ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΑ

ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ

«ΑΣΦΑΛΕΙΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ»

Διπλωματική Εργασία

«Εξασφάλιση ανωνυμίας στην αποστολή αναφορών με χρήση Onion Networks»

Ευάγγελος Δ. Κατσαδούρος

Υπεύθυνος Καθηγητής: Κωνσταντίνος Λαμπρινουδάκης

ΑΘΗΝΑ

ΔΕΚΕΜΒΡΙΟΣ 2018

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες, σε ένα ενδιαφέρον γνωστικό αντικείμενο, όπως αυτό της Εξασφάλισης ανωνυμίας στην αποστολή αναφορών με χρήση Onion Networks. Την προσπάθειά μου αυτή υποστήριξε ο επιβλέπων καθηγητής μου κ. Κωνσταντίνος Λαμπρινουδάκης, τον οποίο θα ήθελα να ευχαριστήσω.

Ακόμα θα ήθελα να ευχαριστήσω την οικογένειά μου για την δύναμη και την εμψύχωση που μου έδινε καθ' όλη τη διάρκεια των σπουδών μου. Επίσης να ευχαριστήσω την ερευνητική ομάδα Co.N.Se.R.T του Πανεπιστημίου Δυτικής του τμήματος Ηλεκτρολόγων και Ηλεκτρονικών Μηχανικών για τη βοήθεια και τις υπηρεσίες τις οποίες μου προσέφερε καθώς και το κ. Χαράλαμπο Πατρικάκη για τη βοήθεια και τις συμβουλές τις οποίες μου έδωσε.

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική στοχεύει στην κατασκευή ενός δικτύου ανωνυμίας βασισμένο στο TOR [6], για αποστολή αναφορών περιστατικών σε αρχές προστασίας του πολίτη. Το δίκτυο αυτό θα δίνει την αίσθηση στον χρήστη της πραγματικής ανωνυμίας και θα προστατεύει την επικοινωνία και ταυτότητα του χρήστη ενώ παράλληλα θα είναι διαχειρίσιμο από λογισμικό το οποίο δε θα δεσμεύει τους εθελοντές του δικτύου. Η παρούσα διπλωματική δομείται σε πέντε ενότητες οι οποίες παρουσιάζονται συνοπτικά στη συνέχεια. Στη πρώτη ενότητα γίνεται εισαγωγή στο αντικείμενο της διπλωματικής “Ανωνυμία με χρήση Onion Networks”, στην οποία παρουσιάζονται οι στόχοι και οι λόγοι που με ώθησαν στην παρούσα διπλωματική και κατασκευή. Στη δεύτερη ενότητα παρουσιάζονται οι λύσεις που υπάρχουν σήμερα για ανωνυμία με Onion Networks και τις οποίες συμβουλεύτηκα για να μπορέσω να κατασκευάσω το δίκτυο ανωνυμίας της παρούσας διπλωματικής. Στη τρίτη ενότητα παρουσιάζεται η αρχιτεκτονική του δικτύου ανωνυμίας το οποία επέλεξα να κατασκευάσω στη παρούσα διπλωματική. Στην ενότητα αυτή υπάρχουν σχεδιαγράμματα της αρχιτεκτονικής και διαγράμματα ροής της επικοινωνίας των μερών του δικτύου. Στην τέταρτη ενότητα γίνεται παρουσίαση των αποτελεσμάτων της κατασκευής του δικτύου και σχόλια για τη λειτουργία του καθώς και για μέρη του κώδικα. Η παρούσα διπλωματική κλείνει με τον επίλογο ο οποίος αποτελεί και την πέμπτη ενότητα της διπλωματικής στην οποία ενότητα γίνεται μια σύνοψη της διπλωματικής, κάποια σχόλια για το αντικείμενο της πτυχιακής και μελλοντικοί προβληματισμοί και προτάσεις για την εξέλιξη της ιδέας η οποία παρουσιάζεται στη παρούσα διπλωματική. Τέλος υπάρχει και ένα παράρτημα στο οποίο υπάρχει ο κώδικας των βασικών μερών του δικτύου που είναι ο onion-proxy και ο onion-router οι οποίοι αποτελούν και το δίκτυο.

ABSTRACT

The present thesis aims to develop an anonymous network based on the TOR [6], to send incident reports to civil protection authorities. The present network will provide user with anonymity over this network and will protect the user's communication and identity. It will be managed by software that will not provide volunteers with extra duties. The present thesis is structured in five sections, which are summarized below. The first section introduces the subject of the thesis "Ensure anonymity in sending reports using Onion Networks", and presents the reasons of choosing this topic. The second section presents the existing solutions in anonymity with Onion Networks so far. The third section presents the anonymity architecture I chose for my network. This section consists of architecture diagrams and communication flow diagrams of the parts of the network. The fourth section presents the results of the developed network and comments on its operation as well as parts of the code. The present thesis closes with the conclusions which are also the fifth section of the thesis. In this section there is a summary of the thesis, some comments on the subject of dissertation and future considerations and suggestions for the developed network. Finally there is an appendix in which there is the code of the main parts of the network that is the onion-proxy and the onion-router.

Περιεχόμενα

1 Εισαγωγή

Πριν από περίπου 30 χρόνια γεννήθηκε το Διαδίκτυο, το οποίο άλλαξε σε μεγάλο βαθμό τις ζωές των ανθρώπων. Με το πέρασμα των χρόνων, το Διαδίκτυο γινόταν όλο και μεγαλύτερο κομμάτι της καθημερινότητας του ανθρώπου. Σήμερα δε θα ήταν υπερβολή αν λέγαμε ότι η ζωή μας βρίσκεται ολοκληρωτικά στο Διαδίκτυο και πολλές ενέργειες μας εξαρτώνται από αυτό (πληρωμή λογαριασμών, ιατρικό ιστορικό, διαχείριση οικιακών συσκευών κ.α). Το Διαδίκτυο έχει προσφέρει στην ανθρωπότητα πολλά καλά, όπως η άμεση επικοινωνία μεταξύ των ανθρώπων σε όλο τον κόσμο, η τακτοποίηση πολλών υποχρεώσεων χωρίς κόπο, η εξέλιξη της οικονομίας κ.α. Παρόλα αυτά όμως, η φύση του διαδικτύου είναι τέτοια η οποία μπορεί να προσβάλει την ιδιωτικότητα του ανθρώπου στη καθημερινότητα του.

Ο άνθρωπος χρησιμοποιεί καθημερινά το διαδίκτυο για τις προσωπικές του αγορές, για τον έλεγχο των ιατρικών δεδομένων του, τη διαχείριση των οικιακών του συσκευών, την ψυχαγωγία του, την επικοινωνία του με φίλους και συγγενείς ακόμη και για την αποθήκευση και διαχείριση των καθημερινών του υποχρεώσεων (ημερολόγιο). Όλες οι προαναφερθείσες ενέργειες πραγματοποιούνται στο διαδίκτυο δημιουργώντας στοιχεία τα οποία μπορούν να προσβάλουν την ιδιωτικότητα του χρήστη. Σε αρκετές περιπτώσεις ηλεκτρονικών υπηρεσιών ο χρήστης παραχωρεί (εις γνώσιν του) προσωπικά δεδομένα προκειμένου να μπορεί να χρησιμοποιήσει της υπηρεσίες αυτές. Φυσικά αυτό έχει ως αποτέλεσμα κάθε ενέργεια του χρήστη να συνδέεται μοναδικά με αυτόν/η.

1.1 Το Πρόβλημα

Σε πολλές περιπτώσεις διαδικτυακών υπηρεσιών τον χρήστη δεν τον απασχολεί η ύπαρξη στοιχείων που να τον συσχετίζουν με αυτές τις υπηρεσίες. Υπάρχουν όμως και υπηρεσίες οι οποίες έχουν αποδειχθεί, ότι οι χρήστες είναι διστακτικοί στην χρήση τους λόγο ύπαρξης στοιχείων που αποδεικνύουν την χρήση της υπηρεσίας, από έναν συγκεκριμένο χρήστη και πολλές φορές υπάρχουν στοιχεία για την ακριβή χρήση της υπηρεσίας. Τέτοιες υπηρεσίες είναι αυτές που έχουν να κάνουν με την ασφάλεια του πολίτη. Είναι πολλές φορές που δεν αναφέρονται περιστατικά σχετικά με την ασφάλεια του πολίτη καθώς ο αναφέρων το περιστατικό φοβάται για την εμπλοκή του στην υπόθεση. Σύμφωνα με τον Jim Thomas [14], μερικοί από τους λόγους για τους οποίους οι πολίτες διστάζουν στην αναφορά ενός περιστατικού ο φόβος για πιθανά αντίποινα του δράστη προς αυτούς. Ένας άλλος λόγος ο οποίος αναφέρει είναι η ντροπή που νιώθει ένα θύμα ειδικά σε περιπτώσεις σεξουαλικής κακοποίησης ή βιασμού. Επιπροσθέτως αναφέρει ότι τα θύματα επιθυμούν να διατηρήσουν την ιδιωτικότητα τους και να μην μαθευτεί στην κοινωνία ένα τέτοιο περιστατικό το οποίο να σχετίζεται με αυτούς. Σε ένα άλλο άρθρο του Carol Fredrickson [15], αναλύονται οι λόγοι για τους οποίους δεν αναφέρονται περιστατικά στο χώρο εργασίας. Σε αυτό το άρθρο αναφέρονται λόγοι οι οποίοι έχουν να κάνουν με τη ταυτότητα του εργαζομένου. Τέτοιοι λόγοι είναι ο φόβος των αντιποίνων προς αυτόν που έχει αναφέρει ένα περιστατικό, στην οποία περίπτωση μπορεί να δημιουργηθεί κλίμα το οποίοι να βλάπτει τον εργαζόμενο ο οποίος ανέφερε ένα περιστατικό. Άλλες περιπτώσεις οι οποίες αναφέρονται στο άρθρο είναι η περίπτωση ο

εργαζόμενος να χαρακτηριστεί ως “καρφί” στο χώρο εργασίας, με αποτέλεσμα οι συνάδελφοι του να μην του συμπεριφερθούν σωστά και η περίπτωση να έρθει σε κρούση με τον προϊστάμενο του με αποτέλεσμα να δημιουργηθούν δυσκολίες στην εξέλιξη του και στην εργασία του. Οι ανωτέρω λόγοι είναι αυτοί που με οδήγησαν στην σχεδίαση και κατασκευή ενός ελεγχόμενου δικτύου ανωνυμίας για την χρήση συγκεκριμένων υπηρεσιών, όπως αυτές της ασφάλειας του πολίτη. Ένα δίκτυο ανωνυμίας το οποίο θα εγγυάται στο πολίτη ανωνυμία κατά την αναφορά του και ότι δεν θα υπάρξει η οποιαδήποτε επικοινωνία οι οποία μπορεί να οδηγήσει σε συμπεράσματα για την ταυτότητα του χρήστη.

