source
Protocol type	Το πρωτόκολλο της κάθε	Land	Αν dest ip/port ίδια με source ip/port

	σύνδεσης		ίδια τότε 1 αλλιώς 0
Service	Το είδος της υπηρεσίας που χρησιμοποιήθηκε	Wrong Fragment	Συνολικός αριθμός wrong fragments στην σύνδεση.
Flag	Το status της σύνδεσης	Urgent	Αριθμός urgent πακέτων σε αυτή την σύνδεση.
Src_bytes	Ο αριθμός των bytes που εστάλησαν από source σε destination		

- Content

Σε αυτή την κατηγορία ανήκουν τα χαρακτηριστικά που μπορούμε να εξάγουμε εξετάζοντας συνολικά την δραστηριότητα της σύνδεσης και όχι ένα μεμονωμένο πακέτο δηλαδή κοιτώντας το φορτίο.

Hot	Αριθμός hot indicators	Num_root	Αριθμός ενεργειών που εκτελέστηκαν σαν root
Num_failed_logins	Αριθμός αποτυχημένων logins	Num_file_creations	Αριθμός δημιουργίας αρχείων
Logged_in	1 αν loged in 0 αλλιώς	Num_shells	Αριθμός shells που δημιουργήθηκαν
Num_compromised	Αριθμός compromised conditions	Num_access_files	Αριθμός ενεργειών σε access control αρχεία
Root_shell	1 αν έχει	Num_outboun	Αριθμός

	πετύχει root πρόσβαση	d_cmds	outbound εντολών σε ένα ftp session
Su_attempt ed	Αριθμός su προσπαθειών	is_hot_login	1 αν το login ανήκει σε root/admin
		is_guest_login	1 αν το login είναι guest

- Time based features

Σε αυτή την κατηγορία έχουμε χαρακτηριστικά που προκύπτουν από την ανάλυση της κίνησης σε ένα χρονικό παράθυρο δύο δευτερολέπτων. Εδώ καταγράφεται πληροφορία όπως πόσες συνδέσεις καταγράφηκαν προς το host. Όλες οι παράμετροι αφορούν μετρήσεις εντός 2 δευτερολέπτων.

Count	Αριθμός συνδέσεων προς το host σε 2 δευτερόλεπτα	Diff_srv_rate	Ποσοστό συνδέσεων που ήταν σε άλλο service από τοCount
Srv_count	Αριθμός συνδέσεων προς το ίδιο service(port) σε 2 δευτερόλεπτα	Same_srv_rate	Ποσοστό συνδέσεων που ήταν στο ίδιο service από το Count
Serror_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag s0,s1,s2,s3 μεταξύ των συνδέσεων από Count	Srv_diff_host_ rate	Ποσοστό συνδέσεων που ήταν προς διαφορετική συσκευή ανάμεσα στις συνδέσεις από το Srv_count
Srv_serror_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag		

	s0,s1,s2,s3 μεταξύ των συνδέσεων από Srv_count		

- Host based

Σε αυτή την κατηγορία αντί να κρατάμε μετρήσεις σχετικά με τον χρόνο κρατάμε μετρήσεις σχετικά με το host μηχάνημα.

Dst_host_count	Αριθμός συνδέσεων προς ένα host machine	Dst_host_srv_ diff_host_rate	Το ποσοστό των συνδέσεων που ήταν προς διαφορετικό destination μηχάνημα ανάμεσα σε αυτές του Dst_host_srv_count
Dst_host_srv_ count	Αριθμός συνδέσεων προς το ίδιο service(port)	Dst_host_serro r_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag s0,s1,s2,s3 ανάμεσα σε αυτές του dst_host_count
Dst_host_same _srv_rate	Ποσοστό συνδέσεων προς το ίδιο service από αυτές του Dst_host_count	Dst_host_srv_s error_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag s0,s1,s2,s3 ανάμεσα σε αυτές του dst_host_srv_c ount
Dst_host_diff_ srv_rate	Ποσοστό συνδέσεων προςδιαφορετικό service από αυτές του Dst_host_count	Dst_host_rerro r_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag REJ ανάμεσα σε αυτές του dst_host_count

Dst_host_same _src_port_rate	Ποσοστό των συνδέσεων που ήταν προς το ίδιο service μεταξύ αυτών του Dst_host_count	Dst_host_srv_r error_rate	Ποσοστό συνδέσεων που ενεργοποίησαν το flag REJ ανάμεσα σε αυτές του Dst_host_srv_count

Εδώ έχουμε 41 features αλλά δεν είναι όλα ίδιου τύπου. Χωρίζονται σε τρεις κατηγορίες. Αυτές είναι οι categorical, binary και numeric.

- Categorical

Εδώ έχουμε τρία features. Τα protocol type, Flag και service. Το protocol type παίρνει τρεις τιμές το Flag παίρνει δέκα και το service σαράντα.

- Binary

Εδώ έχουμε έξι features που έχουν τιμή 0 ή 1. Αυτά είναι τα Land, logged_in, root_shell, su_attempted, is_host_login, is_guest_login.

- Numeric

Όλα τα υπόλοιπα features ανήκουν σε αυτήν την κατηγορία και παίρνουν αριθμητικές τιμές

Στην συνέχεια παρουσιάζουμε τις επιθέσεις που περιλαμβάνει το dataset. Αυτές οι επιθέσεις χωρίζονται σε τέσσερις κύριες κατηγορίες. Αυτές είναι οι DoS, Probe, U2R, R2L.

- DoS

Αυτές οι επιθέσεις σκοπό έχουν παρεμποδίσουν την παροχή της υπηρεσίας από τους νόμιμους χρήστες της. Ένας απλός τρόπος για να γίνει αυτό είναι η εξάντληση των πόρων ενός συστήματος. Φυσικά υπάρχουν και πιο εξελιγμένοι τρόποι για να γίνει αυτό. Γενικά είναι μία ομάδα επιθέσεων που είναι εύκολο να ανιχνευθεί στις πιο απλές εκδοχές της.

- Probe

Αυτές οι επιθέσεις έχουν ως στόχο την ανακάλυψη πληροφορίας για ένα σύστημα. Αυτό μπορεί να περιλαμβάνει την εύρεση έγκυρων IP διευθύνσεων σε ένα δίκτυο, την ανίχνευση ανοικτών ports σε ένα server ή πληροφορίες για τις τεχνολογίες που χρησιμοποιεί μία web εφαρμογή. Ακόμα πολλές φορές κατατάσσεται σε αυτήν την κατηγορία και η συλλογή πληροφοριών για χρήστες του συστήματος ώστε να χρησιμοποιηθούν σε σενάρια κοινωνικής μηχανικής. Αυτή η κατηγορία περιλαμβάνει επιθέσεις που μπορεί να είναι πολύ εύκολο να ανιχνευθούν, όπως για παράδειγμα το port scanning, καθώς και άλλες που δεν είναι τόσο εύκολο να γίνουν αντιληπτές.

- User to Root(U2R)

Σε αυτές τις επιθέσεις θεωρούμε ότι ο επιτιθέμενος έχει ήδη πρόσβαση στο σύστημα με κάποιον λογαριασμό και στόχος είναι να αποκτήσει πρόσβαση ως υπερχρήστης. Αυτές οι επιθέσεις δεν είναι τόσο εύκολο να γίνουν αντιληπτές και είναι πιο σπάνιες σε σχέση με τις προηγούμενες δύο κατηγορίες. Αυτή η κατηγορία είναι γνωστή και ως privilege escalation.

- Remote to Local(R2L)

Εδώ ο επιτιθέμενος δεν έχει πρόσβαση στο σύστημα και αλληλεπιδρά από απομακρυσμένη τοποθεσία για να αποκτήσει κάποια πρόσβαση στο σύστημα. Και αυτή η κατηγορία είναι πιο σπάνια αλλά όχι τόσο δύσκολη στην ανίχνευση σε σχέση με την U2R.

Στην συνέχεια παρουσιάζουμε τις επιθέσεις που περιλαμβάνει το dataset για κάθε κατηγορία.

Attack Class	Attack type
DoS	Back, Land, Neptune, Pod, Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named

Εδώ να σημειώσουμε ότι παρόλο που το dataset περιλαμβάνει μικρό αριθμό επιθέσεων το μοντέλο που θα εκπαιδευτεί θα μπορεί να αναγνωρίζει και άλλες άγνωστες επιθέσεις καθώς συμπεριφέρονται με τον ίδιο τρόπο με αυτές που υπάρχουν στο dataset. Φυσικά αν είχαμε στην διάθεση μας ένα πιο πλήρες dataset η απόδοση θα ήταν καλύτερη.

Τέλος να αναφέρουμε ότι το dataset έρχεται χωρισμένο σε train και test. Το train αποτελείται από 125793 εγγραφές. Το test αποτελείται από 22544 εγγραφές. Το dataset είναι κατα πολύ imbalanced με την κλάση U2R να έχει τις λιγότερες εγγραφές από τις υπόλοιπες κατηγορίες. Αυτό το πρόβλημα θα το αντιμετωπίσουμε στην επόμενη ενότητα.

5.2 Διαδικασία Δημιουργίας Μοντέλου

Σε αυτήν την ενότητα θα περιγράψουμε την διαδικασία που ακολουθούμε για να δημιουργήσουμε το μοντέλο. Στόχος εδώ είναι πέρα από την καλή απόδοση του μοντέλου σχετικά με το κομμάτι της πρόβλεψης να αντιμετωπίσουμε και άλλα θέματα που παρουσιάστηκαν στις προηγούμενες ενότητες. Αυτά είναι τα εξής.

- Πρώτον το μοντέλο θα πρέπει να είναι όσο πιο απλό γίνεται πράγμα που σημαίνει ότι θα πρέπει να εξαρτάται από όσο το δυνατόν λιγότερα χαρακτηριστικά γίνεται. Αυτό το θέλουμε επειδή το μοντέλο θα πρέπει να έχει πολύ χαμηλό latency για να λειτουργεί σε πραγματικό χρόνο και να επεξεργάζεται μεγάλο όγκο πληροφορίας. Επίσης αν έχουμε λιγότερα χαρακτηριστικά μειώνεται και το overhead της καταγραφής και επεξεργασίας πληροφορίας που θα γίνεται πριν τα δεδομένα προς εξέταση καταλήξουν στο μοντέλο.
- Δεύτερον το μοντέλο θα πρέπει να έχει την ιδιότητα της επεξηγηματικότητας(explainability) σε κάποιο βαθμό. Αυτό το θέλουμε για να αξιολογήσουμε το μοντέλο στην πράξη.

Σχεδιαστική απόφαση του συστήματος που αναπτύσσουμε είναι να χρησιμοποιήσουμε αλγόριθμο shallow για τους εξής λόγους. Το dataset που έχουμε στην διάθεση μας δεν αποτελείται από raw data αλλά από κάλα ορισμένα handcrafted features. Το γεγονός αυτό μας οδηγεί αλλα και μας περιορίζει ταυτόχρονα να χρησιμοποιήσουμε shallow μοντέλο. Ο δεύτερος λόγος είναι ότι σε πολλά παρόμοια σενάρια χρήσης τα shallow μοντέλα αποδίδουν ικανοποιητικά και ο τρίτος λόγος είναι το ότι το dataset δεν περιέχει υπερβολικά πολλές εγγραφές ώστε να δικαιολογείται η χρήση deep μοντέλων.

5.2.1 Δημιουργία νέου συνόλου δεδομένων(Dataset)

Εδώ στόχος είναι να δημιουργήσουμε ένα dataset από τα δύο αρχεία που αποτελείται το NSL-KDD. Το μέγεθος του dataset θα είναι 5000 samples και θα προσπαθήσουμε να το κάνουμε balanced σε αντίθεση με το αρχικό δηλαδή να έχουμε 1000 samples από κάθε κατηγορία επίθεσης καθώς και την Normal κίνηση. Επειδή η κλάσση U2R έχει μόνο μερικές εκατοντάδες δείγματα θα προσπαθήσουμε να αντιμετωπίσουμε το πρόβλημα αυτό με δύο μεθόδους όπως θα δούμε παρακάτω. Τα βήματα που ακολουθούμε εδώ είναι τα εξής.

- Πρώτον ενώνουμε τα δύο CSVs train και test
- Δεύτερον κάνουμε annotate κάθε δείγμα σε κάποια από τις τέσσερις κλάσεις με βάση την στήλη που δείχνει το υπο είδος της επίθεσης.

- Τρίτον κρατάμε από κάθε κλάση από τις τέσσερις καθώς και για την Normal 1000 samples. Αυτό το κάνουμε με τυχαίο τρόπο ώστε να αποφύγουμε προβλήματα σχετικά με το bias προς συγκεκριμένα δείγματα. Επειδή στην U2R έχουμε μόνο μερικές εκατοντάδες τα κρατάμε όλα.

Τέλος σώζουμε το νέο dataset ώστε να συνεχίσουμε με αυτό. Το script που έτρεξε είναι το **script1.**

5.2.2 Προ επεξεργασία Δεδομένων

Εδώ επεξεργαζόμαστε το csv που δημιουργήσαμε στο προηγούμενο με σκοπό να φέρουμε όλα τα χαρακτηριστικά σε μορφή κατάλληλη για τον αλγόριθμο. Αρχικά πρέπει να αντιμετωπίσουμε τα categorical features. Υπάρχουν πολλοί τρόποι να γίνει αυτό. Για το feature protocol και flag τα οποία παίρνουν λίγες τιμές(3 και 10) θα μπορούσαμε να κάνουμε one hot encoding για το protocol ,business logic για το flag και για το service το οποίο παίρνει 40 τιμές θα πρέπει να βρούμε άλλο τρόπο. Ο μόνος τρόπος που καταφέραμε να βρούμε είναι το feature hashing scheme. Αυτός ο αλγόριθμος μπορεί να έχει ως είσοδο strings και να παράγει vectors με προκαθορισμένο μέγεθος, έτσι δεν θα έχουμε διπλασιασμό των features όπως θα γινόταν με το one hot. Μία εναλλακτική θα ήταν να εφαρμόσουμε business logic και να ενοποιήσουμε τις κατηγορίες. Έτσι για το protocol που παίρνει μόνο 3 τιμές θα κάνουμε onehot. Για το flag θα χρησιμοποιήσουμε business logic αφού παρατηρούμε από την επεξήγηση των πιθανών τιμών καθώς και από τα counts που παρουσιάζονται στο dataset οι τιμές δείχνουν συνδέσεις που έχουν επιτυχία και συνδέσεις σε κάποιο στάδιο δεν φαίνονται ενεργές/συνηθισμένες. Εδώ κάποιες τιμές είναι πιο αντιπροσωπευτικές της κατάστασης και δίνουν μεγαλύτερη πληροφορία όπως πχ η S0 με την S1 οπότε θα μπορούσαμε να ερμηνεύσουμε την κατηγορία αυτή ως ordinal categorical και να χρησιμοποιήσουμε την πληροφορία από τα flags με ανάλογο τρόπο ώστε να είναι συγκρίσιμες οι τιμές τους. Για ευκολία αυτό το attribute θα το μετατρέψουμε σε binary όπου η μία τιμή θα δείχνει αν το flag είναι ύποπτο ή όχι. Στην πρώτη κατηγορία θα είναι το flag SF και στην άλλη όλα τα υπόλοιπα. Για ευκολία το flag OTH(other) θα το βάλουμε στην κατηγορία με τα ύποπτα αντί να δημιουργήσουμε μία κατηγορία μόνο για αυτό. Αυτό το δικαιολογούμε με το σκεπτικό ότι για να μην είναι στην κατηγορία με την συνηθισμένη δραστηριότητα μάλλον θα είναι ύποπτο. Οι επεξηγήσεις για τα flags είναι οι παρακάτω:

STATUS FLAG OF THE CONNECTION

State	Meaning
SF	Normal SYN/FIN completion
REJ	Connection rejected, Initial SYN elicited a RST in reply
S0	State 0: initial SYN seen but no reply
S1	State 1: connection established (SYN's exchanged), nothing further seen
S2	State 2: connection established, initiator has closed their side
S3	State 3: connection established, responder has closed their side
RSTO	Connection reset by the originator
RSTR	Connection reset by the responder
OTH	Other, a state not contemplated here.

Τέλος για το feature service θα κάνουμε feature hashing**[126]** με παράμετρο 5.Δεν βρήκαμε guidelines/rule of thumb για το πώς να επιλέξουμε το μέγεθος του vector με βάση το σύνολο των στοιχείων της κλάσης. Πρόκειται για σημείο που θέλει πειραματισμό ως προς τα collisions που πιθανόν να προκύψουν οπότε είναι trial/error διαδικασία. Στην συνέχεια έχουμε numeric attributes. Μεγάλο μέρος τους δείχνουν ρυθμό και έχουν εύρος [0,1] οπότε όσα αφορούν τέτοια παρατήρηση θα μείνουν ως έχουν. Για τα υπόλοιπα θα εφαρμόσουμε normalization ώστε να βρεθούν και αυτά στο [0,1].Παρατηρούμε ότι το num_outbound_cmds δίνει NaN μετά το normalization, αυτό συμβαίνει διότι όλες οι τιμές είναι 0 συνεπώς δεν δίνει πληροφορία και στην συνέχεια θα το κάνουμε drop. Το script που έτρεξε είναι το **script2** και κάνει normalization,hash trick,business logic. Τέλος σώζουμε το csv με το οποίο και θα συνεχίσουμε.

5.2.3 Επιλογή Αλγορίθμου

Για τον αλγόριθμο επιλέγουμε τον Decision Tree. Αυτή την επιλογή την αιτιολογούμε για δύο λόγους. Ο πρώτος είναι για την απαίτηση της επεξηγηματικότητας(explainability) διότι είναι πιο εύκολο να δούμε πως προέκυψε ένα αποτέλεσμα σε σχέση με άλλους αλγορίθμους όπως ο SVC. Ο δεύτερος λόγος είναι το ότι ο Decision Tree είναι πολύ γρήγορος στο χρόνο πρόβλεψης. Άλλος ένας λόγος είναι ότι είναι πιο εύκολο να γίνει incremental learning όταν προκύψουν περισσότερα δεδομένα.