1.2 Υπάρχουσες τεχνολογίες και αδυναμίες

Μέχρι τώρα υπάρχουν τεχνολογίες ανωνυμίας οι οποίες προσφέρουν μεταφορά αρχείων (BitBlinder), πλοήγηση στο διαδίκτυο (TOR, I2P) και κανάλια επικοινωνίας (BitMessage). Όλες οι λύσεις έως τώρα βασίζονται στην ύπαρξη P2P δικτύων τα οποία κατά κύριο λόγο συντηρούνται από εθελοντές οι οποίοι προσφέρουν πόρους για να στηρίξουν τις τεχνολογίες αυτές. Όποιος επιθυμεί γίνεται κόμβος του δικτύου και περνάνε πακέτα από αυτόν ώστε να υπάρξει ανωνυμία. Έτσι δουλεύει και το TOR το οποίο αποτελεί σήμερα τον πιο δημοφιλή μηχανισμό ανωνυμίας στο διαδίκτυο. Το πρόβλημα στις λύσεις που υπάρχουν είναι οι συνδέσεις που δημιουργούνται μεταξύ αποστολέα και παραλήπτη ώστε να επικοινωνούν αλλά και οι εθελοντές – κόμβοι αυτών των δικτύων για τους οποίους δεν υπάρχουν περιορισμοί. Αυτό έχει ως αποτέλεσμα ένας κόμβος να είναι πράγματι ένας εθελοντής, με σκοπό να βοηθήσει τη λειτουργία ενός τέτοιου δικτύου, αλλά μπορεί να είναι και κάποιος ο οποίος θέλει να παρακολουθήσει το δίκτυο και με βάση τις πληροφορίες που θα μαζέψει να εξάγει συμπεράσματα για τη ταυτότητα ενός χρήστη του δικτύου. Επίσης ένα άλλο πρόβλημα είναι η μελέτη του συστήματος και η εξαγωγή συμπερασμάτων από τη παρακολούθηση της δικτυακής κίνησης (όπως έχει συμβεί στο TOR). Στόχος στη παρούσα υλοποίηση είναι η κατασκευή ενός δικτύου το οποίο θα στηρίζεται από έμπιστους εθελοντές οι οποίοι θα εκτελούν το αντίστοιχο λογισμικό και θα βοηθούν τη λειτουργία του δικτύου. Ο χρήστης που θα κάνει χρήση του δικτύου για την αποστολή μιας αναφοράς θα ξέρει ότι η αναφορά του θα περνάει από έμπιστα δίκτυα (κόμβους) οι οποίοι δε θα παραβιάσουν και θα απειλήσουν την ανωνυμία του σε αυτή τη διαδικασία. Επίσης σε αυτή την υλοποίηση δε θα διατηρούνται συνδέσεις και πληροφορίες δρομολόγησης από τους κόμβους ώστε να μπορεί με κάποιο τρόπο ο παραλήπτης να φτάσει πίσω στον αποστολέα. Τέλος στην υλοποίηση που προτείνω οι κόμβοι οι οποίοι θα υπάρχουν θα τους διαχειρίζεται ένα λογισμικό ώστε σε τακτά χρονικά διαστήματα να τους διαγράφει και να τους ξαναδημιουργεί έτσι ώστε να καταστρέφεται όποια πληροφορία μπορεί να έχει υπάρξει. Με αυτό τον τρόπο τα μηχανήματα ανανεώνονται συνεχώς και οι εθελοντές δε χρειάζεται να κάνουν κάτι άλλο πέρα από τη διάθεση μερικών πόρων οι οποίοι είναι λίγοι καθώς το δίκτυο και οι λειτουργίες δεν απαιτούν πολλούς πόρους.

2 Τεχνολογίες Ανώνυμης Δικτύωσης

Η έννοια της ανώνυμης δικτύωσης και η κατασκευή μιας τέτοιας τεχνολογίας φαίνεται να ξεκίνησε στα μέσα του 1990 στα Ερευνητικά Εργαστήρια του Αμερικανικού Ναυτικού (U.S. Naval Research Laboratory) από τους ερευνητές Paul Syverson, Michael G. Reed και David Goldschlag [5]. Η κατασκευή του συνεχίστηκε από τον Οργανισμό Άμυνας Προηγμένων Ερευνητικών Προγραμμάτων της Αμερικής (Defense Advanced Research Projects Agency) και τελικά κατοχυρώθηκε από το Αμερικανικό Ναυτικό το 1998. Το συγκεκριμένο έργο συνέχισαν οι Roger Dingledine και Nick Mathewson το 2002 οι οποίοι μαζί με άλλα πέντε μέλη το 2006 δημιούργησαν το TOR ως μη κερδοσκοπικό οργανισμό. Το TOR σήμερα σύμφωνα με τις μετρήσεις του χρησιμοποιούν μέχρι και σήμερα πάνω από 2 εκατομμύρια χρήστες.[1][2]

2.1 Τεχνολογία Δικτύωσης Onion-Routing

Η τεχνολογία onion-routing χωρίζεται σε δύο γενιές, η πρώτη κατασκευάστηκε και δημοσιεύτηκε το 1998 από τους ερευνητές Paul Syverson, Michael G. Reed και David Goldschlag [4]. Οι ανωτέρω ερευνητές εφεύραν έναν μηχανισμό με τον οποίο ο χρήστης θα μπορούσε να έχει ιδιωτική πλοήγηση στο διαδίκτυο με χρήση ανώνυμων συνδέσεων σε αυτό. Όλος ο μηχανισμός στηρίχτηκε στην ιδέα με την οποία δουλεύουν οι proxy servers οι οποίοι αναλαμβάνουν να προωθούν πακέτα από τον αποστολέα στο παραλήπτη λειτουργώντας έτσι ως κρίκος σύνδεσης μεταξύ αποστολέα και παραλήπτη.

Το onion-routing στη πρώτη έκδοσή του παρουσιάζει ένα τρόπο δρομολόγησης πακέτων ο οποίος προσφέρει ανωνυμία στο χρήστη με χρήση πολλαπλών κόμβων (nodes) οι οποίοι προωθούν το πακέτο στον επόμενο μέχρι να φτάσει στον τελικό προορισμό. Εκτός από τους ενδιάμεσους κόμβους παρέχεται και πολλαπλή κρυπτογράφηση του πακέτου για κάθε κόμβο. Πιο συγκεκριμένα η λειτουργία του ξεκινάει με την εφαρμογή η οποία θα συνδεθεί με το application proxy το οποίο στη συνέχεια θα συνδεθεί με τον onion proxy το οποίο αναλαμβάνει να καθορίσει ένα μονοπάτι στο δίκτυο ανωνυμίας. Το onion proxy είναι αυτό που αναλαμβάνει την κατασκευή του onion. Το onion αποτελείται από ένα σύνολο πακέτων τα οποία είναι το ένα εμφωλευμένο στο άλλο. Κάθε πακέτο αφορά και ένα διαφορετικό κόμβο (onion-router) μέσα στο δίκτυο και κάθε τέτοιο πακέτο είναι κρυπτογραφημένο ώστε να μπορεί να το δει ο κόμβος για τον οποίο έχει κατασκευαστεί. Αμέσως μόλις κατασκευαστεί το μονοπάτι και το onion, τότε από τον onion proxy ξεκινά η αποστολή του onion προς κάθε κόμβο του δικτύου που έχει επιλεγεί. Σε κάθε onion router που φτάνει το πακέτο θα πρέπει αυτό να αποκρυπτογραφηθεί με το κατάλληλο κλειδί έτσι ώστε να μπορέσει το onion-router να προωθήσει το πακέτο στο αμέσως επόμενο. Σημαντικό στη λειτουργία του είναι ότι ο κάθε κόμβος μόλις παραλάβει το πακέτο ξέρει το κόμβο από τον οποίο το πήρε και τον κόμβο στον οποίο πρέπει να το στείλει και δε γνωρίζει καμία άλλη πληροφορία για το μονοπάτι το οποίο έχει κατασκευαστεί αλλά ούτε για τον αποστολέα και παραλήπτη. Όλο αυτό επιτυγχάνεται στη κρυπτογράφηση του κάθε πακέτου για τη δημιουργία του onion. [4]

Στη πρώτη γενιά του onion routing οι δημιουργοί έκαναν χρήση κρυπτογραφίας δημοσίου κλειδιού RSA για τη κρυπτογράφηση του κάθε πακέτου. Με αυτό τον τρόπο ο κάθε κόμβος μπορούσε να αποκρυπτογραφήσει μόνο την πληροφορία που τον αφορά και σε καμία περίπτωση δε μπορούσε να δει τη πληροφορία που αφορά άλλους κόμβους. Επίσης για να μπορέσει να αποτρέψει επιθέσεις επανάληψης σε κάθε πακέτο πρόσθετε μια χρονοσφραγίδα την οποία συμβουλευόταν κάθε κόμβος και αν δεν ήταν έγκυρη τότε απέρριπτε το πακέτο.

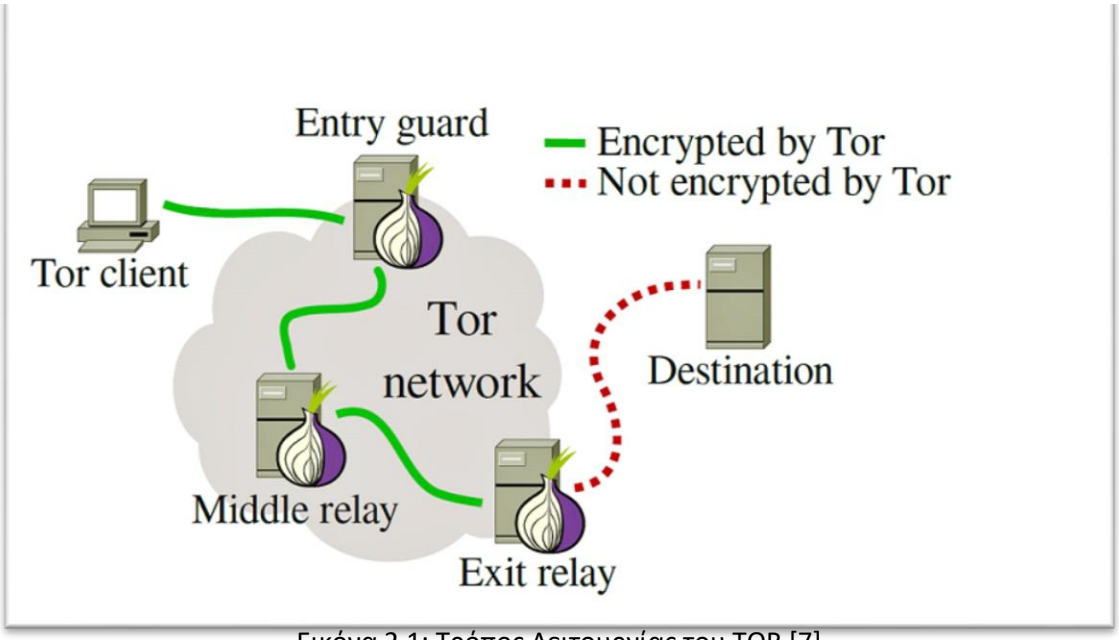
2.2 The Onion Router (TOR - 2η Γενιά)

Οι ερευνητές της πρώτης γενιάς έξι χρόνια μετά τη δημοσίευση του Onion Routing δημοσιεύουν τη δεύτερη γενιά η οποία αποτελεί το TOR. Το TOR είναι ελεύθερο λογισμικό της «The Tor Project Inc.» το οποίο μπορεί να χρησιμοποιηθεί για ανώνυμη πλοήγηση στο διαδίκτυο και σχεδιάστηκε και δημοσιεύτηκε από τους δημιουργούς της πρώτης γενιάς. Το TOR αποτέλεσε μια ολοκληρωμένη λύση με απλό σχεδιασμό και περισσότερες υπηρεσίες. Χρησιμοποιώντας το TOR μπορεί ο χρήστης να πλοηγηθεί ανώνυμα στο διαδίκτυο και με αυτό τον τρόπο να μην συσχετίσει κανείς τη ταυτότητα του με συγκεκριμένες υπηρεσίες ή ιστοσελίδες.

Το TOR δουλεύει χρησιμοποιώντας μια σειρά από κόμβους από τους οποίους περνάει η κίνηση του χρήστη μέχρι να φτάσει στο προορισμό της. Οι κόμβοι από τους οποίους περνάει ένα πακέτο μέχρι να φτάσει στο προορισμό του είναι τυχαίοι κόμβοι του δικτύου TOR. Οι κόμβοι του δικτύου είναι χρήστες οι οποίοι έχουν επιλέξει εθελοντικά να στηρίξουν τη λειτουργία του δικτύου εγκαθιστώντας και εκτελώντας το λογισμικό του TOR ώστε να είναι κόμβοι. Το TOR είναι ένας μη κερδοσκοπικός οργανισμός ο οποίος βασίζεται στους χρήστες του ώστε να στηριχθεί η λειτουργία του δικτύου. Την ανωνυμία δε την επιτυγχάνει μόνο με την ύπαρξη τυχαίων κόμβων από τους οποίους περνάει ένα πακέτο αλλά και με την ύπαρξης μιας ειδικής δομής των πακέτων που τα παρομοιάζει με κρεμμύδια. Ο λόγος για τον οποίο παρομοιάζει τα πακέτα αυτά με κρεμμύδια είναι η ύπαρξη πολλαπλών στρωμάτων κρυπτογράφησης για την απόκρυψη της πληροφορίας του χρήστη προς τον τελικό παραλήπτη από τους διάφορους κόμβους που περνάει.[6][7]

Η λειτουργία του TOR ξεκινάει από το λογισμικό του χρήστη και πιο συγκεκριμένα από το φυλλομετρητή (browser) TOR το οποίο ξεκινάει και χτίζει ένα κύκλωμα με τυχαίους κόμβους του δικτύου. Για να μπορέσει ο χρήστης να πλοηγηθεί ανώνυμα μέσω του TOR πρέπει να έχει διαπραγματευτεί με το κάθε κόμβο τα προσωρινά κλειδιά τα οποία θα χρησιμοποιούν για τη κρυπτογράφηση των πακέτων. Ο χρήστης θα διαπραγματευτεί με το πρώτο κόμβο το κλειδί το οποίο θα χρησιμοποιούν (κάνοντας χρήση του Diffie-Hellman) και στη συνέχεια μέσω του πρώτου κόμβου θα διαπραγματευτεί όλα τα υπόλοιπα κλειδιά. Ο χρήστης επικοινωνεί απευθείας μόνο με το πρώτο κόμβο του κυκλώματος και ποτέ με τους άλλους. Κάθε κόμβος μπορεί να μιλήσει με τον προηγούμενο του και με τον επόμενο. Έτσι για να μπορέσει ο πρώτος κόμβος να διαβιβάσει ένα πακέτο στο τρίτο κόμβο θα πρέπει απαραιτήτως να περάσει το πακέτο στο δεύτερο κόμβο και στην συνέχεια ο δεύτερος κόμβος να το στείλει στο τρίτο. Μόλις ο χρήστης συμπληρώσει όλα τα απαραίτητα κλειδιά τότε είναι έτοιμος για τη κατασκευή των

onions. Για να κατασκευαστεί το onion θα πρέπει το πακέτο κάθε κόμβου να κρυπτογραφείται με το κλειδί το οποίο έχει συμφωνήσει με το χρήστη και ενθυλακώνεται κρυπτογραφημένο στο πακέτο του προηγούμενου του κόμβου. Μόλις κατασκευαστεί το onion τότε μπορεί να σταλεί στο προορισμό του μέσω των κόμβων με την σειρά με την οποία έχουν επιλεγεί. [6][7]



Εικόνα 2.1: Τρόπος Λειτουργίας του TOR [7]

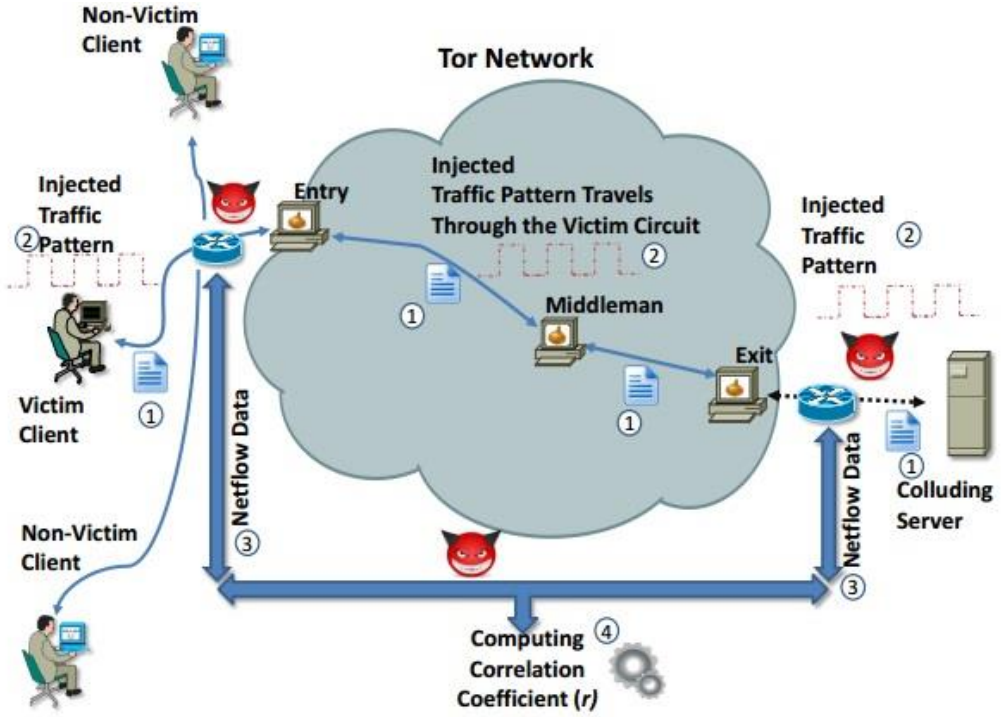
Το TOR διαθέτει επίσης τους Directory Servers στους οποίους υπάρχουν όλοι οι ενεργοί κόμβοι και πληροφορίες σχετικές με αυτούς. Οι Directory Servers είναι από τα βασικά μέρη του TOR καθώς με αιτήματα σε αυτούς οι χρήστες ενημερώνονται για τους ενεργούς κόμβους και μπορούν να χτίσουν το κύκλωμα από το οποίο θα περνάει η κίνηση τους. Οι Directory Servers αποτελούν HTTP servers οι οποίοι υπάρχουν για να παρακολουθούν τη κατάσταση του δικτύου. Όλοι οι κόμβοι για να γίνουν μέρος του δικτύου πρέπει να ενημερώσουν τους Directory Servers και να εγκριθούν από αυτούς. Αν κάποιος κόμβος δεν έχει περαστεί και εγκριθεί από τους Directory Servers τότε δε αποτελεί μέρος του δικτύου και δεν μπορεί να περάσει κίνηση χρηστών από αυτόν. Οι Directory Servers και τα κλειδιά αυτών είναι προ-εγκατεστημένα στο λογισμικό του χρήστη. [6][7]

Το TOR για να περιορίσει τις κακόβουλες ενέργειες μέσα στο δίκτυο δίνει τη δυνατότητα στους κόμβους εξόδου (exit nodes) να εφαρμόζουν πολιτικές εξόδου από το δίκτυο. Αυτές οι πολιτικές περιγράφουν τις διευθύνσεις οι οποίες είναι επιτρεπτές από το κόμβο καθώς και τις διαδικτυακές υπηρεσίες τις οποίες υποστηρίζει ο κόμβος (HTTP, SSH κλπ). Θα μπορούσαμε να πούμε ότι πρόκειται για τείχος προστασίας (firewall) σε κάθε κόμβο, στο οποίο ο κάθε κόμβος θέτει τους δικούς του κανόνες. Με το μηχανισμό αυτό το TOR δεν εξαλείφει τους κινδύνους και τις κακόβουλες ενέργειες στο δίκτυο αλλά σίγουρα τις περιορίζει σε σημαντικό βαθμό. Το TOR δίνει τη δυνατότητα για κρυφές υπηρεσίες. Με αυτό τον τρόπο μπορεί μια υπηρεσία να είναι διαθέσιμη μέσω του TOR χωρίς όμως οι χρήστες οι οποίοι τη χρησιμοποιούν να γνωρίζουν την

πραγματική διεύθυνση της. Για να το πετύχει αυτό η κρυφή υπηρεσία ανακοινώνει μια σειρά από κόμβους οι οποίοι αποτελούν σημεία εισόδου για την υπηρεσία και στη συνέχεια από αυτούς η κίνηση φτάνει στην υπηρεσία. Έτσι αποφεύγονται και επιθέσεις κατανεμημένης άρνησης υπηρεσιών (distributed denial of service) καθώς ο επιτιθέμενος χτυπάει το δίκτυο TOR και όχι την υπηρεσία. [6][7]

2.3 Επιθέσεις στο Δίκτυο Ανωνυμίας TOR

Το δίκτυο TOR αν και αποτελεί μια από τις πιο ολοκληρωμένες και ασφαλή λύσεις ανώνυμης δικτύωσης έχει το τρωτό της σημείο το οποίο τη καθιστά ευπαθή και απειλεί την ανωνυμία των χρηστών στο διαδίκτυο. Ο επιτιθέμενος εκμεταλλεύεται τις πληροφορίες οι οποίες προκύπτουν από τη λειτουργία του δικτύου και την αποστολή των πακέτων με στόχο να σπάσει την ανωνυμία ενός χρήστη. Η επίθεση αυτή ονομάζεται “Correlation Attack” και θεωρείται η μεγαλύτερη απειλή του TOR. Πιο συγκεκριμένα στην επίθεση αυτή ένας επιτιθέμενος που έχει στη κατοχή του το πρώτο και τελευταίο κόμβο μπορεί συσχετίζοντας πληροφορίες που έχουν να κάνουν με το χρόνο να σπάσει την ανωνυμία ενός χρήστη. Φυσικά για να συμβεί αυτό θα πρέπει ο επιτιθέμενος να έχει αρκετούς κόμβους στο δίκτυο ώστε να έχει πιθανότητες να έχει στη κατοχή του το πρώτο και τελευταίο κόμβο ενός κυκλώματος ενός χρήστη. Κάτι τέτοιο είναι αρκετά δύσκολο γιατί εύκολα μπορούμε να αναλογιστούμε τους πόρους τους οποίους θα χρειαστεί να έχει στη κατοχή του ο επιτιθέμενος για να μπορέσει να στήσει ένα πλήθος κόμβων το οποίο πλήθος να είναι τέτοιο ώστε να δώσει στον επιτιθέμενο πιθανότητα τέτοια ώστε να μπορεί να έχει πρώτο και τελευταίο κόμβο. Οι μόνοι οι οποίοι το έχουν καταφέρει είναι οι μυστικές υπηρεσίες οι οποίες είχαν μεγάλη υποστήριξη σε πόρους, και με αυτή τη πρακτική κατάφεραν να βρουν και να κατεβάσουν παράνομες υπηρεσίες από το DarkWeb οι οποίες δούλευαν με το TOR για να μπορούν να παρέχουν ανωνυμία στους χρήστες τους όπως και το SilkRoad. Αυτή η επίθεση μπορεί να πραγματοποιηθεί αν ο χρήστης χρησιμοποιεί απευθείας το δίκτυο TOR για να πλοηγηθεί ανώνυμα στο διαδίκτυο, καθώς ο πρώτος κόμβος θα είναι σε θέση να γνωρίζει τη διεύθυνση του αποστολέα και από το τελευταίο κόμβο θα μπορεί να μάθει το προορισμό. Αυτό μπορεί να αποτραπεί αν ο χρήστης κάνει χρήση VPN και TOR. Η επίθεση θα μπορεί πάλι να πραγματοποιηθεί απλά σε αυτή τη περίπτωση ο πρώτος κόμβος θα βλέπει τη διεύθυνση του παρόχου VPN που χρησιμοποιεί ο χρήστης με αποτέλεσμα να μην μπορεί να γνωρίζει τη πραγματική διεύθυνση του χρήστη. [10][12]



Εικόνα 2.2: Επίθεση στο δίκτυο TOR [12]

Σύμφωνα με όλα τα προαναφερθέντα διαπιστώνεται ότι το δίκτυο TOR δεν έχει κάποιο πρόβλημα ή κενό ασφάλειας στην υλοποίησή του. Πρόκειται για μια επίθεση η οποία βασίζεται αποκλειστικά στα μαθηματικά και πιο συγκεκριμένα στις πιθανότητες. Η πληροφορία η οποία συλλέγεται από μόνη της δεν προδίδει κανέναν στοιχείο για τη ταυτότητα του χρήστη παρά μόνο αν συσχετιστεί με άλλες πληροφορίες τότε είναι πιθανό αλλά όχι σίγουρο να έρθει ο επιτιθέμενος σε ένα συμπέρασμα σχετικά με τη ταυτότητα του χρήστη. [10][12]

3 Δίκτυο Ανωνυμίας για Αναφορά Περιστατικών

Στα πλαίσια της διπλωματικής μου επέλεξα να κατασκευάσω ένα δίκτυο ανωνυμίας βασιζόμενος στην υλοποίηση του TOR [6], για χρησιμοποίηση αυτού σε περιπτώσεις αναφοράς περιστατικών, οι οποίες σχετίζονται με την ασφάλεια του πολίτη. Τέτοιες περιπτώσεις είναι η αναφορά οποιασδήποτε εγκληματικής πράξης (εμπρησμός, φόνος, ξυλοδαρμός, δικαστικές υποθέσεις κλπ). Το δίκτυο αυτό λειτουργεί με την αποστολή συγκεκριμένων σε δομή πακέτων τα οποία είναι για αναφορά των παραπάνω περιπτώσεων. Όλοι οι κόμβοι του δικτύου είναι έμπιστοι κόμβοι οι οποίοι μπορεί να τρέχουν σε υποδομές της αστυνομίας, μη κυβερνητικών οργανώσεων κλπ και όχι σε τυχαίους χρήστες του δικτύου οι οποίοι μπορεί να είναι και κακόβουλοι.