5.2.4 Πρόβλημα με τον αριθμό δειγμάτων ανα κλάση

Εδώ θα πραγματοποιήσουμε τρία πειράματα ώστε να πάρουμε ένα baseline απόδοσης και να συγκρίνουμε τις δύο μεθόδους που μπορούμε να εφαρμόσουμε ώστε αντιμετωπίσουμε το class imbalance πρόβλημα. Ο πρώτος είναι να χρησιμοποιήσουμε διαφορετικά βάρη για κάθε κλάση(class weights) ώστε ο αλγόριθμος να δίνει μεγαλύτερη σημασία στις κλάσεις με τα λιγότερα δείγματα. Ο

δεύτερος τρόπος είναι να κάνουμε smote. Η μέθοδος αυτή δημιουργεί συνθετικά δείγματα από τα αρχικά ώστε να αντιμετωπιστεί το class imbalance πρόβλημα όπως το έχουμε περιγράψει σε προηγούμενη ενότητα. Το πρώτο πείραμα αφορά εκπαίδευση του αλγορίθμου χωρίς κανένα μέτρο αντιμετώπισης του imbalance. Το δεύτερο πείραμα είναι με χρήση βαρών και το τρίτο είναι με το smote. Να σημειωθεί εδώ ότι οι κλάσεις εμφανίζονται πάντα με την σειρά DoS,Normal,Probe,R2L,U2R. Επειδή τα αποτελέσματα διαφέρουν σε κάθε εκτέλεση λόγω της τυχαιότητας στο train/test split θα τρέξουμε τα πειράματα 1000 φορές και θα πάρουμε τον μέσο όρο.

Τα αποτελέσματα για την περίπτωση κανενός μέτρου για την αντιμετώπιση του προβλήματος είναι τα εξής.

```
Accuracy baseline for DT: 0.9456156832298156
Precision baseline for DT: [0.97986017 0.92848064 0.97007854 0.92952817 0.72951489]
Recall baseline for DT: [0.97977673 0.92945173 0.96638168 0.92929365 0.745999  ]
```

Τα αποτελέσματα για τη μέθοδο διαφορετικά βάρη για κάθε κλάση είναι τα εξής.

```
Accuracy with class weights for DT: 0.942828416149071
Precision with class weights for DT: [0.97573332 0.92745908 0.96434262 0.92697062 0.74046506]
Recall with class weights for DT: [0.97798494 0.92437003 0.96439113 0.92769195 0.73503992]
```

Τα αποτελέσματα για την μέθοδο smote είναι τα εξής.

```
Accuracy with SMOTE for DT: 0.9455093167701885
Precision with SMOTE for DT: [0.97778205 0.9302155  0.96756439 0.9329871  0.73178427]
Recall with SMOTE for DT: [0.97893345 0.92577156 0.96504083 0.92926081 0.7923157 ]
```

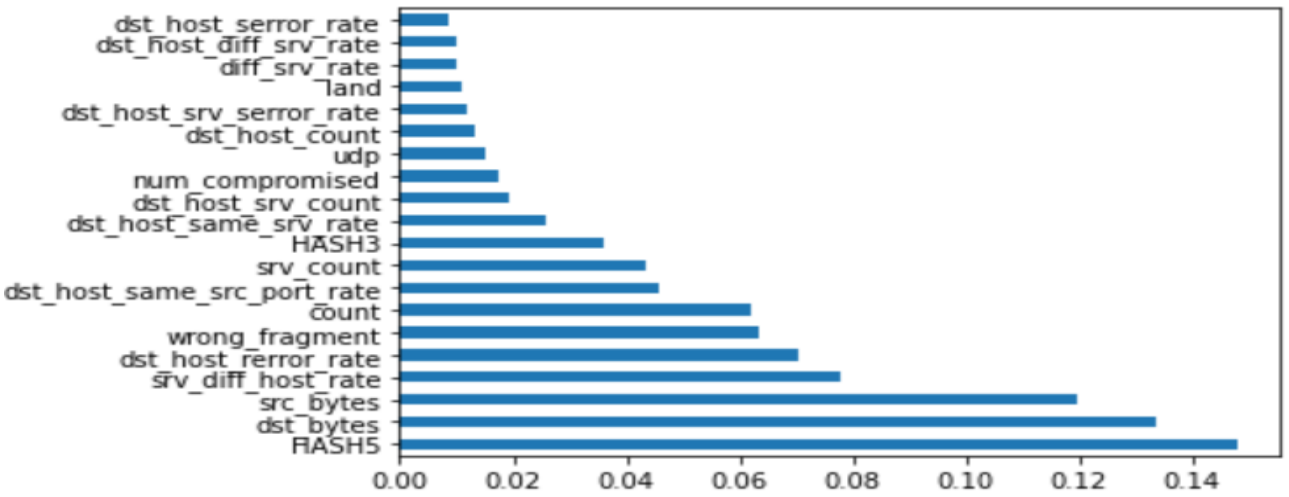
Με βάση αυτά βλέπουμε ότι όλες οι κλάσεις τα πάνε πολύ καλά στο metric του recall το οποίο μας ενδιαφέρει περισσότερο και επίσης το συνολικό accuracy είναι ίδιο και για τις τρεις περιπτώσεις. Η κλάση U2R για την οποία δεν έχουμε παρα μερικές εκατοντάδες δείγματα όπως αναμενόταν μένει πίσω. Η μέθοδος με βάρη(class weights) δίνει ίδια αποτελέσματα με την περίπτωση χωρίς κανένα μέτρο αλλά η μέθοδος smote δίνει καλύτερα αποτελέσματα από την πρώτη αλλά και δεύτερη περίπτωση. Με βάση αυτά θα έπρεπε να συνεχίσουμε με βάση την λύση του smote. Δεν θα το κάνουμε διότι αυτό προϋποθέτει ότι θα χωρίσουμε από τώρα το dataset σε train και test. Αυτό αποτελεί περιορισμό για τα πειράματα που θα πραγματοποιηθούν στην συνέχεια γιατί όπως είδαμε το train/test split παίζει πολύ μεγάλο ρόλο στα αποτελέσματα. Αυτό το βήμα θα μπορούσαμε να το

κάνουμε αφού έχουμε πραγματοποιήσει τα πειράματα της επόμενης ενότητας. Στην περίπτωση του smote θα επανέλθουμε. Το script που έτρεξε είναι το **script3.**

5.2.5 Επιλογή χαρακτηριστικών/Μείωση της διάστασης

Σε αυτή την ενότητα θα προσπαθήσουμε να βρούμε ποια είναι τα κυριότερα χαρακτηριστικά(features) που προσφέρουν την μέγιστη πληροφορία για να κατηγοριοποιήσει ο αλγόριθμος Decision Tree ένα δείγμα. Αυτό όπως αναφέραμε και προηγουμένως το κάνουμε ώστε να είναι πιο γρήγορος ο αλγόριθμος στην προ επεξεργασία και στην πρόβλεψη καθώς και για να είναι πιο εύκολο το έργο μας στην φάση της συγκέντρωσης της απαραίτητης πληροφορίας. Εδώ θα δοκιμάσουμε δύο τρόπους και θα συγκρίνουμε τα αποτελέσματα που δίνουν. Ο πρώτος είναι να πάρουμε τα πρώτα δέκα χαρακτηριστικά(features) που θεωρεί ο Decision Tree πιο σημαντικά. Ο δεύτερος θα είναι να κάνουμε πίνακα συσχέτισης(correlation matrix) στα χαρακτηριστικά μόνο, καθώς οι κλάσεις είναι categorical άρα δεν μπορούμε να δείξουμε συσχέτιση μεταξύ χαρακτηριστικών/κλάσεων. Ο πίνακας συσχέτισης μας δίνει την συσχέτιση ανάμεσα στα χαρακτηριστικά. Από θεωρία ξέρουμε ότι ψάχνουμε για την ελάχιστη συσχέτιση μεταξύ των χαρακτηριστικών δηλαδή κοντά στο μηδέν. Άρα θα υπολογίσουμε για κάθε γραμμή ή στήλη του πίνακα το άθροισμα των συσχετίσεων και θα κρατήσουμε αυτά που έχουν το ελάχιστο. Δεν δίνουμε το heatmap εδώ διότι είναι δύσκολο να δει κάποιος τις τιμές. Θα ξεκινήσουμε με τα 20 σημαντικότερα features και θα προσπαθήσουμε να τα μειώσουμε με τον περιορισμό να παραμένει η απόδοση του αλγορίθμου υψηλή. Πρόκειται για διαδικασία που πρέπει να γίνει πειραματικά και για κάθε επιλογή να τρέχουμε την εκπαίδευση και να ελέγχουμε το αποτέλεσμα.

Για την πρώτη περίπτωση δηλαδή τα 20 πιο σημαντικά features σύμφωνα με τον Decision Tree έχουμε τα παρακάτω αποτελέσματα.



Βλέπουμε ότι το feature service παίζει πολύ σημαντικό ρόλο καθώς δύο από τα features που προκύπτουν μετά το hashing της παραμέτρου αυτής εμφανίζονται στα δέκα κυριότερα. Το script που έτρεξε εδώ είναι το **script4.**

Για την δεύτερη περίπτωση δηλαδή αυτή του correlation matrix έχουμε τα παρακάτω αποτελέσματα. Το script που έτρεξε είναι το **script5**.

```
src_bytes                1.540186
dst_bytes                1.668540
is_host_login            1.946543
su_attempted             2.102771
num_failed_logins        2.700340
urgent                   3.199463
land                     3.641673
num_file_creations       3.737264
num_shells               3.794490
duration                 3.809925
num_access_files         3.985942
hot                      4.073520
is_guest_login           4.132527
num_root                 4.392071
wrong_fragment           4.500331
num_compromised          4.606069
root_shell               5.139466
HASH2                    5.219184
srv_count                5.533944
srv_diff_host_rate       5.740913
```

Βλέπουμε ότι με βάση την μέθοδο του πίνακα συσχέτισης προκύπτουν διαφορετικά χαρακτηριστικά στα σημαντικότερα. Στην συνέχεια θα τρέξουμε πάλι την εκπαίδευση ώστε να δούμε τι επίπτωση έχει στην ακρίβεια η μείωση της διάστασης στο μισό. Δηλαδή από 40 χαρακτηριστικά σε 20. Τρέχουμε πάλι το πείραμα 1000 φορές και παίρνουμε μέσο όρο Το script που έτρεξε είναι το **script6.**

Για την περίπτωση των χαρακτηριστικών που προέκυψαν από τον Decision Tree έχουμε τα παρακάτω αποτελέσματα.

```
Accuracy for DT: 0.9356793478260866
Precision for DT: [0.97015015 0.9198042  0.96420455 0.91588431 0.69745683]
Recall for DT: [0.97268224 0.91286787 0.963463   0.9216461  0.68622678]
```

Βλέπουμε ότι η απόδοση με βάση το συνολικό accuracy παραμένει ίδια αλλά η κλάση με τα λιγότερα δείγματα(minority class) έπεσε σημαντικά με αυτά τα χαρακτηριστικά.

Στην συνέχεια έχουμε τα αποτελέσματα για την περίπτωση των features που προέκυψαν από την μέθοδο του πίνακα συσχέτισης.

```
Accuracy for DT: 0.8856653726708089
Precision for DT: [0.95546754 0.92182348 0.7936787  0.906567   0.77088154]
Recall for DT: [0.94398286 0.85036561 0.95208519 0.80977309 0.78202323]
```

Εδώ βλέπουμε ότι το συνολικό accuracy έπεσε σημαντικά. Αλλά η κλάση U2R κέρδισε πολύ στο recall για την ακρίβεια δέκα μονάδες. Οι κλάσεις που έχασαν πολύ είναι οι Probe και Normal. Έτσι εδώ μένει να αποφασίσουμε αν η υψηλότερη απόδοση για την πιο σημαντική κλάση δηλαδή την U2R η οποία αντιπροσωπεύει τις πιο επικίνδυνες επιθέσεις αξίζει τη μείωση της επίδοσης στην ακρίβεια συνολικά και του μεγαλύτερου ποσοστού false positives για την επίθεση Probe και την κανονική κίνηση. Αποφασιζουμε με βάση τα αποτελέσματα να συνεχίσουμε με τα χαρακτηριστικά που προέκυψαν από την μέθοδο του πίνακα συσχέτισης καθώς φαίνεται να δίνει καλύτερα αποτελέσματα.

Στην συνέχεια τρέχουμε το ίδιο πείραμα με τα 10 κυριότερα features και έχουμε τα εξής αποτελέσματα.

```
Accuracy for DT: 0.7820923913043484
Precision for DT: [0.96744394 0.92874418 0.56666644 0.90976796 0.80917595]
Recall for DT: [0.71354321 0.83291355 0.96958579 0.62875728 0.75827265]
```

Εδώ βλέπουμε πως η απόδοση πλέον έχει πέσει πολύ. Αυτό συμβαίνει διότι η μέθοδος αυτή είναι λάθος. Το να κρατήσουμε τα 20 καλύτερα από τα 40 δεν σημαίνει ότι μπορούμε με την ίδια ιεράρχηση να κρατησουμε τα 10 καλύτερα από τα 20 καθώς οι συσχετίσεις μεταξύ τους αλλάζουν όταν διαγράψουμε τα μισά. Με βάση αυτή την παρατήρηση επαναλαμβάνουμε το πείραμα με το dataset των 20 καλύτερων χαρακτηριστικών και ξανακάνουμε πίνακα συσχέτισης σε αυτά τα 20. Στην συνέχεια κρατάμε τα 10 καλύτερα δηλαδή αυτά με το μικρότερο score. Τα αποτελέσματα είναι τα παρακάτω.

```
Accuracy for DT: 0.8737973602484472
Precision for DT: [0.94565784 0.93059027 0.78015463 0.8806828  0.71520759]
Recall for DT: [0.95518164 0.82726157 0.92617664 0.80013207 0.7530919 ]
```

Και για τα 9 καλύτερα έχουμε τα παρακάτω αποτελέσματα.

```
Accuracy for DT: 0.8331576086956527
Precision for DT: [0.93638604 0.93473117 0.71989397 0.79842025 0.71265167]
Recall for DT: [0.86895834 0.82330788 0.85622338 0.79388868 0.7544156 ]
```

Βλέπουμε πως το δέκατο χαρακτηριστικό παίζει πολύ σημαντικό ρόλο στην διαδικασία για τις κλάσεις DoS και Probe και το συνολικό accuracy πέφτει 4 μονάδες χωρίς αυτό το feature. Με βάση τα τελευταία πειράματα η φάση της επιλογής χαρακτηριστικών δέν μπορεί να προσφέρει άλλη βελτίωση. Παρόλα αυτά καταφέραμε μία πολύ καλή απόδοση μειώνοντας τον αριθμό των features κατά 75% από σαράντα σε δέκα. Επίσης η κλάση U2R έχει μία μικρή βελτίωση 1%από την καλύτερη όλων των πειραμάτων που πραγματοποιήθηκαν. Στην συνέχεια κλείνοντας αυτήν την ενότητα παρουσιάζουμε τα χαρακτηριστικά με τα οποία θα συνεχίσουμε. Με βάση αυτά δημιουργούμε το τελικό csv με το dataset.

dst_bytes	Number of data bytes transferred from destination to source in single connection	num_failed_logins	Count of failed login attempts
src_bytes	Number of data bytes transferred from	srv_count	Number of connections to the

	source to destination in single connection		same service (port number) as the current connection in the past two seconds
land	if source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0	wrong_fragment	Total number of wrong fragments in this connection
is_hot_login	1 if the login belongs to the ``hot'' list i.e., root or admin; else 0	su_attempted	1 if ``su root'' command attempted or used; 0 otherwise
duration	Length of time duration of the connection	srv_diff_host_rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count

Τα αποτελέσματα με βάση τα προηγούμενα πειράματα είναι ενθαρρυντικά ως προς την διαδικασία της συλλογής της πληροφορίας. Τα δέκα features που μείνανε μετά την διαδικασία του feature selection μπορούν εύκολα να συγκεντρωθούν απλά παρατηρώντας την δραστηριότητα μίας διεργασίας εκτός απο ένα το οποίο απαιτεί πληροφορία απο το δίκτυο. Φυσικά υπάρχουν περιορισμοί και θα αναφερθούμε αναλυτικά σε επόμενη ενότητα. Με βάση αυτά το IDS που δημιουργούμε θα είναι υβριδικό.

5.2.6 Ρύθμιση υπερ παραμέτρων

Σε αυτή την ενότητα με βάση την θεωρία πρέπει να βρούμε τις καλύτερες υπερ παραμέτρους για τον αλγόριθμο που επιλέξαμε με χρήση του dataset όπως προέκυψε από τις προηγούμενες ενότητες. Οι κύριες υπερ παράμετροι και οι τιμές που θα δοκιμάσουμε για τον αλγόριθμο Decision Tree είναι οι

- criterion gini/entropy
- splitter best/random
- max depth [16,17,18,19,20,21,22,23,24,25]
- min samples split [2,3,4,5]

Εδώ θα χρησιμοποιήσουμε τον αλγόριθμο GridSearchCV ο οποίος δοκιμάζει όλους τους πιθανούς συνδυασμούς των παραμέτρων και επιστρέφει το καλύτερο μοντέλο. Εδώ όταν λέμε το καλύτερο μοντέλο θα δοκιμάσουμε δύο περιπτώσεις. Η πρώτη είναι η μεγιστοποίηση του average recall όλων των κλάσεων. Στην δεύτερη περίπτωση θα ορίσουμε ώς κριτήριο το recall της minority class U2R. Το δεύτερο θα μπορούσαμε να το κάνουμε και στα προηγούμενα πειράματα αλλά θεωρούμε ότι η απόδοση συνολικά του μοντέλου ήταν σημαντικότερη.