3.1 Μέρη του Δικτύου Ανωνυμίας

Ο Client είναι η εφαρμογή που χρησιμοποιεί ο χρήστης (φυσικό πρόσωπο) για να μπορέσει να κάνει αναφορά ενός γεγονότος μέσω του δικτύου ανωνυμίας. Η εφαρμογή του χρήστη σε κάθε νέα αναφορά του χρήστη συμβουλεύεται έναν Διακομιστή Κόμβων προκειμένου να πάρει τη λίστα με όλους τους ενεργούς κόμβους του δικτύου. Στη συνέχεια μέσα από τη λίστα αυτή θα επιλέξει μια τυχαία τριάδα κόμβων από τους οποίους θα περάσει. Με την επιλογή των κόμβων, ξεκινάει η διαδικασία παραγωγής κλειδιών με στόχο ο χρήστης να μοιράζεται με κάθε κόμβο ένα μοναδικό μυστικό κλειδί. Για την παραγωγή και ανταλλαγή των κλειδιών γίνεται χρήση του Diffie-Hellman. Μόλις η διαπραγμάτευση των κλειδιών ολοκληρωθεί τότε ο Client προωθεί το πακέτο στο πρώτο κόμβο του δικτύου.

Οι κόμβοι του δικτύου ανωνυμίας είναι κόμβοι έμπιστοι οι οποίοι έχουν μοιραστεί σε έμπιστους εθελοντές οι οποίοι επιθυμούν να τρέξουν το λογισμικό για την υποστήριξη της λειτουργίας του δικτύου. Τέτοιοι εθελοντές μπορεί να είναι η αστυνομία, η πυροσβεστική ή μια Μη Κυβερνητική Οργάνωση. Με την εγκατάσταση τους οι κόμβοι ενημερώνουν τον Διακομιστή Κόμβων για την ύπαρξη τους και για πληροφορία που τους αντιπροσωπεύει όπως διεύθυνση IP , πόρτα δικτύου, το όνομα και την τοποθεσία τους. Επίσης όταν οι κόμβοι ενημερωθούν από το λογισμικό που διαχειρίζεται το δίκτυο, πως πρέπει να καταστραφούν, τότε με τη σειρά τους ενημερώνουν τον Διακομιστή Κόμβων για την διακοπή λειτουργίας τους. Οι κόμβοι δέχονται συνδέσεις από clients ή από άλλους κόμβους και η δουλειά τους είναι να προωθούν τα πακέτα που λαμβάνουν στον επόμενο κόμβο ή στον τελικό παραλήπτη. Για κάθε πακέτο που δέχονται πρέπει κάνοντας χρήση το κατάλληλο κλειδί να το αποκρυπτογραφήσουν προκειμένου να διαβάσουν τη πληροφορία που τους αφορά. Κάθε φορά που το onion περνάει από ένα κόμβο τότε αφαιρείται από το onion και ένα στρώμα κρυπτογραφημένης πληροφορίας. Ο κάθε κόμβος χρησιμοποιώντας νήματα (threads) μπορεί να δέχεται πολλές συνδέσεις ταυτόχρονα.

Οι διακομιστές κόμβων αποτελούν HTTP servers οι οποίοι δέχονται αιτήματα από τους κόμβους οι οποίοι ενεργοποιούνται και ενημερώνουν τη βάση δεδομένων με τους ενεργούς κόμβους όλου του δικτύου. Στη βάση δεδομένων τους όπως προαναφέρθηκε υπάρχουν όλες οι πληροφορίες οι οποίες αφορούν έναν κόμβο. Επίσης δέχονται αιτήματα από τους clients

προκειμένου να τους προωθήσουν μια λίστα με όλους τους διαθέσιμους κόμβους. Οι διακομιστές κόμβων είναι διαθέσιμοι σε όλους του clients.

Τέλος οι Διαχειριστές Κόμβων είναι λογισμικό το οποίο υπάρχει για να συγχρονίζει τη λειτουργία των κόμβων. Κάθε κόμβος έχει συγκεκριμένο χρόνο ζωής και με το πέρας του χρόνου αυτού ο κόμβος καταστρέφεται και τη θέση του παίρνει ένας νέος κόμβος στο δίκτυο. Οι πόροι οι οποίοι έχει διαθέσει ο εθελοντής δεν χάνονται απλά διαχειρίζονται από τους διαχειριστές κόμβων έτσι ώστε να καταστρέφεται το μηχάνημα το οποίο ήταν σε λειτουργία και τη θέση του να παίρνει ένα άλλο. Με αυτό τον τρόπο διαγράφεται όποια πληροφορία θα μπορούσε να χρησιμοποιηθεί για εξαγωγή συμπερασμάτων δρομολόγησης και ενισχύεται η διασφάλιση της ανωνυμίας του χρήστη.

3.2 Ασφάλεια στο Δίκτυο Ανωνυμίας

Στη διάρκεια της σχεδίασης του Δικτύου Ανωνυμίας προσδιορίστηκαν όλες οι απαραίτητες τεχνολογίες ασφάλειας για τη διασφάλιση της εμπιστευτικότητας, ακεραιότητας και διαθεσιμότητας πληροφορίας αλλά και της ανωνυμίας του χρήστη με τις υπηρεσίες.

Ο κάθε χρήστης και κόμβος διαθέτουν από ένα πιστοποιητικό με το οποίο πραγματοποιούν μεταξύ τους ένα ασφαλές κανάλι επικοινωνίας χρησιμοποιώντας το ασφαλές πρωτόκολλο επικοινωνίας TLS v1.3. Το πρωτόκολλο TLS μπορεί να διασφαλίσει την εμπιστευτικότητα των δεδομένων και κάνοντας χρήση συμμετρικής κρυπτογράφησης των δεδομένων μπορεί να προσφέρει αυθεντικοποίηση των μερών με χρήση των ψηφιακών πιστοποιητικών και διασφάλιση της ακεραιότητας των δεδομένων κάνοντας χρήση του μηχανισμού ελέγχου ακεραιότητας Message Authentication Code [3].

Η πληροφορία που αφορά κάθε κόμβο κρυπτογραφείται με χρήση συμμετρικής κρυπτογράφησης AES κλειδιού 128 bits. Ο client για την κρυπτογράφηση και τη δημιουργία κάθε στρώματος του onion κάνει χρήση ενός μοναδικού κλειδιού το οποίο έχει ανταλλάξει με κάθε κόμβο χρησιμοποιώντας τον αλγόριθμο ανταλλαγής κλειδιών Diffie-Hellman. Η ακεραιότητα της πληροφορίας που μεταφέρεται μεταξύ κόμβου και χρήστη προστατεύεται από το TLS. Όλα τα κλειδιά διαρκούν για τη αποστολή μιας αναφοράς και μετά καταστρέφονται. Για τη αποστολή μια νέας αναφοράς από το χρήστη επιλέγονται διαφορετικοί κόμβοι και κατ' επέκταση δημιουργούνται νέα συμμετρικά κλειδιά.

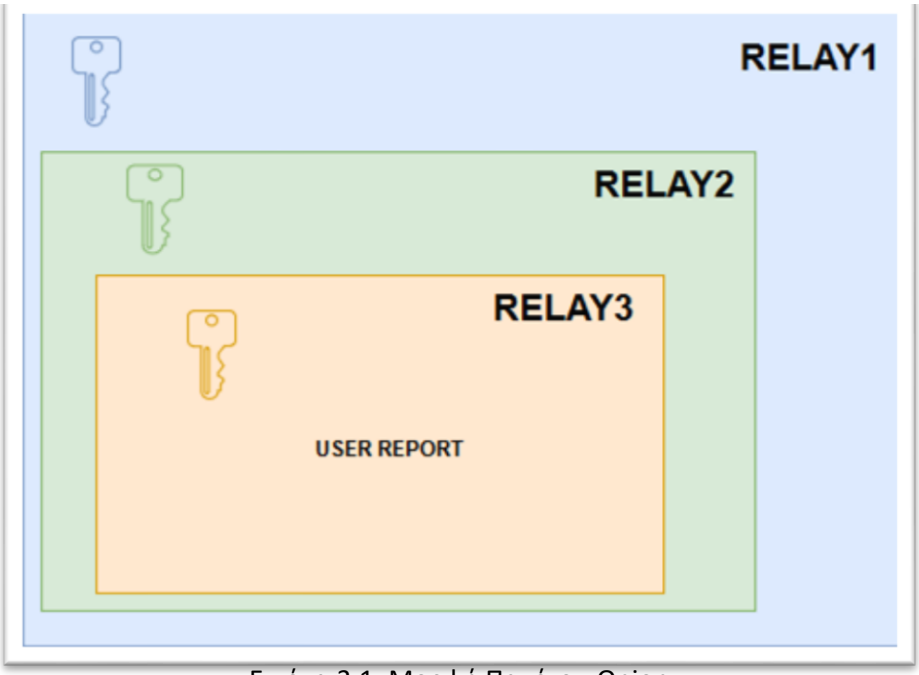
Σύμφωνα με τα παραπάνω ο κάθε κόμβος διαθέτει ένα ζεύγος κλειδιών δημόσιας κρυπτογράφησης και μια λίστα από συμμετρικά κλειδιά τα οποία έχει δημιουργήσει για να εξυπηρετήσει τους clients. Κάθε συμμετρικό κλειδί έχει σύντομο χρόνο ζωής. Αμέσως μόλις ο κόμβος λάβει ένα μήνυμα τύπου data αποκρυπτογραφεί το πακέτο και αφαιρεί από τη λίστα συμμετρικών κλειδιών το κλειδί το οποίο χρησιμοποιήθηκε. Επομένως η λίστα η οποία διατηρούν οι κόμβοι είναι μια λίστα η οποία συνεχώς μεταβάλλεται και ένα κλειδί δε κρατάει για κανέναν client πάνω από ένα ή δύο λεπτά και ποτέ το ίδιο κλειδί δε χρησιμοποιείται για

δεύτερη κρυπτογράφηση. Επίσης ο κόμβος για κάθε client που εξυπηρετεί δημιουργεί ένα μοναδικό κλειδί.