Για την δεύτερη περίπτωση με κριτήριο το recall της U2R έχουμε τα εξής αποτελέσματα.

```
Accuracy Final model: 0.8672360248447205
Confusion Matrix final model :
 [[297   2   2   3   0]
 [  3 247  23  21   5]
 [ 16   7 305   9   0]
 [  0  12  55 240   5]
 [  0   0   3   5  28]]

Precision final model: [0.93987342 0.92164179 0.78608247 0.86330935 0.73684211]
Recall final model: [0.97697368 0.82608696 0.90504451 0.76923077 0.77777778]
```

Βλέπουμε ότι έχουμε ικανοποιητικό accuracy και το recall της U2R είναι κοντά στο 80%

Για την πρώτη περίπτωση με κριτήριο το average recall όλων των κλάσεων έχουμε τα εξής αποτελέσματα.

```
Accuracy Final model: 0.8781055900621118
Confusion Matrix final model :
 [[314   0   7   2   0]
 [  5 251  25  24   3]
 [ 13   4 296   6   0]
 [  2   6  45 243   7]
 [  0   3   1   4  27]]

Precision final model: [0.94011976 0.95075758 0.79144385 0.87096774 0.72972973]
Recall final model: [0.97213622 0.81493506 0.92789969 0.8019802  0.77142857]
```

Βλέπουμε ίδιο σχεδόν συνολικό accuracy και όλα τα metrics είναι περίπου στα ίδια οπότε κρατάμε το μοντέλο που προέκυψε από την περίπτωση με το μετρικό average recall.

Σε επόμενη ενότητα θα σχολιάσουμε τα αποτελέσματα. Το script που έτρεξε είναι το **script7.** Με βάση τα αποτελέσματα από το τελευταίο οι υπερ παράμετροι είναι οι εξής max_depth = 19, criterion = gini, splitter = best, min_samples_slpit = 2.

5.2.7 Σύγκριση με άλλους αλγορίθμους

Εδώ για λόγους σύγκρισης και με το ίδιο dataset που πραγματοποιήσαμε το προηγούμενο βήμα θα δοκιμάσουμε επιπλέον αλγορίθμους. Οι αλγόριθμοι που θα εξετάσουμε είναι οι Naive Bayes, MLP, KNN, Random Forest και SVC. Για τον τελευταίο δεν δίνουμε αποτελέσματα διότι για άγνωστο λόγο η εκπαίδευση δεν ολοκληρώθηκε ποτέ. Τα αποτελέσματα δίνονται παρακάτω.

Naive Bayes

```
Accuracy for NB: 0.468944099378882
Precision for NB: [0.87931034 0.62626263 0.68303571 0.33766234 0.5       ]
Recall for NB: [0.32797428 0.20462046 0.45808383 0.96621622 0.02272727]
```

MLP

```
Accuracy for MLP: 0.48757763975155277
Precision for MLP: [0.78873239 0.71653543 0.68292683 0.36608187 0.        ]
Recall for MLP: [0.35109718 0.28526646 0.39575972 0.96604938 0.        ]
```

KNN

```
Accuracy for KNN: 0.7538819875776398
Precision for KNN: [0.84149856 0.76727273 0.84848485 0.64       0.66666667]
Recall for KNN: [0.90402477 0.6986755  0.54193548 0.9        0.36363636]
```

Random Forest

```
Accuracy for RF: 0.8827639751552795
Precision for RF: [0.96865204 0.93726937 0.79197995 0.87407407 0.75862069]
Recall for RF: [0.94785276 0.84949833 0.96048632 0.76375405 0.88      ]
```

Με βάση αυτά βλέπουμε πως μόνο ο Random Forest έχει την ίδια απόδοση με τον αλγόριθμο Decision Tree που επιλέξαμε. Το notebook που έτρεξε εδώ είναι το δεύτερο.

5.2.8 Αξιολόγηση ταχύτητας Πρόβλεψης

Εδώ θα αξιολογήσουμε την ταχύτητα της πρόβλεψης για το μοντέλο που προέκυψε από την προηγούμενη ενότητα και το dataset που χρησιμοποιήθηκε επίσης στην προηγούμενη ενότητα. Εδώ έχουμε 4291 δείγματα και τρέχοντας το inference στο google colab καθώς θεωρούμε ότι είναι πιο κατάλληλο σε σχέση με ένα laptop με κριτήριο την διαθεσιμότητα των πόρων σε μία χρονική στιγμή έχουμε σαν αποτέλεσμα ένα εύρος τιμών με μέση τιμή περίπου 0.003 δευτερόλεπτα για όλο το dataset. Αυτό σημαίνει 7e-7 για ένα sample. Δεδομένης της φύσης των features μετά την διαδικασία του feature selection φαίνεται να είναι καλή απόδοση αλλά θα πούμε περισσότερα για αυτό στην ενότητα πειραματικά αποτελέσματα.

5.3 Ανάκτηση δεδομένων

Εδώ θυμίζουμε τα χαρακτηριστικά(features) που χρειαζόμαστε για να τροφοδοτούμε το μοντέλο με δεδομένα και παρουσιάζουμε από που μπορούμε να τα βρούμε.Τα περισσότερα όπως θα δούμε μπορούμε να τα έχουμε απλά παρακολουθώντας την δραστηριότητα μιας διεργασίας αλλά ακόμα και με αυτό τον τρόπο παρουσιάζονται προβλήματα. Τα χαρακτηριστικά είναι

- **dst_bytes** Ο αριθμός των bytes που εστάλησαν από κόμβο προορισμού σε κόμβο αφετηρίας. Αυτήν την πληροφορία μπορούμε να την έχουμε από μία διεργασία που παρακολουθείται και παράγει logs για την δραστηριότητα του αλλά χρειάζεται κανονικοποίηση(normalization). Αυτό αποτελεί πρόβλημα καθώς το min/max που κάναμε ακόμα και αν κρατήσουμε τις δύο αυτές τιμές στην πράξη αυτό μπορεί να βγει εκτός εύρους[0,1] πράγμα που θα μειώσει την επίδοση του μοντέλου. Μία λύση για αυτό είναι αν μία τιμή βγεί εκτός εύρους να την αντικαθιστουμε με τις ακραίες τιμές δίνοντας τιμή 0 ή 1.
- **src_bytes** το αντίστροφο του προηγούμενου. ότι ισχύει για το προηγούμενο ισχύει και για αυτό.
- **land** Αυτό το χαρακτηριστικό υποθέτουμε ότι μπορούμε να το έχουμε διαθέσιμο παρατηρώντας μία διεργασία.

- **is_hot_login** Αυτό το feature είναι πολύ εύκολο να το έχουμε και με τον ίδιο τρόπο με τα παραπάνω αλλά και από logs που κρατάει το Linux.
- **duration** Αυτό πάλι μπορούμε να το έχουμε παρακολουθώντας την διεργασία αλλά πρέπει επιπροσθέτως να έχουμε καταγράψει τον χρόνο εκκίνησης και τερματισμού της σύνδεσης. Αποτελεί πρόβλημα η περίπτωση όπου μετά από επιτυχημένη επίθεση U2R μετά από επιτυχημένη επίθεση R2L η σύνδεση θα παραμείνει ανοιχτή οπότε αυτή η πληροφορία δεν θα είναι ποτέ διαθέσιμη. Επίσης στην περίπτωση που θα δούμε παρακάτω με την χρήση CEP αποτελεί σημαντικό πρακτικό πρόβλημα. Θεωρούμε ότι την έχουμε με βάση τα γεγονότα που θα καταγράφουμε αλλά η περίπτωση αυτή πρέπει να αντιμετωπιστεί με άλλους τρόπους.
- **num_failed_logins** Αυτήν την πληροφορία είναι εύκολο να την αποκτησουμε είτε παρακολουθώντας την διεργασία είτε από τα logs του Linux όπως περιγράφηκαν σε προηγούμενη ενότητα.
- **wrong_fragment** Ο συνολικός αριθμός wrong_fragments για αυτή τη σύνδεση. Εδώ έχουμε πάλι το πρόβλημα της κανονικοποίησης και το ότι πρέπει να τερματιστεί η σύνδεση.
- **su_attempted** Αυτό το feature είναι πολύ εύκολο να το έχουμε παρακολουθώντας τη διεργασία ή από logs του Linux.
- **srv_Diff_host_rate** Το ποσοστό των συνδέσεων που ήταν σε διαφορετικά destination machines από αυτές που καταγράφηκαν στο srv_count.
- **srv_count** Αριθμός συνδέσεων στο ίδιο port(service) τα τελευταία 2 δευτερόλεπτα.

Με βάση αυτά βλέπουμε τρία κύρια ζητήματα. Το πρώτο είναι το πρόβλημα της κανονικοποίησης. Μία λύση είναι να έχουμε αποθηκευμένες τις max/min τιμές από το dataset αλλά δεν υπάρχει τίποτα που να εγγυάται ότι δεν θα βγει εκτός εύρους με αποτέλεσμα ο αλγόριθμος να δίνει υπο βέλτιστα αποτελέσματα. Για αυτό θα ακολουθήσουμε την στρατηγική να δίνουμε τις ακραίες τιμές σε όλες τις εκτός εύρους τιμές. Το δεύτερο είναι το πρόβλημα ότι κάποια από τα features θα είναι διαθέσιμα όταν τελειώσει μία σύνδεση πράγμα που σημαίνει ότι δεν θα είναι το η πρόβλεψη πραγματικού χρόνου ή ότι δεν θα πραγματοποιηθεί ποτέ αν δεν τερματιστεί η σύνδεση πράγμα πολύ πιθανό σε μία επιτυχημένη επίθεση. Το τρίτο είναι το ότι τα 9/10 features μπορούν να βρεθούν με ένα IDS που ελέγχει logs δηλαδή host IDS. Για την πληροφορία του δέκατου χρειαζόμαστε πληροφορία που αφορά πολλά hosts δηλαδή απο το δίκτυο. Με βάση την περιγραφή συμπεραίνουμε ότι αυτό το feature είναι σημαντικό για την περίπτωση των επιθέσεων probe όπως ένα port scanning. Για να έχουμε αυτή την πληροφορία θέλουμε ένα IDS σε επίπεδο δικτύου. Με βάση αυτά το IDS που θα φτιάξουμε κατηγοριοποιείται ώς hybrid. Για τα 2 features με αυτόν τον περιορισμό θεωρούμε ότι έχουμε την δυνατότητα να στέλνουμε αυτήν την πληροφορία με τον ίδιο μηχανισμό σε όλα τα hosts που παρακολουθούνται.

5.4 Συσχέτιση Γεγονότων

Σε αυτή την ενότητα θεωρούμε ότι έχουμε διαθέσιμο έναν μηχανισμό που παράγει γεγονότα όπως περιγράφηκε στην προηγούμενη ενότητα. Σε αυτήν την ενότητα σκοπός είναι να δημιουργήσουμε τον μηχανισμό που θα συσχετίζει γεγονότα όπως αυτά έρχονται με σκοπό να δημιουργεί την απαραίτητη πληροφορία όταν αυτή είναι διαθέσιμη και να τροφοδοτεί το μοντέλο ML. Για αυτή την διαδικασία θα χρησιμοποιήσουμε το CEP Engine Software Esper. Ο σκοπός εδώ είναι να έχουμε όσο πιο απλά γεγονότα από ροές που θα προέρχονται από τις διεργασίες που παρακολουθούμε και το esper να ανιχνεύει την ικανοποίηση του πολύπλοκου γεγονότος το οποίο είναι απλά η συγκέντρωση της πληροφορίας για κάθε πιθανή περίπτωση που παρακολουθούμε και η προώθηση του στο μοντέλο. Αυτό δεν σημαίνει ότι θα εξετάζεται μία περίπτωση για κάθε process καθώς θα υπάρχουν και processes τα οποία θα δέχονται παραπάνω από μία σύνδεση. Το τελευταίο αποτελεί σχεδιαστική επιλογή, εδώ θεωρούμε ότι το κάθε process διαχειρίζεται μία σύνδεση αλλά μπορεί εύκολα να γενικευτεί σε κάθε σενάριο. Με βάση αυτά το βασικό μέρος της διαδικασίας είναι ο σωστός ορισμός των απλών γεγονότων ώστε το Esper να μπορεί να κάνει την συσχέτιση της πληροφορίας. Στην συνέχεια θα παρουσιάσουμε τα βήματα για την δημιουργία αυτού του συστατικού μέρους. Δεν θα παρουσιάσουμε αναλυτικά όλη την λειτουργικότητα που παρέχει το Esper καθώς αυτό είναι πέρα από τους σκοπούς της εργασίας. Στην συνέχεια και για όπου χρειάζεται θα αναφέρουμε με βάση το reference**[128]** το που μπορεί ο αναγνώστης να βρεί περισσότερες πληροφορίες.

5.4.1 Δημιουργία Ροών Δεδομένων

Αρχικά πρέπει να ορίσουμε τα simple events. Για να ορίσουμε γεγονότα στο esper υπάρχουν πολλοί τρόποι και αναφέρονται στο **[128]** κεφάλαιο 3. Θα τους παρουσιάσουμε σε αυτήν την ενότητα.

- Ο πρώτος τρόπος είναι τα POJO(Plain Old Java Objects). Εδώ απλά δημιουργούμε κλάσσεις οι οποίες έχουν constructor και properties όπως κάθε αντικείμενο. Τα properties μπορούν να είναι primitives όπως String,int etc αλλά επίσης και άλλα αντικείμενα που πιθανός ορίστηκαν όπως άλλα simple events. Επίσης πρέπει να δημιουργήσουμε setters/getters για κάθε property.
- Ο δεύτερος είναι να χρησιμοποιήσουμε το java.util.Map. Εδώ έχουμε μία συλλογή από κλειδιά και τιμές που αντιστοιχούν σε αυτά.

- Ο τρίτος τρόπος είναι να δημιουργήσουμε ένα Object Array. Εδώ απλά περνάμε objects κάθε είδους αντικείμενα όπως και άλλα simple events αντίστοιχα με το προηγούμενο.
- Ο τέταρτος τρόπος είναι να να χρησιμοποιήσουμε το format Json.
- Ο πέμπτος τρόπος είναι να χρησιμοποιήσουμε το format Apache Avro.
- Ο έκτος τρόπος είναι να χρησιμοποιήσουμε format XML

Εδώ επιλέγουμε τον πρώτο τρόπο. Με βάση τα features που προέκυψαν από την προηγούμενη ενότητα θα δούμε τι properties χρειάζεται να έχει κάθε γεγονός ώστε να επιτευχθεί η λειτουργικότητα.

- Το πρώτο event που θα ορίσουμε θα το στέλνει μία διεργασία/σύνδεση όταν θα ξεκινάει στον μηχανισμό logging. Αντίστοιχα όπως αναφέραμε και προηγουμένως αυτό μπορεί να γενικευθεί όταν κάθε διεργασία δημιουργεί μία καινούργια σύνδεση. Τα απαιτούμενα properties που πρέπει να έχει είναι είναι ένα ID που θα το ξεχωρίζει από τα άλλα processes. Εδώ θα το ονομάσουμε sessionID και θα είναι τύπου int. Το δεύτερο property που χρειάζεται να ξέρουμε με βάση τα features που έχουμε είναι το port της σύνδεσης. Επίσης το property αυτό θα είναι τύπου int. Το τρίτο property που χρειαζόμαστε με βάση τα features είναι ένα timestamp. Το ορίζουμε και αυτό ώς τύπου int. Δημιουργούμε το γεγονός αυτό με το όνομα **InitEvent**.
- Το δεύτερο γεγονός θα είναι το event που θα σηματοδοτεί το τέλος του process. Για αυτό το γεγονός χρειαζόμαστε το sessionID αντίστοιχα με το προηγούμενο ώστε να μπορούμε να τα συσχετίσουμε. Επίσης χρειαζόμαστε ένα property timestamp το οποίο το ορίζουμε ως τύπου int. Δημιουργούμε το event αυτό με το όνομα **TerminationEvent.**
- Στην συνέχεια θα αντιμετωπίσουμε το feature src_bytes. Εδώ θα μπορούσαμε να βασιστούμε στην διεργασία που παρακολουθείται ώστε όταν τερματιστεί να μας δώσει τον συνολικό αριθμό bytes που εστάλησαν από από τη διεργασία/σύνδεση. Επειδή όπως περιγράψαμε και σε προηγούμενη ενότητα είναι πιο εύκολο να δημιουργήσουμε ένα wrapper γύρω από την κάθε send μέθοδο που χρησιμοποιείται σε ένα socket θα δημιουργήσουμε ένα γεγονός που θα καταγράφει κάθε φορά πόσα bytes εστάλησαν από την κλήση της send. Το συνολικό μέγεθος θα το αποκτήσουμε με την μέθοδο aggregation sum Επίσης πρέπει να συσχετίσουμε τέτοια events με τη διεργασία/σύνδεση που παρακολουθούμε. Με βάση αυτά θέλουμε ένα sessionID τύπου int και ένα property bytes τύπου int επίσης. Το γεγονός αυτό το ονομάζουμε **BytesSendEvent**.
- Στην συνέχεια θα αντιμετωπίσουμε το feature dst_bytes. Εδώ κάνουμε ακριβώς ότι και στο προηγούμενο οπότε δημιουργούμε δύο αντίστοιχα properties και ονομάζουμε αυτό το γεγονός **BytesReceiveEvent**.