Τέλος στη συγκεκριμένη υλοποίηση για να αποτραπεί κακόβουλοι χρήστες να εξάγουν αποτελέσματα τα οποία μπορεί να σπάσουν την ανωνυμία του χρήστη μέσω παρακολούθησης και ανάλυσης του δικτύου, ο κάθε κόμβος πριν στείλει το μήνυμα στο επόμενο κόμβο ή στον τελικό παραλήπτη επιλέγει τυχαία μια καθυστέρηση πριν στείλει το πακέτο. Φυσικά η τυχαία επιλογή της καθυστέρησης του πακέτου επιλέγεται ανάμεσα σε ένα συγκεκριμένο εύρος χρόνου έτσι ώστε να μην καταστήσει τη λειτουργία του μη αποδοτική.

3.3 Δομή Πληροφορίας και Πακέτου Onion

Βασικό μέρος της υλοποίησης είναι η ύπαρξη πακέτων σε μορφή κρεμμυδιού. Για να μπορέσει ο χρήστης να στείλει ανώνυμα και με ασφάλεια την αναφορά του στον παραλήπτη τότε θα πρέπει ο κάθε κόμβος από τον οποίο περνάει το μήνυμα να μην μπορεί να δει την πληροφορία των επόμενων κόμβων. Για να επιτευχθεί αυτό ο client κατασκευάζει ένα ειδικό πακέτο το οποίο αποτελείται από τρία στρώματα κρυπτογράφησης, ένα για κάθε κόμβο. Το κάθε στρώμα περιλαμβάνει τη πληροφορία που χρειάζεται ο κόμβος για να αποκρυπτογραφήσει το πακέτο και να δει τη πληροφορία που τον αφορά. Μόλις διαβάσει τη πληροφορία που τον αφορά μεταβιβάζει τη κρυπτογραφημένη πληροφορία που βρίσκεται μέσα στο πακέτο του στον επόμενο χρήστη ή στο τελικό παραλήπτη αν είναι ο τρίτος κόμβος.



Εικόνα 3.1: Μορφή Πακέτου Onion

Η μορφή στην οποία μεταφέρεται το πακέτο είναι JSON και με συγκεκριμένα πεδία. Το πακέτο αποτελείται από τη πληροφορία δρομολόγησης και ελέγχου τα οποία αποτελούν τη κεφαλίδα του κάθε πακέτου και στο τέλος υπάρχει ένα πεδίο “data” στο οποίο υπάρχει η

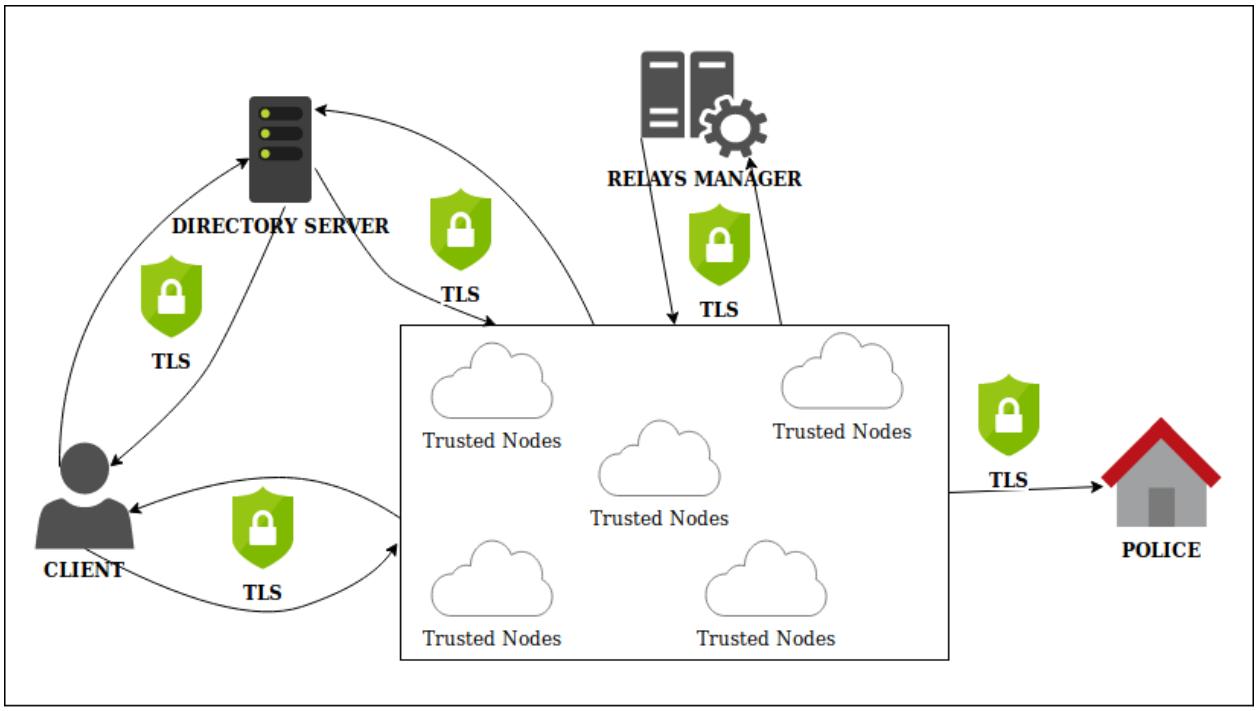
κρυπτογραφημένη πληροφορία για τον επόμενο κόμβο ή η αναφορά του χρήστη στη περίπτωση του τρίτου κόμβου. Πιο συγκεκριμένα οι πληροφορίες που υπάρχουν στο πακέτο είναι οι εξής:

- **Cmd:** Αφορά το είδος πακέτου το οποίο μεταφέρεται. Στην συγκεκριμένη υλοποίηση υπάρχουν πέντε διαφορετικοί τύποι πακέτων. Οι τύποι αυτοί είναι create για να ενημερώσουμε το κόμβο ότι πρέπει να κατασκευάσει ένα συμμετρικό κλειδί για τον client, created για να ενημερώσει ο κόμβος τον client ότι το κλειδί δημιουργήθηκε, extend για να ενημερώσουμε το κόμβο ότι πρέπει να προωθήσει πακέτο κατασκευής κλειδιού στον επόμενο κόμβο, extended για μεταβίβαση πακέτου κλειδιού πίσω στο χρήστη και data για την αποστολή πακέτου κρυπτογραφημένου το οποίο περιέχει την αναφορά του χρήστη.
- **Relay:** Είναι λίστα στην οποία εμπεριέχεται ο επόμενο κόμβος (ip και πόρτα).
- **Id:** Η τιμή του ID αναφέρεται στο μοναδικό αναγνωριστικό που έχει δώσει ο κόμβος στο κλειδί κρυπτογράφησης το οποίο έφτιαξε για τον χρήστη.
- **Data:** Στο πεδίο αυτό υπάρχει είτε το κρυπτογραφημένο πακέτο του επόμενου κόμβο ή στη περίπτωση του πρώτου στρώματος (πακέτο τρίτου κόμβου) την αναφορά του χρήστη.
- **Exit:** Το πεδίο αυτό παίρνει δύο τιμές, τις True ή False. Το True σημαίνει πως έχει φτάσει στο τρίτο κόμβο και αυτός πρέπει να το μεταβιβάσει στην αρχή προστασίας την οποία έχει το πακέτο, ενώ στη περίπτωση του False ο κόμβος πρέπει να μεταβιβάσει τη πληροφορία στον επόμενο κόμβο.

Για να μπορέσει να κατασκευαστεί το πακέτο το λογισμικό του χρήστη τρέχει αναδρομικά τη συνάρτηση κατασκευής πακέτου, κρυπτογραφώντας σε κάθε επανάληψη με το αντίστοιχο κλειδί κάθε κόμβου. Μόλις η συνάρτηση ολοκληρώσει τρεις επαναλήψεις τότε έχει κατασκευαστεί ένα onion τριών κρυπτογραφημένων στρωμάτων το οποίο θα στείλει στο πρώτο κόμβο του κυκλώματος, ο δεύτερος στο τρίτο και τελικά ο τρίτος στην αρχή προστασίας για την οποία προορίζεται.

3.4 Αρχιτεκτονική Δικτύου Ανωνυμίας

Στο δίκτυο υπάρχουν οι χρήστες (Clients) οι οποίοι τρέχουν το λογισμικό του χρήστη προκειμένου να μπορούν να συνδεθούν στο δίκτυο ανωνυμίας και να μεταφέρουν μηνύματα προς τις αρχές προστασίας του πολίτη. Το μήνυμα που θα στείλει ο χρήστης θα περάσει από τρεις έμπιστους κόμβους (Onion Relays) του δικτύου ανωνυμίας και ο τρίτος κόμβος θα μεταβιβάσει το πακέτο στην αντίστοιχη αρχή προστασίας. Ο κάθε Client μαθαίνει για όλους τους διαθέσιμους κόμβους του δικτύου από τους Διακομιστές Κόμβων οι οποίοι αναλαμβάνουν να στέλνουν μια λίστα με τους διαθέσιμους κόμβους του δικτύου σε όποιον Client το ζητήσει. Τέλος το δίκτυο ανωνυμίας διαχειρίζονται οι Διαχειριστές Κόμβων (Relay Managers) οι οποίοι συγχρονίζουν και όλο το δίκτυο.

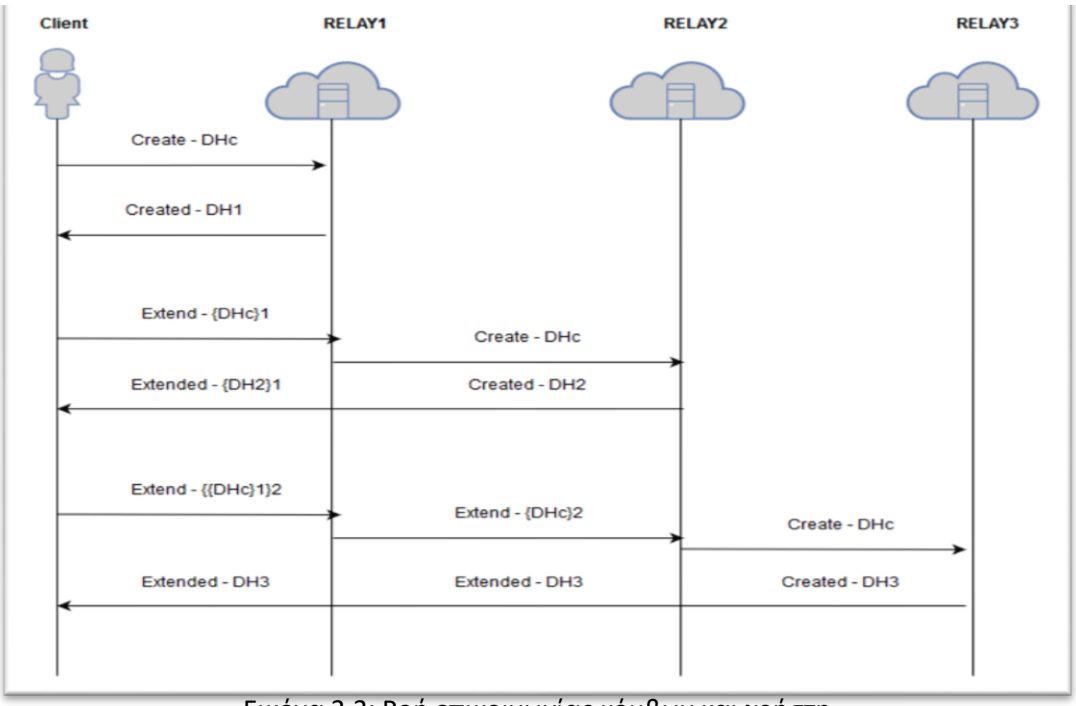


Εικόνα 3.2: Αρχιτεκτονική Δικτύου Ανωνυμίας

Για να λειτουργήσει το δίκτυο πρέπει όλα τα μέρη του να είναι διαθέσιμα και να λειτουργούν σωστά. Η συγκεκριμένη υλοποίηση όπως φαίνεται στην εικόνα 3.2 είναι μονόδρομη καθώς επιτρέπει μόνο την αποστολή μιας αναφοράς από τον χρήστη και όχι την επικοινωνία του παραλήπτη (αστυνομία) με το χρήστη. Πιο συγκεκριμένα όταν ο χρήστης στείλει ένα μήνυμα στην αρχή προστασίας που επιθυμεί τότε όλες οι συνδέσεις κλείνουν και δεν διατηρείται καμία πληροφορία δρομολόγησης σε κανέναν από τους κόμβους από τους οποίους έχει περάσει. Αυτό σε συνδυασμό με την ύπαρξη έμπιστων κόμβων εγγυάται στο χρήστη τη πλήρη ανωνυμία του και την μη επικοινωνία μιας αρχής προστασίας με αυτόν.