- Στην συνέχεια θα ορίσουμε το γεγονός για το feature land. Θεωρούμε ότι στην εκκίνηση κάθε διεργασίας/σύνδεσης έχουμε αυτή την πληροφορία. Οπότε θέλουμε ένα property για την τιμή αυτή το οποίο ορίζουμε ως boolean καθώς και ένα property sessionID για να συσχετιζουμε τα γεγονότα. Το event αυτό το ονομάζουμε **LandEvent**.
- Στην συνέχεια θα ορίσουμε το γεγονός για το feature is_hot_login. Εδώ βασιζόμαστε επίσης στο ότι η διεργασία που παρακολουθείται θα δώσει αυτή την παράμετρο όταν γίνει αυθεντικοποίηση. Επίσης θα μπορούσαμε να το έχουμε από το μηχανισμό logging του λειτουργικού συστήματος. Εδώ θα έχουμε πάλι ένα property sessionID για να συσχετίσουμε το γεγονός με ένα process/connection που παρακολουθείται καθώς και ένα property τύπου boolean για την τιμή. Το γεγονός αυτό το ονομάζουμε **HotLoginEvent**.
- Στην συνέχεια θα ορίσουμε το γεγονός για το feature num_failed_logins. Εδώ δεν χρειαζόμαστε κάτι πέρα από το property sessionID για να συσχετίζουμε το γεγονός με μία διεργασία. Την απαραίτητη πληροφορία θα την έχουμε με το aggregation function count Ονομάζουμε αυτό το γεγονός **FailedLoginEvent**.
- Στην συνέχεια θα ορίσουμε το γεγονός για το feature wrong fragment. Εδώ βασιζόμαστε στο ότι κάθε φορα που ανιχνεύεται ένα wrong fragment σε ένα πακέτο η διεργασία θα το καταγράφει στα logs. Όπως και σε προηγούμενα χρειαζόμαστε ένα property sessionID για να συσχετίζουμε τα γεγονότα καθώς και άλλο ένα property το οποίο θα αναφέρει τον αριθμό wrong fragments. Εδώ μπορούμε να βασιστούμε στο ότι η διεργασία θα συλλέγει σωρευτικά την πληροφορία και θα την στέλνει όταν ολοκληρωθεί η σύνδεση ή θα παράγει ένα log entry για κάθε wrong fragment. Την πληροφορία που θέλουμε από αυτό το property θα την έχουμε με την aggregation function count. Επιλέγουμε τον δεύτερο τρόπο όποτε δεν χρειαζόμαστε το δεύτερο property σε αυτήν την κλάση. Ονομάζουμε το event αυτό **WrongFragEvent.**
- Στην συνέχεια θα ορίσουμε το event για το feature su_attemted. Εδώ θέλουμε πάλι ένα property sessionID για να συσχετιζουμε τα γεγονότα. Επειδή μας ενδιαφέρει αν πραγματοποιήθηκε επιτυχώς η όχι η εντολή su root αλλά όχι πόσες φορές δεν χρειαζόμαστε άλλο property, η παρουσία ή μη του γεγονότος είναι αρκετή. Ονομάζουμε αυτό το γεγονός **SuEvent**.
- Στην συνέχεια θα ορίσουμε το γεγονός για το feature duration. Εδώ μπορούμε να υπολογίσουμε αυτό το feature με βάση τα timestamps από τα events InitEvent και TerminationEvent οπότε δεν χρειάζεται να ορίσουμε καινούριο event. Εναλλακτικά θα μπορούσαμε να ορίσουμε ένα event το οποίο θα βασίζεται στο ότι η διεργασία θα υπολογίζει την παράμετρο αυτή και θα την καταγράφει στα logs πρίν τερματιστεί και στείλει το TerminationEvent.

- Στην συνέχεια θα ορίσουμε το γεγονός για το feature srv_count. Εδώ μας ενδιαφέρει να μάθουμε πόσες συνδέσεις στο ίδιο port(service) με την τρέχουσα σύνδεση που αναλύουμε πραγματοποιήθηκαν στα τελευταία δύο δευτερόλεπτα. Το event που θα δημιουργήσουμε πρέπει να έχει ένα property sessionID όπως και τα άλλα καθώς και ένα property port για να μπορούμε να συσχετιζουμε τα γεγονότα. Την πληροφορία που θέλουμε θα την έχουμε με το aggregation function count. Τον περιορισμό των δύο δευτερολέπτων θα πούμε πως θα τον αντιμετωπίσουμε στην συνέχεια. Το event αυτό το ονομάζουμε **SrvCountEvent**.
- Στην συνέχεια μένει να ορίσουμε το event για το feature srv_Diff_host_rate. Εδώ υπενθυμίζουμε ότι θέλουμε το ποσοστό των συνδέσεων από αυτές που καταγράφηκαν στο προηγούμενο. Καταλαβαίνουμε ότι ζητάμε την τιμή των unique συνδέσεων με βάση την IP διά τον συνολικό αριθμό συνδέσεων. Εδώ θα μπορούσαμε να το πραγματοποιήσουμε με πολλούς τρόπους. Για παράδειγμα θα μπορούσαμε να προσθέσουμε ένα property IPAddress στο προηγούμενο event και όταν έχουμε το αποτέλεσμα να βρούμε με βάση την IP σε κάθε event που καταγράφηκε να υπολογίζουμε την ζητούμενη τιμή. Επειδή προσπαθούμε να πραγματοποιήσουμε την λειτουργικότητα χρησιμοποιώντας αυτά που προσφέρει το Esper επιλέγουμε να δημιουργήσουμε ένα event παρόμοιο με το παραπάνω το οποίο θα έχει τα ίδια properties δηλαδή sessionID και port καθώς επίσης και ένα property IPaddress ώστε να ξεχωρίζουμε τα events με βάση τις IP(host machines). Το event αυτό θα δημιουργείται κάθε φορά που δημιουργείται και το παραπάνω. Στην συνέχεια όταν θα έχουμε τα αποτελέσματα του query που θα τρέξουμε θα μπορούμε να υπολογίζουμε την τιμή αυτή με βάση αυτό το event αλλά και το προηγούμενο. Την πληροφορία που θέλουμε θα την έχουμε με την aggregation function count.Ονομάζουμε το γεγονός αυτό **DiffHostRateEvent.**

5.4.2 Δημιουργία Query

Εδώ θα περιγράψουμε τον τρόπο με τον οποίο θα συσσωρεύουμε την πληροφορία για κάθε διεργασία που παρακολουθούμε. Με βάση τα διαθέσιμα features και τα simple events streams όπως τα ορίσαμε στην προηγούμενη ενότητα μπορούμε να συμπεράνουμε τί πληροφορία χρειαζόμαστε. Για το πως θα την αποκτησουμε το Esper**[128]** προσφέρει πολλούς τρόπους. Όπως αναφέραμε και προηγουμένως προσπαθούμε να αξιοποιήσουμε αυτά που προσφέρει το Esper Framework και να κάνουμε με χρήση άλλων μηχανισμών όσο λιγότερα γίνεται αλλά θα περιγράψουμε εναλλακτικούς τρόπους για το πως μπορεί αυτό να γίνει. Σκοπός της εργασίας δεν είναι να απαριθμησει και να περιγράψει αναλυτικά όλη την λειτουργικότητα του framework. Συνεπώς θα κάνουμε μία σύντομη περιγραφή κάποιων δυνατοτήτων γενικά και στην συνέχεια θα περιγράψουμε του πιο προφανής τρόπους να πετύχουμε την λειτουργικότητα ώστε να αιτιολογήσουμε το γιατί διαλέξαμε τον τελικό.

Το Esper αποτελείται από τρία βασικά μέρη, αυτά είναι ο compiler το runtime και η γλώσσα EPL. Ο compiler**[128]** κεφάλαιο 15, στόχο έχει να παράγει byte code από τα queries που δημιουργήσαμε. Το runtime**[128]** κεφάλαιο 16, σκοπό έχει να παρέχει τους τρόπους για να εκτελούνται τα queries. Δεν θα πούμε περισσότερα για το πως λειτουργούν αυτά τα μέρη, αντί για αυτό θα εστιάσουμε στο τρίτο μέρος, δηλαδή την EPL**[128]** κεφάλαιο 5-14. Η EPL είναι μία επέκταση των γλωσσών για Queries σε βάσεις δεδομένων. Η βασική αρχή είναι ότι οι βάσεις δεδομένων με γεγονότα από το παρελθόν αντικαθίστανται από Ροές Δεδομένων πραγματικού χρόνου. Η EPL αποτελεί επέκταση αυτών των γλωσσών παρέχοντας επιπλέον λειτουργικότητα η οποία είναι πολύ χρήσιμη για το σενάριο χρήσης όπου τα δεδομένα είναι διαθέσιμα και πρέπει να επεξεργαστούν σε πραγματικό χρόνο. Επίσης η EPL είναι φτιαγμένη ώστε να ικανοποιεί τις απαιτήσεις που πρέπει να ικανοποιεί ένα CEP Engine όπως ορίστηκαν σε προηγούμενη ενότητα. Αυτό σημαίνει ότι με βάση τα queries που έχουν δηλωθεί είναι σε θέση να πραγματοποιεί βελτιστοποιήσεις**[128]** κεφάλαιο 24, για την διαχείριση της πληροφορίας που απαιτείται ώστε να εκτελεστεί το query καθώς και στον τρόπο που οργανώνει αυτά τα δεδομένα. Επίσης παρέχει πολύπλοκη λειτουργικότητα ώστε να μπορούν να υλοποιηθούν διάφορα σενάρια.

Στην συνέχεια αναφέρουμε βασική λειτουργικότητα.

- **Ανάκτηση Αποτελεσμάτων(Results retrieval).**

  Για να πάρουμε αποτελέσματα από ένα query υπάρχουν τρεις τρόποι**[128]** κεφάλαιο 16.5 . Ο πρώτος είναι οι subscribers. Εδώ συσχετιζουμε ένα statement(query) με μία κλάση η οποία υλοποιεί μία μέθοδο update με αριθμό και τύπο παραμέτρων ίδιο με αυτόν που επιστρέφει το statement(strongly typed). Ο δεύτερος τρόπος είναι να υλοποιήσουμε ένα listener. Σε αυτή την περίπτωση τα αποτελέσματα επιστρέφονται μέσα σε αντικείμενα EventBeans. Είναι παρόμοια μέθοδος με τους subscribers αλλά πιο αργή διότι οι τιμές που επιστρέφονται χρησιμοποιούνται για να δημιουργηθούν νέα αντικείμενα. Τρίτον υπάρχει το PULL API το οποίο προσφέρει την δυνατότητα να ζητάμε από τα statements διαθέσιμη πληροφορία όποτε την χρειαζόμαστε και όχι όποτε πραγματοποιείται μία αλληλουχία δεδομένων.

- **EPL Clauses**

  Η EPL**[128]** κεφάλαιο 5, προσφέρει όλα τα clauses που προσφέρουν και οι γλώσσες τύπου SQL και στις περισσότερες περιπτώσεις η λειτουργία είναι ίδια. Αυτό περιλαμβάνει τα clauses select,from,where,group by, having,limit,order by. Επίσης υποστηρίζει το clause output το οποίο χρησιμοποιείται για να ελέγξουμε τον τρόπο και τις συνθήκες κάτω από τις οποίες παράγεται περιεχόμενο. Επίσης υποστηρίζει το clause insert to. Αυτό το clause χρησιμεύει στο να κάνει διαθέσιμα τα αποτελέσματα ενός query σε άλλα queries ή να

δημιουργήσει state εισάγοντας αποτελέσματα σε Named Windows ή Tables. Περισσότερα για αυτές τις έννοιες στην συνέχεια.

- **Named Windows and Tables**

  Πρόκειται για δομές δεδομένων που παρέχει το esper**[128]** κεφάλαιο 6, όπου μπορούμε να διατηρούμε state. Τα Named Windows και τα tables είναι παρόμοια. Αποθηκεύουν και έχουν διαθέσιμα για κάθε άλλο statement δεδομένα. Η κύριες διαφορές είναι ότι τα Named Windows αποθηκεύουν immutable δεδομένα ενώ τα tables αποθηκεύουν δεδομένα τα οποία μπορούν να αλλάξουν συμπεριλαμβανομένων δεδομένων aggregation. Επίσης τα δεδομένα στα tables μπορούν να έχουν και primary key(s) με βάση τα οποία οργανώνονται.

- **Context**

  Αυτή η λειτουργικότητα**[128]** κεφάλαιο 4, υπάρχει για να διευκολύνει κάποια use cases. Ο σκοπός είναι όταν έχουμε κάποια επεξεργασία η οποία πρέπει να γίνεται ανα οντότητα να έχουμε την δυνατότητα να την υλοποιούμε με απλό τρόπο. Για παράδειγμα μπορεί να θέλουμε να συλλέγουμε στοιχεία, όπως aggregations ανα πελάτη για μία τράπεζα ή ανα διεργασία στην δική μας περίπτωση. Τα στιγμιότυπα που βρίσκονται σε λειτουργία ονομάζονται context partitions. Η EPL παρέχει πολλά είδη contexts και μπορούμε να τα κατηγοριοποιήσουμε με βάση τον τρόπο που τα χειρίζεται το Esper runtime. Αυτά είναι τα key segmented contexts, hash segmented context, overlapping contexts, non-overlapping contexts και category segmented context.

- **Subqueries**

  Η EPL**[128]** κεφάλαιο 5.11, προσφέρει την δυνατότητα ενθυλακωμένων(nested) queries μέσα σε άλλα queries.

- **Joins**

  Η EPL**[128]** κεφάλαιο 5.12 προσφέρει την δυνατότητα που δίνουν όλες οι γλώσσες τύπου SQL να συνδυάζονται διάφορα streams από διαφορετικές πηγές και να εκτελούνται πολύπλοκα queries. Υποστηρίζει τα γνωστά inner joins, outer joins,full outer joins,left join και right join. Επίσης σημαντικό ρόλο παίζει και η εντολή unidirectional όπου αυτό λέει στο runtime να κάνει evaluate το join μόλις έρθει ένα συγκεκριμένο γεγονός από κάποιο simple event. Αυτές οι λειτουργίες επεκτείνονται και τροποποιούνται με χρήση του output clause όπως θα δούμε και στην συνέχεια.

- **Aggregation Functions**

  Η EPL**[128]** κεφάλαιο 13, προσφέρει όλες τις απλές και γνωστές μεθόδους συνάθροισης όπως η count,sum,avedev,max,stdev max ever και άλλα.

- **On Demand Fire and Forget Queries**

Πρόκειται για lightweight μηχανισμό του Esper**[128]** κεφάλαιο 16.7, που προσφέρει την δυνατότητα να εκτελούνται οποιαδήποτε queries όπως ακριβώς κάθε statement που δημιουργείται. Είναι lightweight διότι δεν αφήνει κανένα ίχνος στο runtime και δεν χρειάζεται να δημιουργηθεί statement για αυτό το query. Το πρώτο βήμα είναι να κάνουμε compile το query. Το δεύτερο βήμα να καλέσουμε το EPFireAndForgetService από το runtime. Τέλος εκτελούμε το query και έχουμε τα αποτελέσματα αν επιστρέφει αποτελέσματα. Η μέθοδος αυτή έχει επιλογές και για query που πρόκειται να εκτελεστεί πολλές φορές αλλά και για την περίπτωση που θα εκτεστεί πολλές φορές αλλά με άλλες παραμέτρους.

- **Patterns**

  Τα patterns στην EPL**[128]** κεφάλαιο 7, συμπεριφέρονται παρόμοια με τα απλά statements για τα οποία έχουμε αναφερθεί έως τώρα. Όταν ένα σύνολο από συνθήκες ικανοποιηθούν εκτελείται μία ενέργεια. Η διαφορά έγκειται στους διαφορετικούς operators που χρησιμοποιούνται σε αντιθεση με τα απλά queries. Πέρα από τα event streams εδώ μπορούμε να βασιζόμαστε και σε χρονικές συνθήκες αλλά και σε ροές γεγονότων που έχουν χρονική συσχέτιση ή ακολουθιακή συσχέτιση. Οι βασικοί operators είναι guard postfix, unary, repeat, and, or, not, followed by.

- **Match Recognize**

  Πρόκειται για παρόμοια μέθοδο με την προηγούμενη αλλά εδώ**[128]** κεφάλαιο 8, χρησιμοποιείται η γνωστή σύνταξη των regular expressions. Ο περιορισμός αυτής της περίπτωσης είναι ότι μπορεί να χρησιμοποιηθεί με ένα μόνο είδος event σε αντίθεση με την προηγούμενη μέθοδο εκτός αν μιλάμε για variant streams. Τα variant events είναι events τα οποία δέχονται είτε κάποια συγκεκριμένα αλλά διαφορετικά events ή ακόμα και όλα τα events.

- **Data Windows**

  Το Esper**[128]** κεφάλαιο 14, προσφέρει την λειτουργικότητα των data windows. Αυτό σημαίνει ότι μπορούμε να ρυθμίσουμε το Esper να χρησιμοποιεί εν δυνάμη υποσύνολα των συνολικών events από κάθε event stream. Αυτό δεν βρίσκει εφαρμογή σε tables. Τα συνηθέστερα είναι το time window που περιορίζει το runtime σε events που ήρθαν σε ένα χρονικό παράθυρο και το length window που κρατάει ένα υποσύνολο events άσχετα με τον χρόνο που αυτά έφτασαν.

- **Event Filters**

  Εδώ μπορούμε να κρατήσουμε events με βάση μία συνθήκη από κάποιο property. Μοιάζει σε λειτουργικότητα με συνθήκες στο where clause. Οι συνθήκες τοποθετούνται μέσα σε παρενθέσεις όταν δηλώνουμε το Event Stream.

- **Διάφορα**

Η EPL επίσης υποστηρίζει την χρήση στα statements γνωστών μεθόδων που είναι διαθέσιμες στην JAVA καθώς επίσης και custom μεθόδων ή μεθόδων που έχουν οριστεί κατα την δημιουργία των events. Επίσης υποστηρίζεται η χρήση ταυτόχρονα με τα event streams και πληροφορίας η οποία αφορά το παρελθόν και είναι αποθηκευμένη σε σχεσιακές βάσεις δεδομένων. Επίσης υποστηρίζει και την χρήση μεταβλητών και σταθερών που έχουν οριστεί μέσα στα statements.

Πιθανοί τρόποι υλοποίησης

Εδώ παρουσιάζουμε πιθανούς τρόπους υλοποίησης και δικαιολογούμε αυτόν που επιλέξαμε αλλά και τους λόγους που αναφέραμε τον κάθε τρόπο.