Στην εικόνα 3.3 φαίνεται αναλυτικά η ροή επικοινωνίας μεταξύ των μερών του δικτύου ανωνυμίας για την αποστολή αναφοράς ενός χρήστη σε μια αρχή προστασίας. Ο client επικοινωνεί με τη σειρά που έχει επιλέξει τους τρεις κόμβους ώστε να δημιουργήσει με το κάθε ένα το συμμετρικό κλειδί για τη κρυπτογράφηση του μηνύματος. Όπως προαναφέρθηκε το δίκτυο κάνει χρήση του Diffie-Hellman για την ανταλλαγή κλειδιών. Ο client στέλνει στο πρώτο κόμβο το δημόσιο κλειδί που έχει υπολογίσει γι' αυτόν και στη συνέχεια ο κόμβος του απαντάει με το δικό του δημόσιο κλειδί. Με την ανταλλαγή δημοσίων κλειδιών και παραμέτρων κατασκευάζουν το συμμετρικό κλειδί που θα μοιράζονται για την επικοινωνία τους. Η ροή συνεχίζεται στο δεύτερο κόμβο. Ο client καλείται να δημιουργήσει ένα πιο σύνθετο πακέτο το οποίο αποτελεί ένα onion το οποίο θα περνάει από τον πρώτο κόμβο ως πακέτο extend και θα το προωθεί στον δεύτερο κόμβο ως create ώστε να δημιουργήσει ο δεύτερος το δικό του δημόσιο κλειδί το οποίο θα επιστρέψει στον client μέσω του πρώτου κόμβου. Η φάση κατασκευής κλειδιών θα ολοκληρωθεί στέλνοντας με τον ίδιο τρόπο ένα πακέτο create σε

αυτόν ώστε να απαντήσει με το δικό του δημόσιο κλειδί και ο client να ολοκληρώσει με αυτόν τον τρόπο τη λίστα συμμετρικών κλειδιών που ανταλλάσει με τους κόμβους που τυχαία έχει



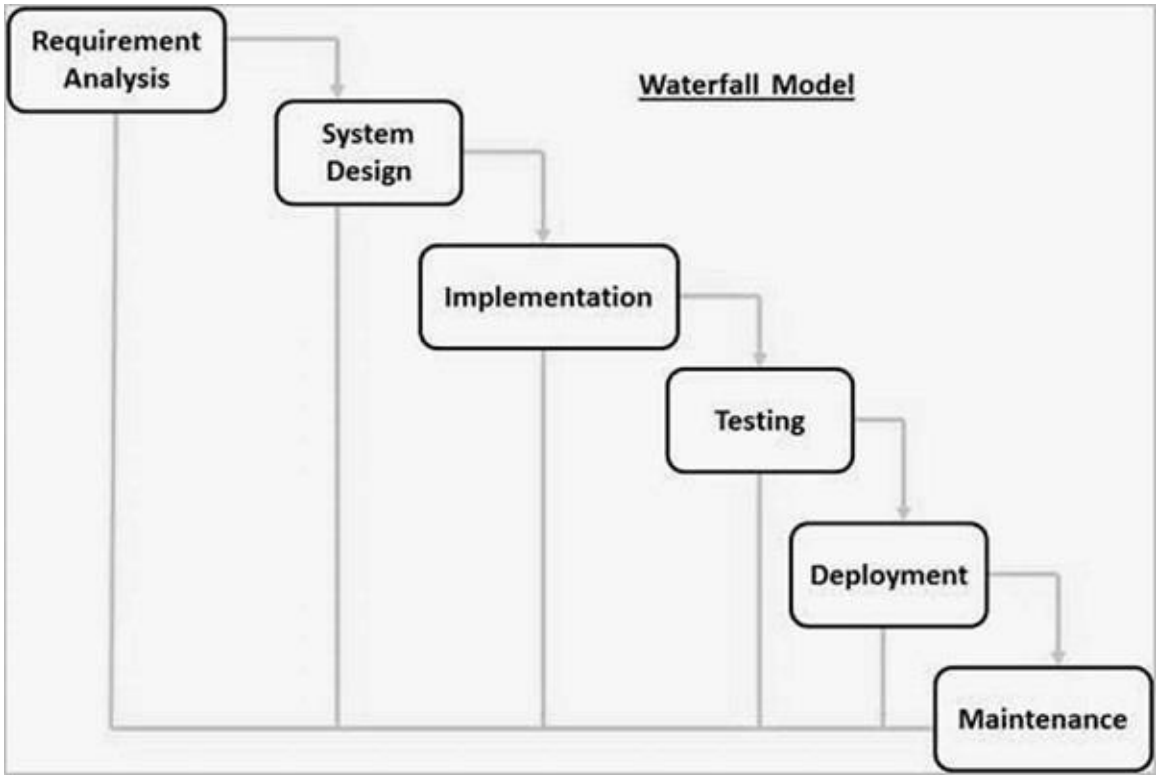
Εικόνα 3.3: Ροή επικοινωνίας κόμβων και χρήστη

επιλέξει. Με τη συμπλήρωση της λίστας συμμετρικών κλειδιών ο client κατασκευάζει το κρυπτογραφημένο onion το οποίο θα μεταφέρει την αναφορά του χρήστη μέσω των κόμβων στο τελικό παραλήπτη. Όπως φαίνεται και στην εικόνα 3.2 ο client δεν έρχεται ποτέ σε άμεση επικοινωνία με τους κόμβους 2 και 3 και με το παραλήπτη αλλά επικοινωνεί μόνο με το κόμβο 1 και αυτός είναι λόγος για τον οποίο ο παραλήπτης δε μπορεί να μάθει ποτέ ποιος είναι ο αποστολέας καθώς δεν το γνωρίζει ο κόμβος 3 και ο κόμβος 2 και ο κόμβος 3 δεν γνωρίζει το κόμβο 1 με αποτέλεσμα να χάνονται τα ίχνη δρομολόγησης.

4. Μεθοδολογία και Αποτελέσματα

4.1 Μεθοδολογία

Για την κατασκευή του λογισμικού χρησιμοποιήθηκε η μεθοδολογία του καταρράκτη (waterfall) η οποία αποτελείται από μια σειρά ενεργειών για την κατασκευή ενός λογισμικού. Στη μεθοδολογία αυτή για να ξεκινήσει μια ενέργεια του καταρράκτη θα πρέπει να έχει ολοκληρωθεί πλήρως η προηγούμενη της. Οι ενέργειες από τις οποίες αποτελείται το μοντέλο του καταρράκτη είναι έξι, η ανάλυση απαιτήσεων, ο σχεδιασμός, η κατασκευή, οι δοκιμές της κατασκευής η εγκατάσταση και τέλος η συντήρηση του λογισμικού. Τα στάδια της κατασκευής και δοκιμής μπορούν να επαναληφθούν δημιουργώντας ένα μεγαλύτερο καταρράκτη ενεργειών.



Εικόνα 4.1: Μοντέλο Καταρράκτη [11]

Σύμφωνα με τα παραπάνω στη διάρκεια κατασκευής λογισμικού πέρασα από τις φάσεις ανάλυσης απαιτήσεων, σχεδίασης, κατασκευής δοκιμής και εγκατάστασης του λογισμικού. Σε όλη τη διάρκεια κατασκευής υπήρξαν επαναλήψεις των φάσεων κατασκευής και δοκιμής. Η επανάληψη αυτών των ενεργειών μου έδωσε τη δυνατότητα να διορθώσω και να βελτιστοποιήσω το κώδικα μου και επίσης να διορθώσω όποια κενά ασφάλειας δημιουργούσαν οι αλλαγές στο κώδικα.

Στη φάση ανάλυσης απαιτήσεων καθόρισα ενδεχόμενο κόστος το οποίο μπορεί να χρειαζόταν για την κατασκευή του λογισμικού για αγορά εξειδικευμένων εργαλείων ή λογισμικού. Στο

παρόν λογισμικό χρησιμοποιήθηκαν εργαλεία και βιβλιοθήκες ανοιχτού κώδικα τα οποία δεν απαιτούσαν κάποιο κόστος. Πιο συγκεκριμένα για τη κατασκευή του δικτύου ανωνυμίας χρησιμοποίησα τη γλώσσα προγραμματισμού python και το περιβάλλον ανάπτυξης λογισμικού pycharm. Για τη στατική ανάλυση κώδικα κατέληξα στο εργαλείο ανοιχτού κώδικα pylint. Οι λόγοι για τους οποίους κατέληξα στη επιλογή της γλώσσας python είναι

- γρήγορη ανάπτυξη κώδικα
- απλή δομή της γλώσσας
- πληθώρα βιβλιοθηκών
- ικανή για την ανάπτυξη του συγκεκριμένου λογισμικού
- έμπιστες βιβλιοθήκες κρυπτογράφησης

Στην επιλογή του εργαλείου pylint κατέληξα γιατί είναι ανοιχτού κώδικα, είναι συμβατό με το πρόγραμμα ανάπτυξης κώδικα pycharm και είναι το ευρέως διαδεδομένο και δοκιμασμένο από τη κοινότητα προγραμματιστών python.

Μόλις προσδιόρισα όλες τις απαιτήσεις του λογισμικού συνέχισα στη σχεδίαση του. Στη φάση της σχεδίασης σχεδίασα την αρχιτεκτονική του συστήματος που ήθελα να φτιάξω βασιζόμενος πάντα στο TOR [6], και μέσα από τη διαδικασία της σχεδίασης προσδιόρισα επιπλέον απαιτήσεις τις οποίες χρειαζόταν το δίκτυο ανωνυμίας. Σημαντικό μέρος της σχεδίασης ήταν ο προσδιορισμός της ασφάλειας του δικτύου το οποίο σχεδίασα και κατασκεύασα. Η ασφάλεια η οποία επέλεξα να έχει το σύστημα μου είναι η εξής:

- εμπιστευτικότητα των δεδομένων
- ακεραιότητα των δεδομένων
- διασφάλιση της ανωνυμίας

Οι τεχνολογίες και μέθοδοι που χρησιμοποίησα για να το καταφέρω αυτό περιγράφονται αναλυτικά στο κεφάλαιο 3.

4.2 Αποτελέσματα

Σε αυτή την ενότητα θα παρουσιαστούν και θα σχολιαστούν ευρήματα και κομμάτια κώδικα από το δίκτυο το οποίο κατασκεύασα. Το δίκτυο στη μορφή που βρίσκεται τώρα είναι Proof of Concept (PoC) και μπορεί να μεταβιβάσει ένα πακέτο μέσα από το δίκτυο αποδεικνύοντας την ύπαρξη κρυπτογραφίας και την ύπαρξη πακέτων onion τα οποία προστατεύουν τη ταυτότητα του χρήστη.