1.Variant Streams και δομές δεδομένων JAVA εκτός Esper

Αυτή η μέθοδος βασίζεται και χρησιμοποιεί ελάχιστη λειτουργικότητα από αυτή που προσφέρει το Esper. Εδώ απλά δημιουργούμε ένα variant stream που δέχεται κάθε τύπο δεδομένων(simple events) και επιλέγει όλα τα properties του event(τελεστής *). Συσχετιζουμε το μοναδικό αυτό statement με έναν listener. Δεν το συσχετίζουμε με subscriber διότι είναι strongly typed οι παράμετροι στην μέθοδο update του subscriber και μπορεί να περιμένει μόνο τιμές συγκεκριμένου τύπου. Όλη η λειτουργικότητα της εφαρμογής βρίσκεται μέσα στον listener. Το πρώτο βήμα είναι να αναγνωρίσουμε το τί είδους event έχουμε και με χρήση κάποιας δομής δεδομένων που προσφέρει η JAVA όπως ένα HashMap και με βάση το sessionID του κάθε event θα προσθέτουμε το αντίστοιχο field/property στην δομή δεδομένων. Ειδικές περιπτώσεις είναι το InitEvent και TerminationEvent. Στην πρώτη περίπτωση θα δημιουργείται μία καταχώρηση στην δομή δεδομένων καθώς και θα αρχικοποιούνται στο μηδέν όλα τα fields τα οποία αφορούν γεγονότα τα οποία μπορεί να μην εμφανιστούν κατα την διάρκεια που μία διεργασία παρακολουθείται όπως τα γεγονότα BytesSendEvent και BytesReceiveEvent. Στην δεύτερη περίπτωση όταν έρθει ένα TerminationEvent η λειτουργικότητα μέσα στον listener θα διαβάζει όλα τα πεδία από την δομή δεδομένων και θα τα προωθεί στο μοντέλο για το inference και θα διαγράφει την αντίστοιχη καταχώρηση. Ξεχωριστή περίπτωση αποτελεί το πεδίο duration το οποίο θα πρέπει να υπολογίζεται όταν φτάνει το TerminationEvent και να προστίθεται στα άλλα.Για να γίνει αυτό η δομή δεδομένων πρέπει να έχει ένα επιπλέον property για το timestamp του InitEvent με το αντίστοιχο sessionID. Επίσης αφού συγκεντρωθεί όλη η πληροφορία στην περίπτωση του TerminationEvent θα πρέπει να διαγράφεται το entry με το συγκεκριμένο sessionID από την δομή δεδομένων.

Η μέθοδος αυτή χρησιμοποιεί ελάχιστη λειτουργικότητα από αυτή που προσφέρει το Esper και δεδομένου ότι το σενάριο που έχουμε είναι σχετικά απλό σε σχέση με τα σενάρια που μπορεί και έχει σχεδιαστεί να αντιμετωπίζει το Esper και έχοντας έναν σχετικά μικρό αριθμό από simple events

αποτελεί μία σύντομη λύση. Επίσης έχοντας μόνο ένα statement και έναν listener ενεργούς μειώνουμε τον φόρτο που αντιμετωπίζει το runtime. Εκτός από αυτά τα πιθανά πλεονεκτήματα η λύση αυτή δεν έχει βέλτιστη δομή από την άποψη ευανάγνωστου και σωστά δομημένου κώδικα καθώς όλες οι ενέργειες που πραγματοποιούνται είναι συγκεντρωμένες μαζί. Περιορισμό εδώ αποτελούν τα δύο τελευταία features όπως ορίστηκαν. Σε αυτές τις περιπτώσεις θέλουμε τα γεγονότα στα δύο τελευταία δευτερόλεπτα. Η περίπτωση να κρατάμε πληροφορία για αυτά τα δύο σε δομές δεδομένων εκτός Esper δεν είναι καλή επιλογή, αντί για αυτο θα μπορούσαμε να περιορίσουμε το variant stream σε όλα τα γεγονότα εκτός από αυτά τα δύο και να φτιάξουμε ξεχωριστά statements για τις δύο περιπτώσεις χρησιμοποιώντας το aggregation function sum με time windows #time(2 sec). Συσχετίζοντας τα καινούρια statements με δύο καινούργιους subscribers θα μπορούσαμε να εισάγουμε στην δομή δεδομένων την κατάλληλη πληροφορία. Με αυτή την επιλογή χάνουμε τις thread safe λειτουργίες του Esper και πρέπει να δημιουργήσουμε δικές μας λύσεις όπως locks.

2. Variant Streams και δομές δεδομένων του Esper

Εδώ το πρώτο μέρος είναι ίδιο με την προηγούμενη μέθοδο. Η διαφορά εδώ είναι ότι χρησιμοποιούμε δομές δεδομένων που προσφέρει το Esper. Οι επιλογές που έχουμε είναι δύο, τα NamedWindows και τα Tables. Δεδομένου του σεναρίου χρήσης δηλαδή ύπαρξη sessions και aggregation πληροφορίας όπως count και sum η κατάλληλη επιλογή είναι τα tables χωρίς αυτό να σημαίνει ότι δεν θα μπορούσαν να χρησιμοποιηθούν και Named Windows. Όπως και στην προηγούμενη περίπτωση αφού αναγνωρίσουμε το είδος του γεγονότος που έχουμε θα εισάγουμε δεδομένα σε κάποιο column του πίνακα. Βασικό ρόλο για αυτήν την περίπτωση της υλοποίησης είναι τα on demand fire and forget queries όπως περιγράφηκαν προηγουμένως.

Βήμα1

Το πρώτο βήμα για αυτήν την περίπτωση της υλοποίησης είναι να δημιουργήσουμε ένα table με τα κατάλληλα columns σύμφωνα με την πληροφορία από τα events που θέλουμε. Όπως αναφέραμε στην ενότητα που ορίσαμε τα simple events εξηγήσαμε τι τύπου είναι το κάθε πεδίο που θέλουμε οπότε πρίν αρχίσει οποιαδήποτε επεξεργασια και με το statement create table δημιουργούμε ένα table. Το table αυτό θα αποτελείται από δέκα πεδία. Το πρώτο θα είναι το sessionID το οποίο θα είναι και το μοναδικό primary key. Τα υπόλοιπα πεδία θα είναι αυτά που προκύπτουν από τα simple events όπως τα ορίσαμε. Το τελευταίο πεδίο θα είναι το timestamp από το InitEvent καθώς θα το χρειαστούμε για να υπολογίσουμε το duration. Εδώ πρέπει να επισημάνουμε ότι όταν δημιουργήσουμε τον πίνακα δεν μπορούμε να ορίσουμε σαν types στα columns aggregation functions όπως count και sum καθώς δεν υποστηρίζονται τα on-demand fire and forget queries που θα χρησιμοποιήσουμε με αυτούς τους τύπους. Για τα συνηθισμένα statements αποτελούν την ιδανική λύση. Επίσης δεν μπορούμε να χρησιμοποιήσουμε keys για τον πίνακα. Αυτό συμβαίνει επειδή τα

fire and forget queries δεν βασίζονται σε κάποιο event που φτάνει στο runtime και δεν έχουμε διαθέσιμο το sessionID του οποίου το row θέλουμε να ενημερώσουμε χρησιμοποιώντας κάποιο εισερχόμενο event καθώς στα clauses into table και insert into το from clause είναι απαραίτητο. Τέλος δεν μπορούμε με απόλυτο τρόπο(explicitly) να δώσουμε το primary key για να εισάγουμε στον πίνακα. Αντίθετα μπορούμε να το κάνουμε για να διαβάσουμε από τον πίνακα. Με βάση αυτά δημιουργούμε τον πίνακα χωρίς primary key για αυτή την περίπτωση. Ο πίνακας δημιουργείται με ένα create table clause ακολουθούμενο από τα ονόματα των columns, τον τύπο τους που στην δική μας περίπτωση θα είναι int και boolean καθώς και τον ορισμό του primary key οπού εδώ δεν υπάρχει. Τον πίνακα θα τον κάνουμε index εμέσως με χρήση του sessionID.

Βήμα2[128 pg302]

Όταν φτάσει ένα Event τύπου InitEvent αφού αναγνωριστεί ως τέτοιο, θα εκτελείται μέσα στον κώδικα του Listener ένα on demand query το οποίο με χρήση του clause insert into για την ειδική περίπτωση των fire and forget on demand queries. Εδώ απλα αναφέρουμε το όνομα του πίνακα και με την λέξη κλειδί values και την τιμή του sessionID, του timestamp εκκίνησης καθώς και αρχικές τιμές για τις υπόλοιπες παραμέτρους. Για αυτό το query καλό είναι να το έχουμε prepared και να κάνουμε bind παραμέτρους.

Βήμα3[128 pg303]

Για όλα τα άλλα events εκτός από τα δύο τελευταία όπου έχουμε τον περιορισμό για τα δύο δευτερόλεπτα κάθε φορά που φτάνουν θα ενημερώνουμε την αντίστοιχη παράμετρο χρησιμοποιώντας το update clause(special FAFQ)**[128]** κεφάλαιο 6.10 και σαν index στο where clause το sessionID του event που λάβαμε. Για τα events που πρέπει να αθροίζουν όπως με χρήση count,sum θα χρησιμοποιούμε στο update clause μετά το codeword set “value = value + 1 ” και “value = value + sum”. Και εδω καλό είναι να χρησιμοποιούμε prepared queries και bind παραμέτρους. Επειδή στα on demand fire and forget queries δεν μπορούμε να χρησιμοποιήσουμε time windows το οποίο χρειαζόμαστε για τα δύο τελευταία features πρέπει να βρούμε άλλο τρόπο. Μία λύση είναι να δημιουργήσουμε δύο διαφορετικά statements ένα για κάθε feature. Εδώ θα χρησιμοποιήσουμε ένα συνηθισμένο select from clause όπου θα παίρνουμε το άθροισμα των γεγονότων που μας ενδιαφέρει για τα features χρησιμοποιώντας τα event streams με time window(2 sec) καθώς και το sessionID. Στην συνέχεια θα συσχετίσουμε έναν subscriber με κάθε statement όπου θα καλείται και θα δίνει την τιμή για κάθε αλλαγή στο time window. Με αυτή την τιμή διαθέσιμη μέσα στον subscriber θα εκτελείται ένα fire and forget on demand query ακριβώς όπως και στο Βήμα2.

Βήμα4

Εδώ όταν φτάνει ένα event TerminationEvent θα εκτελούνται δύο queries. Το πρώτο θα διαβάζει το row που μας ενδιαφέρει με βάση ένα sessionID. Αυτό δεν χρειάζεται ειδικά clauses για fire and forget queries όπως τα προηγούμενα. Στο γνωστό select clause μπορούμε να έχουμε στο from

clause ένα table και με χρήση του where clause να πάρουμε το row που μας ενδιαφέρει**[128]** κεφάλαιο 6.3.5. Το δεύτερο query θα διαγράφει το row που μόλις ανακτήσαμε. Για να γίνει αυτό έχουμε πάλι ένα ειδικό clause delete**[128]** κεφάλαιο 6.10.3 για τα fire and forget queries. Η σύνταξη είναι απλή, ορίζουμε το όνομα του πίνακα και στο where clause ορίζουμε το sessionID που μας ενδιαφέρει.

3.On-event XXX με χρήση δομών δεδομένων του Esper

Σε αυτό το σενάριο θα χρησιμοποιήσουμε την λειτουργικότητα που προσφέρει το Esper ως προς τα triggering events**[128]** κεφάλαια 6.4-6.8. Εδώ όταν φτάνει ένα γεγονός εκτελείται μία ενέργεια η οποία στην περίπτωση μας θα είναι μία δημιουργία μιας γραμμής ενός πίνακα, η ενημέρωση μίας τιμής σε μία γραμμή του πίνακα και τέλος η ανάκτηση και διαγραφή μιας γραμμής ενός πίνακα. Σε αυτή την περίπτωση αντίθετα με την προηγούμενη μπορούμε και θέλουμε να χρησιμοποιήσουμε primary key(s) στον πίνακα δεδομένου ότι εδώ εφόσον θα έχουμε triggering events μπορούμε να αξιοποιήσουμε τα indexes του πίνακα. Το clause on-xxx μπορεί να χρησιμοποιηθεί με τα clauses select, select+delete, update, delete και merge. Επίσης θα χρησιμοποιήσουμε το clause insert to που αφορά tables. Θα περιγράψουμε με λεπτομέρεια τις περισσότερες από αυτές στην συνέχεια. Και αυτή η μέθοδος που περιγράφουμε εδώ μπορεί να εφαρμοστεί με δομές δεδομένων της JAVA αντί του Esper με κάποιες μετατροπές.

Βήμα1

Το πρώτο βήμα όπως και στην προηγούμενη περίπτωση αφορά την δημιουργία του πίνακα.

Βήμα2

Στο δεύτερο βήμα έχουμε την περίπτωση του event InitEvent. Εδώ χρησιμοποιώντας το clause on merge**[128]** κεφάλαιο 6.8.1, θα δημιουργήσουμε ένα row με βάση το sessionID από το InitEvent το οποίο θα είναι και το μοναδικό primary key. Το statement αν δεν δίνει τιμές στα columns τα οποία θα έχουν όλα την τιμή null. Αυτό μας περιορίζει στο να χρησιμοποιήσουμε το keyword set ώστε να ενημερώνουμε τις τιμές aggregation οπότε θέλουμε να δώσουμε αρχικές τιμές στα columns του row αυτού στα παρακάτω βήματα. Για να το κάνουμε αυτό στο clause on merge στο clause select εκτός από primary key βάζουμε και σταθερές(τιμή 0 και boolean false) σε όλα τα columns χρησιμοποιώντας το keyword as και τα ονόματα των στηλών που δώσαμε στον πίνακα. Αυτό το statement θα εκτελείται κάθε φορά που έρχεται ένα InitEvent. Εδώ σχετικά με τις αρχικές τιμές για τα boolean γεγονότα τις βάζουμε όλες false με την λογική ότι αν συμβούν τα γεγονότα έτσι όπως τα ορίσαμε ή θα αλλάξουν ή θα μείνουν ίδια. Και στις δύο περιπτώσεις καλύπτουμε σωστά τις περιπτώσεις. Δεν συσχετίζουμε κανένα listener/subscriber με αυτό το statement διότι δεν μας ενδιαφέρουν το νέο row που δημιουργείται.

Βήμα3

Στο βήμα αυτό θα χρησιμοποιήσουμε το clause on update**[128]** κεφάλαιο 6.6. Αυτό το clause το χρησιμοποιούμε για να ενημερώσουμε τα columns για κάθε row που έχει δημιουργηθεί από κάθε InitEvent στο προηγούμενο. Για κάθε event όπως τα ορίσαμε σε προηγούμενη ενότητα εκτός από τα InitEvent και TerminationEvent θα δημιουργήσουμε ένα statement on-update. Στην συνέχεια όταν έρχεται ένα γεγονός θα το ονομάζουμε καθώς και τον πίνακα με χρήση του keyword as. Θα κάνουμε update το row εκείνο του πίνακα που αντιστοιχεί στο γεγονός που μόλις λάβαμε και με χρήση του keyword set θα θέτουμε την νέα τιμή με βάση την παλιά τιμή του πίνακα όπου αυτό χρειάζεται. Στην συνέχεια το statement ολοκληρώνεται με συσχέτιση του sessionID του εισερχόμενου γεγονότος με το primary key του πίνακα. Αυτό το πετυχαίνουμε με χρήση του clause where όπου με βάση τα ονόματα που δώσαμε σε πίνακα και γεγονός ορίζουμε να εκτελείται η ενέργεια για τα sessionIDs που ταιριάζουν. Όπως και στην προηγούμενη ενότητα τα δύο τελευταία features που έχουν περιορισμό time windows δεν μπορούμε να τα αντιμετωπίσουμε όπως τα υπόλοιπα. Σε αυτή την περίπτωση όμως έχουμε table με primary key οπότε μπορούμε να το εκμεταλλευτούμε. Με βάση αυτά για τα δύο features θα χρησιμοποιήσουμε το clause into table**[128]** κεφάλαιο 6.3.2. Εδώ απλά δίνουμε το όνομα του πίνακα, με ένα select query από το Event με το time window 2sec έχουμε την πληροφορία που χρειαζόμαστε και στο group by clause δίνουμε το primary key, δηλαδή το sessionD. Σε αυτά τα statements που ορίσαμε εδώ δεν τα συσχετίζουμε με listeners/subscribers καθώς δεν μας ενδιαφέρει τι αποτελέσματα παράγουν. Απλά ενημερώνουν τιμές.

Βήμα4

Στο τελευταίο στάδιο θα χρησιμοποιήσουμε το clause on-select+delete**[128]** κεφάλαιο 6.5. Το Esper προσφέρει την επιλογή αυτή οπότε δεν χρειάζεται να εκτελέσουμε δύο statements. Εδώ όταν έρχεται ένα TerminationEvent θα ονομάζουμε και πάλι το event με χρήση του keyword as και στο from clause θα έχουμε το όνομα του table το οποίο θα ονομάζουμε επίσης. Τέλος στο where clause θα έχουμε την συνθήκη ότι το sessionID του TerminationEvent πρέπει να είναι ίδιο με αυτό του row του table. Στην περίπτωση αυτή θα πρέπει να συσχετίσουμε το statement αυτό με έναν subscriber ή έναν listener. Όπως αναφέραμε προηγουμένως ο subscriber έχει καλύτερη επίδοση σε σχέση με το listener και επειδή κατα την δημιουργία κάθε row δώσαμε αρχικές τιμές σε κάθε τιμή ξέρουμε ότι τίποτα δεν είναι null άρα μπορούμε να χρησιμοποιήσουμε subscriber.