Το δίκτυο βασίζεται στη κρυπτογραφία για να πετύχει τους στόχους ασφάλειας τους οποίους έθεσα κατά τη σχεδίαση. Το δίκτυο κάνει χρήση συμμετρικής και ασύμμετρης

κρυπτογράφησης. Κάθε μέρος του δικτύου διατηρεί ένα μόνιμο κλειδί, και κάθε χρήστης ένα μοναδικό προσωρινό κλειδί το οποίο μοιράζεται με το κάθε κόμβο. Για να δημιουργηθούν τα κλειδιά μιας χρήσης χρησιμοποιείται ο αλγόριθμος Diffie Hellman για την ανταλλαγή αυτών όπως φαίνεται στην εικόνα 4.2.

```
on_proxy.py

def diffie_hellman(public_b=None, id=None):
    ##########################################################################################
    #                                                                                        #
    #                    ****Diffie-Hellman Symmetric key exchange****                       #
    #                                                                                        #
    #                        q = large prime number                                          #
    #                        random_a = random number < q (secret)                           #
    #                        g = static number = 2                                           #
    #                        public_a = (g**random_a) mod q                                  #
    #                        random_b = random integer for b < q (secret)                    #
    #                        key_a = (public_b**random_a) mod q - must be equal to key_b     #
    #                        key_b = (public_a**random_b) mod q - must be equal to key_a     #
    #                                                                                        #
    ##########################################################################################

    if public_b:
        # calculate the key
        key = pow(public_b, random_a, q)
        session_key = hashlib.sha256(str(key).encode()).hexdigest()[:32]

        keys.append({'id': id,
                     'key': session_key})
        return
    else:
        # calculate the public_a
        public_a = pow(g, random_a, q)

        return [str(public_a), str(g), str(q)]

    return
```

Εικόνα 4.2: Κώδικας για Diffie Hellman

Στην συνάρτηση για το Diffie Hellman γίνονται όλοι οι υπολογισμοί για να δημιουργηθεί ένα συμμετρικό κλειδί και καταχωρείται και το ID το οποίο επιστρέφεται κάθε φορά από το κόμβο με το οποίο επικοινωνεί άμεσα ή έμμεσα ο χρήστης. Στη συνάρτηση του Diffie-Hellman φαίνεται όλη η διαδικασία και οι μεταβλητές οι οποίες χρειάζονται για να ολοκληρωθεί οι παραγωγή κλειδιού. Στο κώδικα του onion-proxy η συνάρτηση του Diffie-Hellman έχει δύο μέρη καθώς το δεύτερο είναι για τη παραγωγή του κλειδιού μόλις λάβει τη κατάλληλη παράμετρο που έχει υπολογίσει ο κόμβος με βάση τις παραμέτρους που έστειλε νωρίτερα ο onion-proxy. Στην εικόνα 4.3 φαίνεται το παραγόμενο κλειδί της συνάρτησης Diffie-Hellman.

```
[
{'id': '884b9567-728c-420a-bd54-b51cd0d9105c', 'key':
  '662c0c7687ca76437e40fd82429b23e3'},
{'id': '1082d2df-2536-4d0f-a34e-310595a25758', 'key':
  'd55c8f8f7f5df080f33d73d5eb4fccf2'},
{'id': '03ee98ca-ab02-4ff3-8fd4-a4131e096787', 'key':
  '6e69d6417e1808bd2eb12b511b80924f'}
]
```

Εικόνα 4.3: Κλειδιά μιας χρήσης (Diffie-Hellman)

Στην εικόνα 4.3 εκτός από τα κλειδιά βλέπουμε ότι το κάθε κλειδί έχει και ένα ID. Το ID αυτό είναι το ID το οποίο δημιουργεί ο κάθε κόμβος για κάθε χρήστη που εξυπηρετεί, όπως περιγράφηκε στο κεφάλαιο 3. Η λίστα που φαίνεται στην εικόνα 4.3 διατηρείται από τον onion-proxy μέχρι να σταλεί μια αναφορά και μετά καταστρέφεται και τη θέση της παίρνει μια νέα λίστα με νέους κόμβους και νέα κλειδιά.

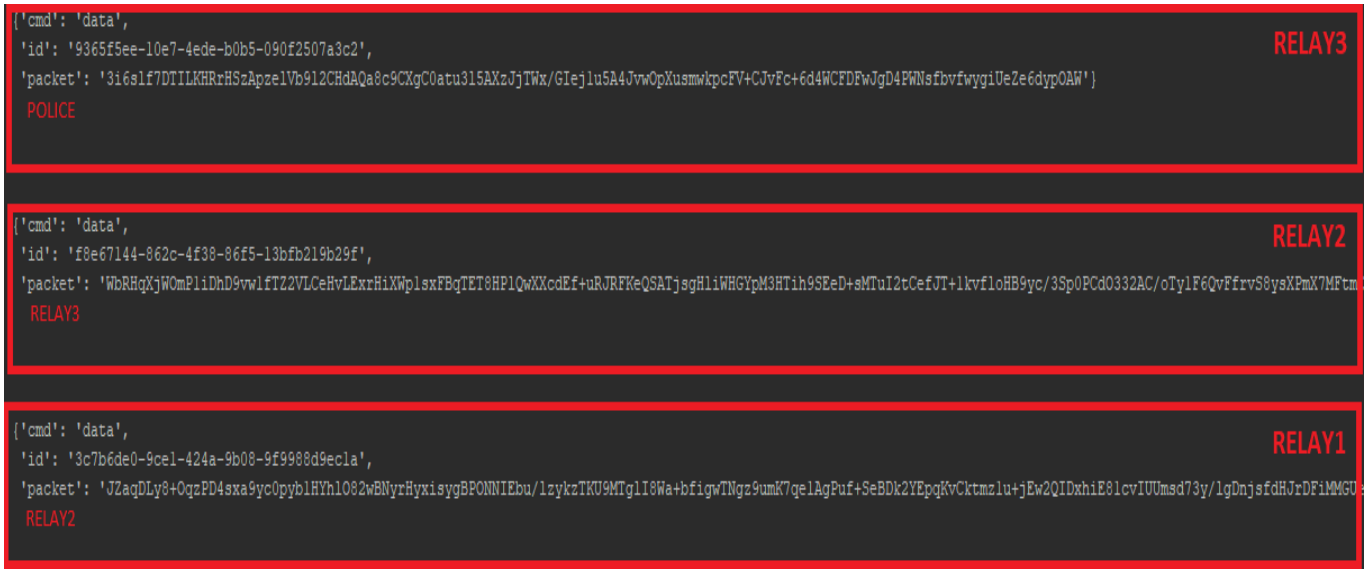
Μόλις δημιουργηθούν όλα τα κλειδιά από όλους τους κόμβους οι οποίοι έχουν επιλεγεί τότε η συνάρτηση δημιουργίας πακέτου onion ξεκινάει να κατασκευάζει το πακέτο onion τρέχοντας αναδρομικά η συνάρτηση όπως φαίνεται στην εικόνα 4.4

```
elif cmd == 'data':
    keys_len = len(keys) - 1
    key = keys[num]['key']
    id = keys[num]['id']

    json_data['cmd'] = cmd
    json_data['id'] = str(id)
    json_data['packet'] = {
        'relay': [hops[(num + 1)]['host'], str(hops[(num + 1)]['port'])],
        'exit': exit,
        'data': data
    }
    #print (key)
    encryption_suite = AES.new(key, AES.MODE_CFB, IV)
    json_data['packet'] = json.dumps(json_data['packet'])
    cipher_text = encryption_suite.encrypt(json_data['packet'])
    json_data['packet'] = base64.b64encode(cipher_text)
    #print json_data
    if num == 0:
        return json.dumps(json_data)
    else:
        return create_packet(num - 1, data=json.dumps(json_data), cmd="data")

return
```

Recursion

Εικόνα 4.4: Αναδρομική κλήση της συνάρτησης πακέτου

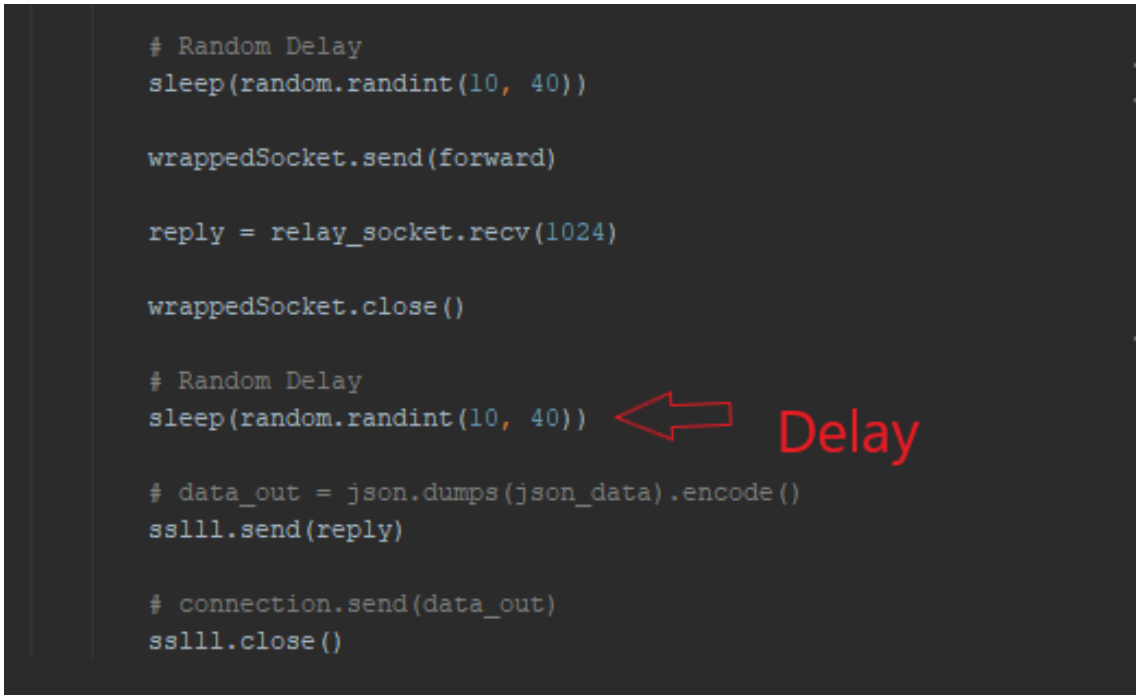
Στα πακέτα που κατασκευάζει υπάρχουν μέσα τα πεδία cmd που είναι ο τύπος του πακέτου, το ID που είναι για το κόμβο, ώστε να ξέρει ποιο κλειδί να χρησιμοποιήσει για να μπορέσει να αποκρυπτογραφήσει το πεδίο packet που έχει τη κρυπτογραφημένη πληροφορία που τον αφορά (στοιχεία δρομολόγησης και το πακέτο του επόμενου). Στην εικόνα 4.5 φαίνονται τα πακέτα που έχουν δημιουργηθεί για το κάθε κόμβο.



Εικόνα 4.5: Πακέτα data κάθε κόμβου

Στην εικόνα 4.5 στο πεδίο packet φαίνεται η κρυπτογραφημένη πληροφορία με AES για κάθε επόμενο κόμβο. Κάθε κρυπτογράφημα έχει κωδικοποιηθεί σε base64 για να μπορεί να γίνει η μεταφορά του μέσω sockets. Γι' αυτό το λόγο κάθε κόμβος πριν αποκρυπτογραφήσει τη πληροφορία που τον αφορά πρέπει να αποκωδικοποιεί από base64.