4.Outer Joins

Στην μέθοδο αυτή δεν θα χρησιμοποιήσουμε δομές δεδομένων για να κρατάμε state ανάμεσα στα statements. Το Esper θα κρατάει από μόνο του όλες τις απαραίτητες τιμές είτε απλές είτε aggregation. Εδώ αντίθετα με τις προηγούμενες περιπτώσεις θα χρησιμοποιήσουμε aggregation functions και το Esper θα φροντίζει να κρατάει τις κατάλληλες τιμές και μόνο την απαραίτητη πληροφορία για να διατηρείται αυτή η πληροφορία. Όταν λέμε απαραίτητη εννοούμε ότι για

παράδειγμα αν κρατάμε απλά ένα count γεγονότων το runtime δεν θα κρατάει κάθε γεγονός που έρχεται με όλα τα properties αλλά μόνο μία τιμή, το πόσα γεγονότα δέχθηκε. Εδώ θα χρησιμοποιήσουμε ένα μόνο statement το οποίο θα κάνει join**[128]** κεφάλαιο 5.12 εισερχόμενα streams. Στο join θα αναφέρονται όλα τα streams όπως τα ορίσαμε. Διαλέγουμε outer join διότι κάποια events όπως το Source bytes και destination Bytes μπορεί να μην εμφανιστούν ποτέ και θέλουμε το statement να εκτελεστεί ακόμα και αν λείπουν κάποια events από συγκεκριμένα streams. Αντίθετα το outer join θα εκτελεστεί. Σε αυτήν την περίπτωση χρησιμοποιούμε στο έπακρο τις δυνατότητες του Esper αν και το query που προκύπτει είναι μεγάλο. Επίσης θέλουμε το outer join να εκτελεστεί όταν φτάσει ένα TerminationEvent οπότε χρησιμοποιούμε το keyword unidirectional για αυτό το Event Stream. Για όλα τα άλλα Event Streams πρέπει να έχουν ένα data window και να συσχετίσουμε το κάθε event με όλα τα άλλα events με βάση το sessionID. Εδώ πρέπει να σημειωθεί ότι το statement παράγει αποτελέσματα άσχετα με τους συσχετισμούς στο where clause οπότε κατελάχιστο το statement θα δώσει αποτελέσματα όταν έρθει ένα TerminationEvent. Έτσι λειτουργεί το unidirectional outer join. Ακολουθούν αναλυτικά όλα τα βήματα για αυτήν την λύση. Πριν ξεκινήσουμε και για να μην επαναλαμβανόμαστε να σημειωθεί ότι όλα τα Event streams θα ονομαστούν με το keyword as. Επίσης με εξαίρεση το unidirectional event όλα τα event streams πρέπει να ορίζονται με χρήση data window.

Βήμα1

Για το InitEvent θέλουμε να ανακτήσουμε το timestamp από το query οπότε το βάζουμε στο clause select. Για time window πρέπει να χρησιμοποιήσουμε το #keepall.

Βήμα2

Για όλα τα άλλα event streams εκτός των TerminationEvent και των δύο τελευταίων όπου χρειάζονται time window 2sec. Θα επιλέξουμε στο select clause όλα τα properties που μας ενδιαφέρουν,είτε είναι απλές τιμές είτε aggregation, και θα θέσουμε time window #keepall.

Βήμα3

Για τα δύο τελευταία features θα χρησιμοποιήσουμε το aggregation function count και time window #time(2 sec).

Βήμα4

Για το Termination Event θα κρατήσουμε στο select clause το timestamp και θα το θέσουμε ως unidirectional. Αυτό σημαίνει ότι θα είναι το simple event που θα πυροδοτεί το query. Σε event stream που ορίζεται ως unidirectional δεν μπορούμε και δεν έχει νόημα να χρησιμοποιήσουμε data window.

Βήμα5

Για να ολοκληρώσουμε την συσχέτιση των γεγονότων στο where clause θα θέσουμε τον περιορισμό τα sessionIDs όλων των events να είναι ίδια με το sessionID του TerminationEvent που πυροδότησε το query.

Βήμα6

Τέλος συσχετίζουμε με το statement ένα listener και μέσα στον listener ανακτούμε όλα τα δεδομένα και τα προωθούμε στο μοντέλο. Χρησιμοποιούμε listener διότι μπορεί να έχουμε τιμές null και ο subscriber θα έδινε exception. Το τελευταίο μπορεί να αντιμετωπιστεί με χρήση του function coalesce όπως θα δούμε στην επόμενη μέθοδο, οπότε υπάρχει και αυτό σαν επιλογή.

5.Context και output clause

Αυτή είναι η τελευταία μέθοδος που θα παρουσιάσουμε καθώς επίσης είναι και αυτή που επιλέγουμε τελικά να υλοποιήσουμε. Εδώ εκμεταλλευόμαστε στο έπακρο την λειτουργικότητα που παρέχει το Esper και για αυτό τον λόγο θεωρητικά είναι και η πιο αποδοτική. Τα κύρια μέρη που θα χρησιμοποιήσουμε είναι δύο, το context**[128]** κεφάλαιο 4 και το output clause**[128]** κεφάλαιο 5.7. Το context είναι μία έννοια στο esper η οποία για την δική μας περίπτωση δίνει την δυνατότητα να εκτελούμε statements για διαφορετικά στιγμιότυπα δεδομένων. Εδώ θα χρησιμοποιήσουμε τα key segmented contexts**[128]** κεφάλαιο 4.2.2 . Πιο συγκεκριμένα για κάθε διαφορετικό InitEvent που θα δέχεται το runtime θα ξεκινάει ένα νέο partition δηλαδή ένα στιγμιότυπο του statement που θα δημιουργήσουμε το οποίο θα κρατάει state ξεχωριστά για κάθε session. Το δεύτερο βασικό μέρος είναι το output clause. Το clause αυτό μεταξύ άλλων μας δίνει την δυνατότητα να εκτελούμε ένα statement και συνεπώς να παίρνουμε αποτελέσματα από αυτό το statement μόνο όταν ικανοποιείται μία συνθήκη όπως όταν λαμβάνεται ένας συγκεκριμένος αριθμός events, όταν περάσει ένα συγκεκριμένο χρονικό όριο και άλλα. Εδώ θα χρησιμοποιήσουμε την λειτουργικότητα να εκτελείται το statement όταν τερματίζεται ένα partition. Επίσης θα διαλέξουμε με το select clause την πληροφορία που μας ενδιαφέρει από τα Events. Στην συνέχεια παρουσιάζουμε αναλυτικά τα βήματα για την υλοποίηση και δίνουμε και τον κώδικα ώστε να είναι πιο κατανοητά. Για τον κώδικα του select clause θα αναφέρουμε το Event Stream που χρησιμοποιούμε καθώς και το όνομα που του δίνουμε καθώς και το πεδίο που αντιστοιχεί σε αυτό στο select clause. Επίσης να σημειωθεί ότι παρόλο που χρησιμοποιούμε το output when terminated clause το query παραμένει join συνεπώς πρέπει όλα τα Event Streams να συνοδεύονται από data windows. Επίσης να αναφέρουμε ότι το γεγονός που δημιούργησε το κάθε partition είναι διαθέσιμο στο select statement σαν context property με το όνομα context.init, κάτι το οποίο θα χρησιμοποιήσουμε. Στις περισσότερες περιπτώσεις θα χρησιμοποιούμε το #keepall αλλά το Esper δεν θα κρατάει όλα τα events αλλά μόνο πληροφορία aggregation στις περισσότερες περιπτώσεις.

Βήμα1

Το πρώτο βήμα είναι να ορίσουμε το context. Αυτό γίνεται με το clause create context. Το context θα γίνεται partitioned με βάση το sessionID από το InitEvent και θα τερματίζεται από το TerminationEvent με το ίδιο sessionID. Για να συσχετίσουμε το κάθε InitEvent με το αντίστοιχο TerminationEvent θα δώσουμε όνομα στο πρώτο με χρήση του codeword as. Το statement που εκτελείται είναι το ακόλουθο.

```
@public create context ids partition by sessionID from InitEvent as init terminated by TerminationEvent(sessionID=init.sessionID);
```

Βήμα2

Στην συνέχεια θα συσχετίσουμε το context με το query που θα παράγει τα αποτελέσματα. Για να το κάνουμε αυτό θα δώσουμε ένα όνομα στο query ώστε να μπορούμε να αναφερθούμε σε αυτό στην συνέχεια και θα το συσχετίσουμε με το context που δημιουργήσαμε στο προηγούμενο.

```
@public @name('idsStatement') context ids select
```

Στην συνέχεια θα επιλέξουμε τα πεδία που μας ενδιαφέρουν. Θα τα αναφέρουμε με την ίδια σειρά που ορίσαμε τα Simple Events σε προηγούμενη ενότητα.

Βήμα3

Εδώ θα πάρουμε το sessionID από το γεγονός που ξεκίνησε το partition. Αυτό πρέπει να το κάνουμε διότι σε κάθε statement πρέπει να υπάρχει τουλάχιστον ένα stream από αυτά που χρησιμοποιήθηκαν για να δημιουργηθεί το partition. Το μόνο τέτοιο γεγονός είναι το IniEvent οπότε πρέπει να το βάλουμε στο query.

Για το from clause έχουμε το παρακάτω.

```
InitEvent(sessionID = context.init.sessionID)#keepall as IE
```

Για το select clause έχουμε το παρακάτω.

```
IE.sessionID as session
```

Βήμα4

Για το event BytesSendEvent θέλουμε το άθροισμα όλων των properties από τα συγκεκριμένα events όταν αυτά αφορούν το sessionID του InitEvent που ξεκίνησε το συγκεκριμένο partition οπότε θα χρησιμοποιήσουμε το context property που αφορά το InitEvent και με χρήση φίλτρων θα κρατήσουμε μόνο τα BytesSendEvent events με το ίδιο sessionID. Επίσης θέλουμε το άθροισμα των properties bytessent οπότε θα χρησιμοποιήσουμε το aggregation function sum. Επειδή αυτό το event μπορεί να μην εμφανιστεί ποτέ και επειδή θα χρησιμοποιήσουμε subscriber για να πάρουμε τα αποτελέσματα χρησιμοποιούμε το function coalesce. Αυτό το function επιστρέφει την πρώτη τιμή που δεν είναι null. Έτσι αν δεν εμφανιστεί ποτέ το συγκεκριμένο event θα επιστρέψει την τιμή μηδέν.

Για το from clause έχουμε το παρακάτω.

```
BytesSendEvent(sessionID = context.init.sessionID)#keepall as BSEE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(sum(coalesce(BSEE.bytessent,0)),0) as BSEsum
```

Βήμα5

Για το event BytesReceiveEvent θα κάνουμε τα ίδια με προηγουμένως. Δίνουμε τον κώδικα παρακάτω για λόγους πληρότητας.

Για το from clause έχουμε το παρακάτω.

```
BytesReceiveEvent(sessionID = context.init.sessionID)#keepall as BREE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(sum(coalesce(BREE.bytesreceived,0)),0) as BREsum
```

Βήμα6

Για το LandEvent θέλουμε το boolean property. Αυτό το event θα είναι πάντα διαθέσιμο.

Για το from clause έχουμε το παρακάτω.

```
LandEvent(sessionID = context.init.sessionID)#keepall as LE
```

Για το select clause έχουμε το παρακάτω.

```
LE.land as L
```

Βήμα7

Για το HotLogInEvent θέλουμε το boolean property. Αυτό το γεγονός είναι πάντα διαθέσιμο. Για το from clause έχουμε το παρακάτω.

```
HotLogInEvent(sessionID = context.init.sessionID)#keepall as HLE
```

Για το select clause έχουμε το παρακάτω.

```
HLE.hotlogin as HL
```

Βήμα8

Για το FailedLogInEvent θέλουμε τον άθροισμα όλων των γεγονότων που αφορούν το συγκεκριμένο sessionID. Για να το πετύχουμε αυτό θα χρησιμοποιήσουμε το aggregation function count. Επίσης επειδή αυτό το γεγονός μπορεί να μην εμφανιστεί ποτέ θα χρησιμοποιήσουμε και εδώ το function coalesce.

Για το from clause έχουμε το παρακάτω.

```
FailedLogInEvent(sessionID = context.init.sessionID)#keepall as FLIE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(count(FLIE.*),0) as FLC
```

Βήμα9

Για το WrongFragmentEvent θέλουμε το άθροισμα όλων των γεγονότων που θα λάβουμε οπότε έχουμε όπως και στο προηγούμενο.

Για το from clause έχουμε το παρακάτω.

```
WrongFragEvent(sessionID = context.init.sessionID)#keepall as WFE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(count(WFE.*),0) as WFC
```

Βήμα10

Για το SuEvent θέλουμε αν εμφανίστηκε ή όχι οπότε μπορούμε να το αντιμετωπίσουμε όπως το προηγούμενο.

Για το from clause έχουμε το παρακάτω.

```
SuEvent(sessionID = context.init.sessionID)#keepall as SUE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(count(SUE.*),0) as SUC
```

Βήμα11

Για το SrvCountEvent θέλουμε το σύνολο των events που έφτασαν και ήταν στο ίδιο port με την τρέχουσα σύνδεση στα τελευταία δύο δευτερόλεπτα. Για να το κάνουμε αυτό συσχετίζουμε τα events αυτά με το event InitEvent που ξεκίνησε το partition και θέτουμε στο φίλτρο να έχουν το ίδιο port με το init event. Επίσης για το data window το θέτουμε να είναι δύο δευτερολέπτων. Χρησιμοποιουμε την coalesce για τους ίδιους λόγους με τα προηγουμενα.

Για το from clause έχουμε το παρακάτω.

```
SrvCountEvent(port = context.init.port)#time(2 sec) as SRVCE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(count(SRVCE.*),0) as SRVCC
```

Βήμα12

Για το event DiffHostRateEvent θέλουμε τον αριθμό των συνδέσεων που είναι στο ίδιο port με αυτή που αναλύουμε σε αυτό το partition αλλά πραγματοποιήθηκαν σε διαφορετικά hosts. Άρα στα φίλτρα θέλουμε το port ίδιο με της σύνδεσης που αναλύουμε και το IP διαφορετικό.

Για το from clause έχουμε το παρακάτω.

```
DiffHostRateEvent(port = context.init.port and IP != context.init.IP)#time(2 sec) as DHRE
```

Για το select clause έχουμε το παρακάτω.

```
coalesce(count(DHRE.*),0) as DHRC
```

Βήμα13

Τέλος θέλουμε πληροφορία για το duration. Αυτό το κάνουμε σε δύο βήματα. Το πρώτο είναι να πάρουμε το timestamp από το InitEvent που ξεκίνησε το partition.

Στο select clause έχουμε το παρακάτω.

```
context.init.timestamp as INETS
```

Στην συνέχεια θέλουμε το timestamp από το event που τερμάτισε το partition.

Στο from clause έχουμε το παρακάτω.

```
TerminationEvent(sessionID = context.init.sessionID) as TE
```

Στο select clause έχουμε το παρακάτω.

```
TE.timestamp as TETS
```

Τέλος τερματίζουμε το query με το output when terminated clause όπως φαίνεται παρακάτω.

```
String output = "output last when terminated";
```

Βήμα14

Αφού δημιουργήσουμε το query με όλα τα παραπάνω και έχοντας φροντίσει να μην είναι καμία τιμή null το επόμενο βήμα είναι να φτιάξουμε το subscriber. Εδώ απλά δημιουργούμε μία κλάση JAVA η οποία θα υλοποιεί μία μέθοδο με το όνομα update. Η μέθοδος αυτή δεν επιστρέφει μία τιμή και οι παράμετροι της πρέπει να είναι ίδιου τύπου με τις τιμές που επιστρέφει το query και με την ίδια σειρά με την οποία εμφανίζονται. Επίσης ο subscriber στον constructor θα δέχεται και μία παράμετρο από ένα PrintWriter αντικείμενο που θα έχει δημιουργηθεί στο main function του προγράμματος από ένα socket. Αυτό το PrintWriter στην συνέχεια θα χρησιμοποιείται από τον subscriber ώστε κάθε φορά που εκτελείται να στέλνει την πληροφορία στο μοντέλο για επεξεργασία. Από την πληροφορία θα δημιουργείται ένα string. Δίνουμε ενδεικτικά τον κώδικα για την update.

```
public void update(int session, int BSEsum,int BREsum, boolean L, boolean HL, Long FLC, Long WFC, Long SUC, Long SRVCC, Long DHRC, int TETS, int INETS)
```

και στην συνέχεια αφού μετατρέψει όλες τις παραμέτρους σε String χρησιμοποιεί το delimeter χαρακτήρα “/” και δημιουργεί το τελικό string το οποίο στέλνει στο ServerSocket το οποίο θα δημιουργηθεί από τον κώδικα που θα τρέχει το μοντέλο.

Στην συνέχεια έχοντας το query και τον subscriber δείχνουμε τα επιπλέον βήματα που χρειάζονται για να τρέξει αυτό το component.

Βήμα15

Αρχικά θέλουμε ένα instance αντικειμένου configuration για να το χρησιμοποιήσουμε στο runtime και στον compiler.

```
Configuration configuration = new Configuration();
```

Στην συνέχεια θα δηλώσουμε όλα τα event streams που θα χρησιμοποιήσουμε στο αντικείμενο configuration. Δίνουμε ένα για παράδειγμα.

```
configuration.getCommon().addEventType(InitEvent.class);
```

Επίσης δηλώνουμε ότι θέλουμε να ενεργοποιήσουμε την δυνατότητα να χρησιμοποιήσουμε subscriber. Το τελευταίο αφορά τον compiler.

```
configuration.getCompiler().getByteCode().setAllowSubscriber(true);
```

Στην συνέχεια δημιουργούμε ένα αντικείμενο Compiler Arguments από το αντικείμενο configuration.

```
CompilerArguments arguments = new CompilerArguments(configuration);
```

Στην συνέχεια έχουμε ένα instance runtime και ένα instance του compiler.

```
EPRuntime runtime1 = EPRuntimeProvider.getDefaultRuntime(configuration);
```

```
EPCompiler compiler = EPCompilerProvider.getCompiler();
```

Στην συνέχεια κάνουμε compile το query που δημιουργήσαμε προηγουμένως με χρήση των arguments.

```
epCompiled2 = compiler.compile(query, arguments);
```

Στην συνέχεια κάνουμε deploy το compiled query από το προηγούμενο.

```
deployment = runtime1.getDeploymentService().deploy(epCompiled2);
```

Στην συνέχεια θα δημιουργήσουμε ένα αντικείμενο statement από το query που κάναμε compile.

```
EPStatement statement1 = runtime1.getDeploymentService().getStatement(deployment.getDeploymentId(), "mystatement2");
```

Στην συνέχεια δημιουργούμε ένα instance του subscriber. Σαν παράμετρο δίνουμε ένα αντικείμενο PrintWriter που δημιουργήσαμε με βάση το socket που έχει συνδεθεί με το module που τρέχει το μοντέλο και εκεί θα προωθεί την πληροφορία όποτε αυτή προκύπτει.

```
Subscriber sub = new Subscriber(writer);
```

Και συσχετιζουμε το statement με αυτόν.

```
statement1.setSubscriber(sub);
```

Εδώ τελειώνει το setup αυτού του module. Για να ολοκληρωθεί πρέπει να έχουμε ένα while loop το οποίο θα δέχεται είσοδο από το αρχείο που θα γράφονται όλα τα logs και θα διαβάζει κάθε γραμμή όταν αυτή είναι διαθέσιμη. Όταν θα διαβάζει μία γραμμή θα καλεί ένα function το οποίο θα βρίσκει τι είδους event έχει έρθει και θα κάνει extract όλα τα πεδία. Τέλος θα δημιουργεί το κατάλληλο event και θα το στέλνει στο runtime. Εδώ υποθέτουμε ότι το κάθε log εκτός από τα πεδία που ορίσαμε θα έχει πληροφορία για τον τύπο του event ώστε να είναι πιο απλή η υλοποίηση. Το format που θα στέλνεται η πληροφορία από τα logs θα είναι ίδιο με αυτό που χρησιμοποιείται για να τα στείλει στο module που τρέχει το μοντέλο και πραγματοποιείται το inference. Δηλαδή θα έχουμε τις απαραίτητες τιμές με delimiter χαρακτήρα το “/”. Το πρώτο μέρος θα είναι το event type. Ο κώδικας για το function αυτό είναι παίρνει δύο παραμέτρους. Η πρώτη είναι το String που διαβασε από το socket και η άλλη το runtime που δημιουργήσαμε προηγουμένως. Αρχικά κάνει split με βάση

το delimiter “/” και στην συνέχεια με χρήση case και χρησιμοποιώντας το runtime και το event type δημιουργεί και στέλνει το event στο runtime. Δίνουμε για λόγους παρουσίασης τον ορισμό του function και ένα μόνο case.

```
public void insertEvent(String log, EPRuntime runtime) {

String[] event = log.split("/");

switch (event[0]) {

case "InitEvent":
runtime.getEventService().sendEventBean(new InitEvent(Integer.parseInt(event[1]),
Integer.parseInt(event[2]), Integer.parseInt(event[3]), Integer.parseInt(event[4])), "InitEvent");
break;
```

Όλα τα υπόλοιπα events δημιουργούνται και αποστέλλονται έτσι.

Στην συνέχεια δημιουργείται το loop που όταν έχει διαθέσιμο μήνυμα από το socket που ακούει στα logs θα το διαβάζει και θα το προωθεί στην μέθοδο insertEvent που ορίσαμε παραπάνω.

```
while(true) {

message = reader.readLine();
insertEvent(message,runtime1);

}
```

6.Συμπεράσματα

Στις προηγούμενες ενότητες δώσαμε την θεωρία σχετικά με ένα σύστημα IDS και τις απαιτήσεις που πρέπει να ικανοποιεί και παρουσιάσαμε τα εργαλεία που χρειάζονται για να ολοκληρωθεί και να λειτουργεί ένα τέτοιο σύστημα και πραγματοποιήσαμε την υλοποίηση των βασικών modules ενός τέτοιου συστήματος. Σε αυτή την ενότητα θα αναφερθούμε πάλι στο συνολικό pipeline αλλά από την οπτική των αδυναμιών και των ιδιαιτεροτήτων που προκύπτουν. Θα

χωρίσουμε την ενότητα αυτή σε δύο μέρη. Στο πρώτο μέρος θα παρουσιάσουμε θεωρητικά τα πλεονεκτήματα και τα μειονεκτήματα της χρήσης μηχανικής μάθησης στον τομέα της ασφάλειας και θα τονίσουμε κάποια από αυτά συγκεκριμένα για το use case που ασχολούμαστε. Στην συνέχεια θα παρουσιάσουμε πρακτικά προβλήματα σχετικά με το deployment που προκύπτουν και επηρεάζουν την ικανοποίηση των απαιτήσεων για τα IDSs.

6.1 Πλεονεκτήματα και Μειονεκτήματα χρήσης ML στην Ασφάλεια

Τα κύρια πλεονεκτήματα της χρήσης Μηχανικής μάθησης για το πρόβλημα που εξετάζουμε αλλά και γενικά είναι τρία. Το πρώτο είναι η ευελιξία και η απόδοση ενός μοντέλου ML. Τα μοντέλα ML στην πράξη έχουν συγκρίσιμη απόδοση, ή και καλύτερη όσον αφορά το detection αλλά και τα false positives με τις καθιερωμένες μεθόδους όπως οι στατικές υπογραφές. Το πλεονέκτημα είναι το ότι οι υπογραφές είναι πολύ συγκεκριμένα στιγμιότυπα επιθέσεων και αν ο επιτιθέμενος τις γνωρίζει μπορεί εύκολα να παρακάμψει τον έλεγχο αλλάζοντας κατάλληλα την συμπεριφορά του. Αυτό ναι μεν είναι πρόβλημα και για το ML σενάριο αλλά πιο δύσκολο στην πράξη. Με βάση αυτό το κύριο πλεονέκτημα είναι η γενίκευση σε νέες επιθέσεις που δεν έχουν ξανα προκύψει αλλά και σε παραλλαγές ήδη γνωστών επιθέσεων. Το δεύτερο πλεονέκτημα είναι το ότι δεν χρειάζεται επίβλεψη και ανθρώπινο δυναμικό για να δουλεψει σε αντίθεση με την επιπονη και κοστοβόρα διαδικασία όπου ένας τεχνικός αναλύει ημι αυτόματα logs και παράγει χειροκίνητα υπογραφές. Το τρίτο πλεονέκτημα είναι ότι θεωρητικά το ML μοντέλο θα έχει καλύτερη επίδοση όσον αφορά το latency και την απαίτηση για πρόβλεψη σε πραγματικό χρόνο σε σχέση με στατικές υπογραφές καθώς για κάθε νέο δείγμα δεν χρειάζεται να το συγκρίνει με έναν μεγάλο αριθμό στατικών υπογραφών.

Σχετικά με τα μειονεκτήματα θα μπορούσαμε να αναφέρουμε περιπτώσεις που θα μπορούσαν να χαρακτηριστούν ώς τέτοια αλλά δεν αποτελούν με την απόλυτη έννοια προβλήματα που προκύπτουν από την χρήση γενικά Μηχανικής Μάθησης. Συγκεκριμένα μάλλον είναι αδυναμίες ή προκλήσεις που πρέπει να αντιμετωπιστούν παρά μειονεκτήματα. Για τον λόγο αυτό θα αναφέρουμε, αναλύσουμε και θα δώσουμε κατευθύνσεις για αυτά στην επόμενη ενότητα. Για να αναφέρουμε σε αυτήν την ενότητα τρία από τα πιο σημαντικά χαρακτηριστικά που μπορούν να χαρακτηριστούν ως μειονεκτήματα αυτά είναι η έλλειψη datasets που μπορούν να χαρακτηριστούν πλήρη και ως προς το εύρος των περιπτώσεων που περιλαμβάνουν αλλά και ως προς την ακρίβεια των δεδομένων. Το δεύτερο είναι η ανάγκη για συλλογή δεδομένων σε πραγματικό χρόνο από πολλές πηγές. Το τρίτο το overhead που προκαλούν στο σύστημα.

Για την πρώτη περίπτωση αυτό αποτελεί πρόβλημα για όλα τα σενάρια ML και όχι μόνο για το σενάριο χρήσης που αναλύουμε. Το δεύτερο σχετίζεται με αυτό που αναλύουμε. Το τρίτο αφορά γενικά σενάρια ML. Δύο παραδείγματα για το τελευταίο είναι τα μοντέλα που εκτελούνται σε microcontrollers καμερών σε smartphones και ένα σύστημα σαν αυτό που εξετάζουμε το οποίο θα πρέπει να εκτελείται σε μία συσκευή IoT. Θα αναφερθούμε ξανά σε αυτά τα προβλήματα αναλυτικά στην επόμενη ενότητα.

Επίσης πρέπει να αναφέρουμε ότι όλα τα πλεονεκτήματα της χρήσης Μηχανικής Μάθησης στην ανίχνευση επιθέσεων μεταξύ άλλων δέν είναι δεδομένα. Όπως οι αμυνόμενοι χρησιμοποιούν αυτά τα ισχυρά εργαλεία για να πετύχουν το σκοπό τους και οι επιτιθέμενοι**[109,110]** χρησιμοποιούν τα ίδια εργαλεία για τον αντίθετο λόγο. Με βάση αυτά συμπεραίνουμε ότι αν ένα μοντέλο Μηχανικής Μάθησης πετυχαίνει πολύ καλή απόδοση σε ένα περιορισμένου εύρους dataset που δημιουργήθηκε πριν δεκαετίες δεν σημαίνει ότι το πρόβλημα λύθηκε για πάντα. Παραδοσιακά στον τομέα του CyberSecurity οι τεχνικές των επιτιθέμενων και των αμυνόμενων εξελίσσονται παράλληλα.

6.2 Πρακτικά προβλήματα που πρέπει να αντιμετωπιστούν

Όπως αναφέρεται εδώ**[127]** το ML και το cybersecurity μπορούν να συνδυαστούν με δύο τρόπους. Ο πρώτος είναι να χρησιμοποιήσουμε το cyber security ώστε να κάνουμε πιο ανθεκτικά σε λάθος χρήση τα μοντέλα που θέλουμε να προστατέψουμε. Ο δεύτερος είναι να χρησιμοποιήσουμε το ML ώστε να δημιουργήσουμε λύσεις cyber security. Σε αυτή την ενότητα θα αναφερθούμε και στις δύο περιπτώσεις.

Με βάση τα όσα είδαμε σε προηγούμενες ενότητες η έρευνα γύρω από την χρήση μεθόδων Μηχανικής Μάθησης για την υλοποίηση του inference ενός IDS είναι πολλές. Συνολικά βλέπουμε ότι έχει εφαρμοστεί κάθε τεχνολογία από shallow μοντέλα μέχρι deep μοντέλα για την υλοποίηση αυτού του μέρους έχοντας πολύ καλά αποτελέσματα. Παρόλα αυτά χρησιμοποιούνται ακόμα σε production συστήματα τεχνικές όπως υπογραφές που παράγονται από domain experts. Τα ML μοντέλα ναι μεν παράγουν πολύ καλά αποτελέσματα με βάση τα μετρικά αλλά εισάγουν και δικές τους αδυναμίες.

Ανάγκη για ασφάλεια του IDS

Σαν πρώτο σημείο πρέπει να αναφέρουμε ότι το IDS, όποια τεχνολογία και αν χρησιμοποιεί είναι ένα κομμάτι λογισμικού που έρχεται σε επαφή με άλλο λογισμικό ή με κάποιον χρήστη και μπορεί και αυτό να γίνει στόχος επίθεσης όπως έχει συμβεί πολλές φορές με άλλο λογισμικό προστασίας όπως τα antivirus. Με βάση αυτά έχουμε ένα σύνολο από επιθέσεις που μπορούν να

επηρεάσουν ένα οποιοδήποτε λογισμικό. Η πρώτη κατηγορία επιθέσεων έχει στόχο να παραπλανήσει ή να θέσει εκτός λειτουργίας το λογισμικό αυτό και περιλαμβάνει κάθε μέσο που μπορεί να το πετύχει όπως να παράγει πολλή δραστηριότητα από κάποια διεργασία που ελέγχει ή να βρει ένα πρόβλημα το οποίο θα επιτρέψει τον τερματισμό του IDS. Αυτά τα σενάρια πρέπει να αντιμετωπιστούν διότι αφορούν κάποιες απο τις σημαντικότερες απαιτήσεις όπως το fault tolerance. Με βάση αυτό το λογισμικό IDS θα πρέπει να εκτελείται σε περιβάλλον με αυξημένη ασφάλεια. Αυτό μπορεί να επιτευχθεί με κατάλληλο configuration σχετικά με το access control και το authentication ταυτόχρονα με τεχνικές sandboxing. Για παράδειγμα θα μπορούσε να εκτελείται αποκλειστικά σε δικό του VM ή να εκτελείται ως kernel module. Ιδανικά θα μπορούσε να υπάρχει ένα σύστημα όπου σε τακτά χρονικά διαστήματα θα επιβεβαιώνει την σωστή λειτουργία με τυχαία παραδείγματα ώστε ανιχνευθεί κάποια απόκλιση από την κανονική λειτουργία.

Ανάγκη για ασφάλεια του μοντέλου

Εδώ εξετάζουμε επιθέσεις που στοχεύουν το ίδιο το μοντέλο και όχι τη διεργασία που το εκτελει.

Η κατηγορία αφορά επιθέσεις που σχετίζονται αποκλειστικά σε ML. Αυτές περιλαμβάνουν κυρίως adversarial attacks όπως dataset poisoning και model poisoning attacks. Στην πρώτη περίπτωση ο επιτιθέμενος εισάγει σχεδιασμένα δεδομένα στο dataset με σκοπό να δημιουργήσει ένα pattern το οποίο δεν θα ανιχνεύεται στην συνέχεια από το μοντέλο. Στην δεύτερη περίπτωση εισάγει δεδομένα κατα την φάση της πρόβλεψης ώστε να "εκβιάσει" ένα αποτέλεσμα από το μοντέλο κάτι που μπορεί να επιτευχθεί εύκολα αν ο επιτιθέμενος έχει στην διάθεση του το dataset. Ένα τέτοιο παράδειγμα είναι το malware που προσπαθεί να παρακάμψει ένα σύστημα ανίχνευσης.

Ανάγκη για authentication

Αυτή η περίπτωση δεν αφορά μόνο το σενάριο να αποφύγει ο επιτιθέμενος τον έλεγχο αλλά αφορά και το σενάριο όπου ο επιτιθέμενος εισάγει δεδομένα στο μοντέλο που αφορούν άλλα processes και μπορεί με αυτόν τον τρόπο να προκαλέσει κατάσταση DoS για νόμιμους χρήστες. Αυτή η περίπτωση αφορά το δικό μας σενάριο χρήσης αλλά παράλληλα αποτελεί και γενικό πρόβλημα. Το IDS θα πρέπει να μπορεί να ξεχωρίζει νόμιμη πληροφορία από παράνομη πληροφορία και να εφαρμόζει μέτρα για να το πετύχει. Αυτό το σενάριο δεν το αντιμετωπίσαμε αλλά ένα production σύστημα θα πρέπει να το αντιμετωπίσει διότι αν ο επιτιθέμενος καταφέρει να στέλνει δεδομένα χωρίς έλεγχο μπορεί να αχρηστεψει το IDS ή ακόμα χειρότερα όλο το σύστημα που παρακολουθείται.

Ανάγκη για πλήρη datasets

Εδώ θα αναλύσουμε αυτό που με συντομία αναφέραμε στην ενότητα με τα μειονεκτήματα. Στην βιβλιογραφία είδαμε πολλές λύσεις να πετυχαίνουν σχεδόν άριστη απόδοση αλλά όλα αυτά σε benchmark datasets. Τα dataset αυτά, ένα εκ των οποίων χρησιμοποιήσαμε και εμείς, καλύπτουν ένα μικρό ποσοστό κατηγοριών επιθέσεων και αυτό αποτελεί πρόβλημα γιατί οι αλγόριθμοι είναι άχρηστοι χωρίς δεδομένα. Πιο συγκεκριμένα το dataset που χρησιμοποιήσαμε καλύπτει 5 είδη επιθέσεων. Το πρώτο είναι το σενάριο των επιθέσεων DoS. Το σενάριο αυτό είναι σχετικά εύκολο να ανιχνευθεί. Το ίδιο ισχύει και για τις επιθέσεις τύπου probe. Στην πράξη όμως κανένα σύστημα IDS δεν θα μπορούσε να βγει σε production έχοντας εκπαιδευτεί σε αυτό ή παρόμοιο dataset. Αυτό συμβαίνει διότι εκτός του ότι υπάρχουν άλλου είδους επιθέσεις για τις οποίες δεν υπάρχουν samples, κάποιες από αυτές τις επιθέσεις δεν είναι δυνατόν να ανιχνευθούν με την ίδια ευκολία. Για παράδειγμα οι επιθέσεις injection είναι απλές επιθέσεις να ανιχνευθούν αλλά δεν υπάρχουν δημόσια διαθέσιμα datasets που να περιλαμβάνουν τέτοια σενάρια. Μία ακόμα πιο δύσκολη κατάσταση προκύπτει από επιθέσεις για τις οποίες δεν έχουμε datasets και επιπλέον είναι δύσκολο να ανιχνευθούν. Τέτοιες επιθέσεις είναι οι passive επιθέσεις. Τέλος το σενάριο των Advanced Persistent Threats είναι το δυσκολότερο δεδομένου ότι απαιτεί συσχέτιση γεγονότων τα οποία χρονικά μπορεί να απέχουν πολύ. Το Esper CEP engine προβλέπει και την χρήση ιστορικών δεδομένων εκτός από real time streams αλλά το σενάριο μπορεί να γίνει πολύ πολύπλοκο. Ενδεικτικά παρουσιάζουμε κατηγορίες επιθέσεων οι οποίες αναφέρονται εδώ**[129]** για να δείξουμε πόσο μακριά από το βέλτιστο βρισκόμαστε.

- Eavesdropping

Αυτή η κατηγορία ανήκει στις passive επιθέσεις πράγμα που την καθιστά ιδιαίτερα δύσκολη στην ανίχνευση.

- Replay attack

Αυτή η κατηγορία αποτελεί σοβαρή επίθεση της οποίας το μόνο χαρακτηριστικό είναι ότι δημιουργεί κίνηση με βάση νόμιμη κίνηση. Από μόνη της στο naive σενάριο είναι πολύ δύσκολο να ανιχνευθεί διότι πρόκειται για νόμιμη κίνηση που επαναλαμβάνεται.

- Man-in-the-middle attack

Δύσκολο να ανιχνευθεί και η χρήση ML ίσως να μην προσφέρει τιποτα. Μπορεί να ανιχνευθεί αν υπάρχουν αποθηκευμένα τοπικά certificates και public keys για την οντότητα που πραγματοποιεί την επικοινωνία.

- Impersonation attack

Active επίθεση γενικά εύκολο να ανιχνευθεί.

- Malware attack

Εύκολο να ανιχνευθεί σε μεγάλη κλίμακα. Δύσκολο όταν το malware χρησιμοποιείται για πρώτη φορά.

Η λίστα αυτή δεν είναι εξαντλητική και δεν πραγματοποιήσαμε κατηγοριοποίηση για τα APTs. Με βάση αυτά βλέπουμε ότι ένα μοντέλο ML το οποίο πετυχαίνει σχεδόν 100% accuracy σε επιθέσεις DoS και probe δεν μπορει καν να χαρακτηριστεί σαν IDS.

Ανάγκη για αρκετά δεδομένα στο dataset

Αυτό είναι προφανές ιδιαίτερα για datasets που χαρακτηρίζονται ως imbalanced ή/και σε σενάρια Deep Learning. Όπως είδαμε κατα την διάρκεια του implementation η απόδοση στα minority classes ήταν υποβέλτιστη σε σχέση με τις άλλες. Επίσης είδαμε ότι το τυχαίο split στις minority κλάσσεις παίζει ρόλο στο evaluation κάτι που αποτελεί μεγάλο πρόβλημα και είναι ο λόγος που δεν κάναμε SMOTE.

Πρόβλημα με τα τελικά features

Όπως αναφέραμε η διαδικασία του feature selection μπορεί να γίνει με πολλές επιλογές. Η αντιμετώπιση που επιλέξαμε είναι να κρατήσουμε αυτά που δίνουν την περισσότερη πληροφορία. Βλέπουμε όμως ότι καταλήξαμε να έχουμε 2/10 features που προέρχονται από την συνολική δραστηριότητα ενός δικτύου και όχι από το host που παρακολουθείτε αυτό αποτελεί πρόβλημα διότι δεν μπορούμε να συγκεντρώσουμε δεδομένα από ένα δίκτυο. Η επιλογή αυτή μας υποχρεώνει να δημιουργήσουμε ένα hybrid IDS. Εκτός από αυτό είδαμε ότι κάποια features αποτελούν flow based features, άλλα αφορούν features τα οποία είναι διαθέσιμα όταν τελειώσει μία σύνδεση όπως το duration και άλλα αφορούν στιγμιαιες μετρήσεις. Η συλλογή όλων αυτών για την πρόβλεψη πολύ πιθανόν να μην είναι λειτουργική. Για παράδειγμα ο επιτιθέμενος μπορεί να μην τερματίσει ποτέ την σύνδεση μετά από μία επιτυχημένη επίθεση R2L με αποτέλεσμα να μην γίνει ποτέ inference για αυτό το process. Μία πιθανή λύση για αυτό είναι να εκτελούμε inference με βάση τα timestamps που έχουμε διαθέσιμα αλλά είτε βελτιώσει την απόδοση στην περίπτωση αυτή το πρόβλημα γενικά παραμένει. Επίσης πρόβλημα αποτελούν τα data points με τιμές που πρέπει να κανονικοποιηθούν και βρίσκονται εκτός range από αυτές που υπήρχαν στο dataset. Αυτό μπορούμε να το αντιμετωπίσουμε δίνοντας ακραίες τιμές σε όσα data points βρίσκονται εκτός range. Σίγουρα όμως αυτό επηρεάζει την απόδοση του μοντέλου.

Βέλτιστη απόδοση για το event correlation

Στο κομμάτι της υλοποίησης του event correlation παρουσιάσαμε πολλούς τρόπους για να επιτύχουμε τη λειτουργικότητα. Κάποιοι από αυτούς τους τρόπους χρησιμοποιούν πολλούς μηχανισμούς που προσφέρει το Esper και άλλοι λιγότερους. Γενικά το σενάριο που υλοποιήσαμε είναι σχετικά απλό σε σχέση με αυτά που μπορεί να αντιμετωπίσει το Esper. Θεωρούμε με βάση το ότι το Esper εφαρμόζει optimizations ως προς τι πληροφορία κρατάει και πως κάνει διαθέσιμα

δεδομένα στα statements η επιλογή των key segmented context partitions και η χρήση subscriber αντί για listener είναι η ιδανική και δίνει τα καλύτερα αποτελέσματα θεωρητικά. Παρόλα αυτά πριν γίνει το deployment θα έπρεπε να αξιολογηθεί αυτό καθώς το module μπορεί να αποτελέσει bottleneck της όλης διαδικασίας.

Με βάση όλα αυτά βλέπουμε ότι υπάρχουν πολλά θέματα που πρέπει να αντιμετωπιστούν ώστε να μπορούμε να πούμε ότι έχουμε ένα σύστημα που μπορεί να δουλέψει με αξιόπιστο και αποτελεσματικό τρόπο. Γενικά η χρήση μοντέλων ML έναντι στατικών υπογραφών δίνει πολύ καλύτερα αποτελέσματα και ως προς το latency αλλά και τα metrics. Μία στατική υπογραφή ναι μεν θα ανιχνεύσει μία συγκεκριμένη επίθεση με μεγάλη ακρίβεια αλλά το μοντέλο ML θα ανιχνεύσει και παραλλαγές της με το ίδιο μοντέλο αντί να χρειάζεται να δημιουργήσει νέες υπογραφές για κάθε παραλλαγή. Τέλος, να τονίσουμε ότι οι περισσότερες δουλειές αναφέρονται σε δημιουργία μοντέλων που δίνουν καλά αποτελέσματα σε benchmark datasets. όπως είδαμε υπάρχουν πολλά ακόμα που πρέπει να γίνουν πέρα από αυτό.

Βιβλιογραφία

Επιστημονικά άρθρα

[1] Satrajit Chatterjee, Piotr Zielinski "On the generalization mystery in deep learning"
[2] Farhad Maleki et al "Generalizability of Machine Learning Models:Quantitative Evaluation of Three Methodological Pitfalls"
[3] Dimiter Dobrev "A Definition of Artificial Intelligence"
[4] Stefan Maetschkea et al "Understanding in Artificial Intelligence"
[5] Soumil Rathi "Approaches to Artificial General Intelligence: An Analysis"
[6] Amirsina Torfi et al "Natural Language Processing Advancements By Deep Learning: A Survey"
[7] Zhixing Tana et al "Neural Machine Translation: A Review of Methods, Resources, and Tools"
[8] Liping Zhao et al "Classification of Natural Language Processing Techniques for Requirements Engineering"
[9] Jacob Devlin et al "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"
[10] Keting Lu et al "Robot Representation and Reasoning with Knowledge from Reinforcement Learning" **DUP**
[11] Keting Lu et al "Robot Representation and Reasoning with Knowledge from Reinforcement Learning" **DUP**
[12] IntechOpen SeriesArtificial Intelligence, Volume 7 "Machine Learning Algorithms, Models and Applications"
[13] Zixing Song et al "Graph-based Semi-supervised Learning: A Comprehensive Review"
[14] Springer book "Graph Neural Networks Foundations Frontiers and Applications"
[15] Katarzyna Janocha, Wojciech Marian Czarnecki "On Loss Functions for Deep Neural Networks in Classification"
[16] Shiliang Sun, Zehui Cao, Han Zhu, and Jing Zhao "A Survey of Optimization Methods from a Machine Learning Perspective"
[17] T. Soni Madhulatha "An Overview On Clustering Methods"
[18] Fadi AlMahamid, Katarina Grolinger "Reinforcement Learning Algorithms: An Overview and Classification"
[19] Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency "Foundations & Recent Trends In Multimodal Machine Learning: Principles, Challenges, & Open Questions"
[20] Neelamadhab Padhy, Dr. Pragnyaban Mishra, and Rasmita Panigrahi "The Survey of Data Mining Applications And Feature Scope"
[21] Nhan Cach Dang, María N. Moreno-Garcíaand Fernando De la Prieta "Sentiment Analysis Based on Deep Learning: A Comparative Study"
[22] Ashish Vaswani et al "Attention Is All You Need"
[23] Muhammad Mudassar Qureshi, Amjad Farooq, Muhammad Mazhar
Qureshi, "Current eHealth Challenges and recent trends in eHealth applications"
[24] Wenjie Ruan et al "Adversarial Robustness of Deep Learning: Theory, Algorithms, and Applications"
[25] Emmanouil Antonios Platanios et al "Learning from Imperfect Annotations"
[26] Yazhou Ren et al "Deep Clustering: A Comprehensive Survey"
[27] Dongkuan Xu, Yingjie Tian "A Comprehensive Survey of Clustering Algorithms"
[28] William L. Hamilton "Graph Representation Learning"
[29] C.O.S. Sorzano et al "A survey of dimensionality reduction techniques"
[30] Dor Bank, Noam Koenigstein, Raja Giryes "Autoencoders"

[31] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, Anil Anthony Bharath "A Brief Survey of Deep Reinforcement Learning"
[32] Anonymous Authors "Incremental Learning On Growing Graphs"
[33] Maher Maalouf "Logistic regression in data analysis: an overview"
[36] W.Z. Liu, A.P. White "The Importance of Attribute Selection Measures in Decision Tree Induction"
[66] Shi Dong, Ping Wang, Khushnood Abbas "A survey on deep learning and its applications"
[68] Volodymyr Mnih et al "Playing Atari with Deep Reinforcement Learning"
[69] William L. Hamilton et al "Inductive Representation Learning on Large Graphs"
[74] Hojjat Salehinejad et al "Recent Advances in Recurrent Neural Networks"
[76] Daniel Schatz et al "Towards a More Representative Definition of Cyber Security"
[78] Fran Casino et al "Research trends, challenges, and emerging topics of digital forensics: A review of reviews"
[80] George Karantzas, Constantinos Patsakis "An Empirical Assessment of Endpoint Detection and Response Systems against Advanced Persistent Threats Attack Vectors"
[81] Ministry of Economy, Trade and Industry (METI) "Cybersecurity Management Guidelines"
[83] Chris Simmons et al "AVOIDIT: A Cyber Attack Taxonomy"
[84] Fadi Mohsen et al "A Taxonomy for Large-Scale Cyber Security Attacks"
[85] Bonnie Zhu et al "A Taxonomy of Cyber Attacks on SCADA Systems"
[86] Chuck Easttom et al "A Modified McCumber Cube as a Basis for a Taxonomy of Cyber Attacks"
[87] Richard P. Lippmann "Evaluating Intrusion Detection Systems:The 1998 DARPA Off-line Intrusion Detection Evaluation*"
[88] Gustavo González-Granadillo et al "Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures"
[89] Pedro Ramos Brandao et al "Extended Detection and Response Importance of Events Context"
[90] ibm.com "What is SIEM"
[92] Marco Bucchi et al "Foundations of Complex Event Processing"
[93] Nikos Giatrakos et al "Complex Event Recognition in the Big Data Era: A Survey"
[96] David Luckham "The Power of Events An Introduction to Complex Event Processing in Distributed Enterprise Systems"
[97] Michael Eckert "A CEP Babelfish: Languages for Complex Event Processing and Querying Surveyed"
[98] Alexandros Bousdekis et al "A Framework for Integrated Proactive Maintenance Decision Making and Supplier Selection"
[101] Bruno Berstel et al "Reactive Rules on the Web"
[102] Rajeev Alur et al "Automata for modeling real-time systems"
[104] Natalia G. Miloslavskaya "Analysis of SIEM Systems and Their Usage in Security Operations and Security Intelligence Centers"
[107] mitre paper "Intrusion Detection System Requirements"
[108] Yasir Hamid et al "IDS Using Machine Learning -Current State of Art and Future Directions"
[111] Hongyu Liu, Bo Lang "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey"
[112] Elike Hodo et al"Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey"
[113] Geeta Kocher, Gulshan Kumar "Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges"
[114] Lekha Jayabalan et al "A Comprehensive Study on Classification of Passive Intrusion and Extrusion Detection System"
[115] Fahimeh Farahnakian et al "A deep auto-encoder based approach for intrusion detection system"
[116] Jihyun Kim et al"Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection"

[117] Chuanlong Yin et al "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks"
[118] Kehe Wu, Zuge Chen, Wei Li "A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks"
[119] Tuan A Tang et al "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking"
[120] Zakaria El Mrabet et al "Deep Learning-Based Intrusion Detection System for Advanced Metering Infrastructure"
[121] Yuyang Zhoua et al "Building an Efficient Intrusion Detection System Based on Feature Selection and Ensemble Classifier"
[122] Baha Rababah et al "Hybrid Model For Intrusion Detection Systems"
[123] Yogita B. Bhavsar, Kalyani C.Waghmare "Intrusion Detection System Using Data Mining Technique: Support Vector Machine"
[124] Mohammad Almseidin et al "Evaluation of Machine Learning Algorithms for Intrusion Detection System"
[125] L.Dhanabal "A Study on NSL-KDD Dataset for IntrusionDetection System Based on Classification Algorithms"
[127] Mohammad Wazida et al "Uniting cyber security and machine learning: Advantages, challenges and future research"

Διαδικτυακές Πηγές/Blogs

[34] www.mastersindatascience.org "What Is Logistic Regression?"
[35] Diego Lopez Yse, towards data science "The Complete Guide to Decision Trees"
[37] Neelam Tyagi www.analyticssteps.com "What is Information Gain and Gini Index in Decision Trees"
[38] machinelearningmastery.com "Overfitting and Underfitting With Machine Learning Algorithms"
[39] Edward Krueger towards data science "Build Better Decision Trees with Pruning"
[40] cse.usf.edu "Reduced Error Pruning for Decision Trees"
[41] Tony Yiu towards data science "Understanding Random Forest"
[42] Joseph Rocca towards datascience "Ensemble methods: bagging, boosting and stacking"
[43] Marco Peixeiro towards data science "The complete guide to support vector machine(SVM)"
[44] Indhumathy Chelliah towards data science "An Introduction to K-Nearest Neighbors Algorithm"
[45] Collins Ayuya section.io"Parametric versus Non-Parametric Models"
[46] A.Aylin Tocuc "Generative versus Discriminative Algorithms "
[47] Gaurav Chauhan towards data science "All about Naive Bayes"
[48] javapoint.com "Machine Learning Pipeline"
[49] vidora.com "What is a Machine Learning Pipeline"
[50] Eugenio Zuccarelli towards data science "Handling Categorical Data, The right way"
[51] Emre Rençberoğlu towards data science "Fundamental Techniques of Feature Engineering for Machine Learning"
[52] Tara Boyle towards data science "Dealing with Imbalanced Data" **DUP**
[53] machinelearningmastery.com "Training-validation-test split and cross-validation done right"
[54] machinelearningmastery.com "Hyperparameter Optimization With Random Search and Grid Search"
[55] Ramya Vidiyala towards data science "How to Select the Right Machine Learning Algorithm"
[56] Ejiro Onose neptune.ai blog "Explainability and Auditability in Machine Learning Definitions Techniques and tools"
[57]Shervin Minaee towards data science "20 Popular Machine Learning Metrics. Part 1: Classification & Regression Evaluation Metrics"
[58] Indhumathy Chelliah towards data science "Confusion Matrix — Clearly Explained"

[59] Dmytro Nikolaiev (Dimid) towards data science“Overfitting and Underfitting Principles”
[60] Prashant Gupta towards data science “Regularization in Machine Learning”
[61] Joos Korstanje towards data science “SMOTE”
[62]Harsh Yadav towards data science “Dropout in Neural Networks”
[63] Olga Chernytska towards data science “Complete Guide to Data Augmentation for Computer Vision”
[64] Tara Boyle towards data science “Dealing with Imbalanced Data” **DUP**
[65] machinlearningmastery.com “SMOTE for Imbalanced Classification with Python”
[67] Sumit Saha towards data science “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way”
[70] Pablo Ruiz towards data science “Neural Networks I: Notation and building blocks”
[71] Simeon Kostadinov towards data science “Understanding Backpropagation Algorithm”
[72]Sanket Doshi towards data science “Various Optimization Algorithms For Training Neural Network”
[73] Joseph Rocca towards data science “Understanding Generative Adversarial Networks (GANs)”
[xxx] Michael Phi towards data science “Illustrated Guide to LSTM’s and GRU’s: A step by step explanation”
[75] Matt Przybyla towards data science “When to Avoid Deep Learning”
[77] IEEE SPectrum “The Real Story of Stuxnet”
[79] globalknowledge.com “Cyber Security Glossary and Terms”
[82] cobalt.io “Confidentiality Integrity Availability in Cyber Security”
[91] ibm.com “Logging and Monitoring”
[94] loggly.com “Analyzing Linux Logs”
[95] loggly.com “Linux Logs Basics”
[99] Jonathan Johnson bmc.com “Event Stream Processing”
[100] espertech “Esper Tech Requirements”
[103] rapid7.com “What is Threat Detection”
[105] logrythm.com “A Guide to EDR, NDR, XDR, and SIEM”
[106] Thu Pham bluira.com “SIEM, SOC, SOAR, MDR, EDR & XDR Defined”
[109] Coleman Wolf “Breaking Down the Pros and Cons of AI in Cybersecurity”
[110] servreality.com “Artificial-Intelligence-In-Cybersecurity-Pros-And-Cons”
[126] Max Pagels towards data science “Introducing One of the Best Hacks in Machine Learning: the Hashing Trick”
[128] espertech.com “Esper Reference version 8.8.0”