Στην εικόνα 4.6 φαίνεται στο κώδικα η ύπαρξη καθυστέρησης αποστολής ενός πακέτου για να εμποδίσω κακόβουλους χρήστες να εξάγουν συμπεράσματα από την ανάλυσης κίνησης πακέτων του δικτύου. Καθυστέρηση υπάρχει και στις απαντήσεις που στέλνει ο κόμβος από το αίτημα ενός άλλου κόμβου ή από ένα χρήστη. Η καθυστέρηση όπως φαίνεται και στο κώδικα γεννάται από τυχαία συνάρτηση ακεραίων, από τα δέκα δευτερόλεπτα έως και τα σαράντα για λόγους δοκιμών. Το εύρος επιλογής καθυστέρησης μπορεί να μεγαλώσει για να κάνει πιο πολύπλοκη και δύσκολη την ανάλυση με σεβασμό όμως στην απόδοση του δικτύου αλλά και την εμπειρία του χρήστη.




```
# Random Delay
sleep(random.randint(10, 40))

wrappedSocket.send(forward)

reply = relay_socket.recv(1024)

wrappedSocket.close()

# Random Delay
sleep(random.randint(10, 40))

# data_out = json.dumps(json_data).encode()
sslll.send(reply)

# connection.send(data_out)
sslll.close()
```

Εικόνα 4.6: Καθυστέρηση αποστολής πακέτων

Τοπικά στον υπολογιστή δοκιμάστηκαν τρεις κόμβοι και ένας χρήστη ώστε να ελεγχθεί η καθυστέρηση των πακέτων. Για να κάνω τη δοκιμή αυτή, εκτέλεσα πέντε φορές την αποστολή μιας αναφοράς από ένα χρήστη ώστε να περάσει από τους κόμβους και να δω τις διαφορές στο χρόνο ανάμεσα στις πέντε εκτελέσεις. Για να το καταφέρω αυτό παραμετροποίησα το κώδικα κάθε κόμβου έτσι ώστε σε κάθε μήνυμα που δέχεται να εμφανίζει τη χρονική στιγμή που το δέχθηκε. Στη συνέχεια συσχετίζοντας του χρόνους σε κάθε κόμβο έβγαλα συμπεράσματα για τη καθυστέρηση των πακέτων. Στο πίνακα 4.1 φαίνονται τα αποτελέσματα αυτής της δοκιμής, από τα οποία συμπεραίνεται ότι η καθυστέρηση αποστολής πακέτων από κάθε κόμβο λειτουργεί ορθά. Στο πίνακα φαίνονται οι χρόνοι αποστολής πακέτου από το κόμβο 1 στο κόμβο 2 και από το κόμβο 2 στο κόμβο 3 στις πέντε εκτελέσεις που έγινε. Οι χρόνοι οι οποίοι υπολογίστηκαν και παρουσιάζονται είναι για πακέτα Data, καθυστέρηση όμως υπάρχει σε όλους τους τύπους πακέτων.

A/E	**Relay1 → Relay2**	**Relay2 → Relay3**	**Relay1 → Relay3**
1	34 sec	16 sec	50 sec
2	10 sec	38 sec	48 sec
3	13 sec	19 sec	32 sec

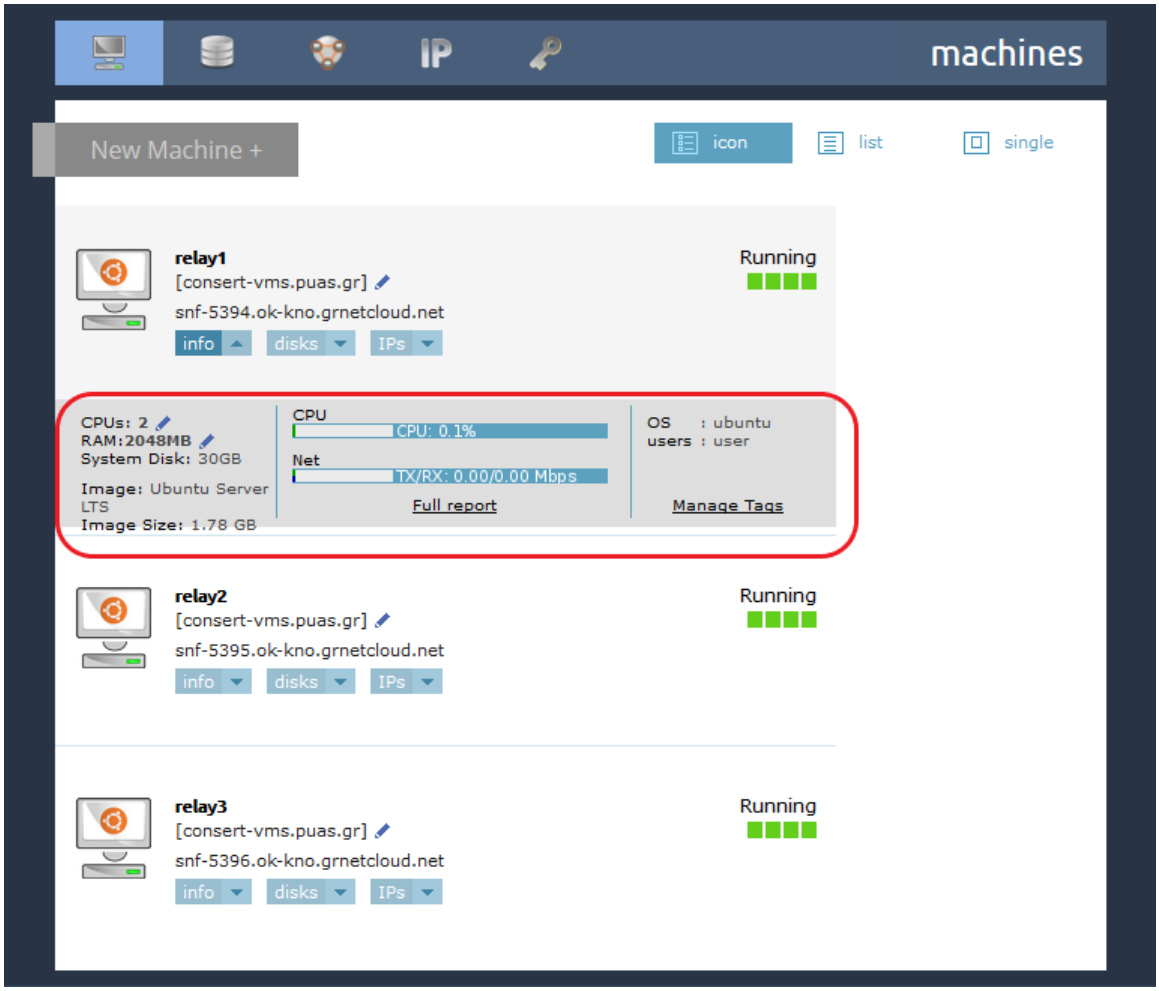
4	33 sec	17 sec	50 sec
5	21 sec	11 sec	33 sec

Πίνακας 4.1: Χρόνοι αποστολής πακέτων με καθυστέρηση

Όπως φαίνεται στο πίνακα 4.1 υπάρχει μεταβολή στο χρόνο αποστολής πακέτων η οποία γίνεται πολύ πιο σύνθετη αν λάβουμε υπόψη και τα πακέτα τα οποία στέλνονται κατά τη διαπραγμάτευση πακέτων. Το εύρος του χρόνου καθυστέρησης μπορεί να μεγαλώσει για να γίνει πιο σύνθετη και δύσκολη η ανάλυση κίνησης των πακέτων. Για να υπολογιστεί σωστά το εύρος θα πρέπει να λάβουμε υπόψη το ανώτερο όριο του καθώς είναι και αυτό το οποίο μπορεί να δημιουργήσει πρόβλημα στο λειτουργία και απόδοση του δικτύου.

Τέλος για να δοκιμάσω το πειραματικό δίκτυο που έφτιαξα σε πραγματικές συνθήκες εγκατέστησα τρεις κόμβους στον okeano. Για να στηθεί κάθε κόμβος έκανα χρήση τριών εικονικών μηχανών με τις παρακάτω δυνατότητες:

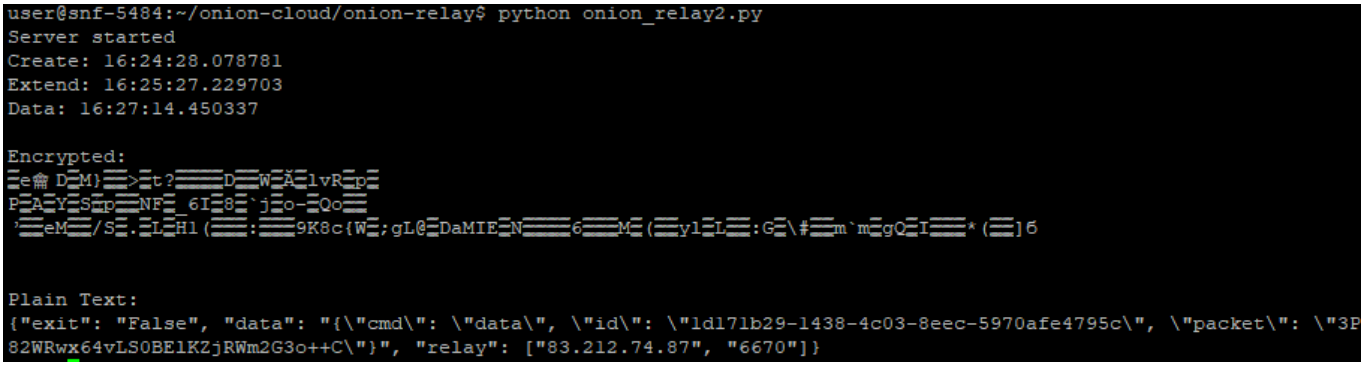
- CPU: 2
- RAM: 2 GB
- System Disk: 30GB
- OS: Ubuntu Server



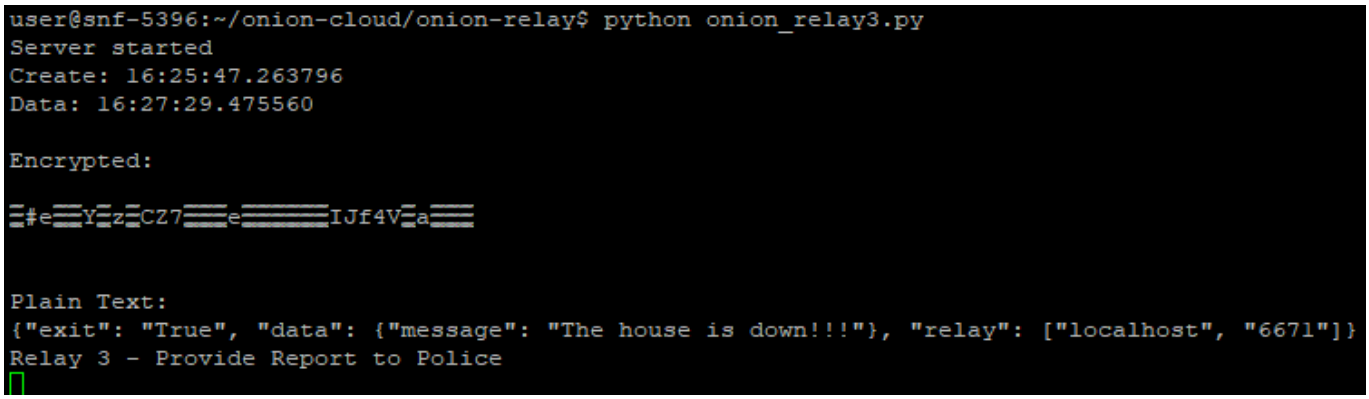
Εικόνα 4.7: Κόμβοι του δικτύου εγκατεστημένοι στον Okeanos

Στόχος αυτού του πειράματος είναι ένα πακέτο το οποίο θα περιέχει μια αναφορά να ξεκινήσει από το προσωπικό μου υπολογιστή και να φτάσει μέχρι και το τρίτο κόμβο ο οποίος βρίσκεται στον Okeanos. Κατά τη διάρκεια της αποστολής κάθε φορά που περνάει από ένα κόμβο γίνεται εκτύπωση στο τερματικό του κόμβου η ώρα που δέχθηκε ένα πακέτο data το κρυπτογραφημένο πακέτο και από κάτω το πακέτο αποκρυπτογραφημένο.

Εικόνα 4.8: Πακέτο Data στο πρώτο κόμβο του δικτύου (Okeanos)



Εικόνα 4.9: Πακέτο Data στο δεύτερο κόμβο του δικτύου (Okeanos)



Εικόνα 4.10: Πακέτο Data στο τρίτο κόμβο του δικτύου (Okeanos)

Στις εικόνες 4.8, 4.9 και 4.10 φαίνονται οι τρεις κόμβοι από τους οποίους περνάει ένα πακέτο στο δίκτυο. Οι κόμβοι όπως προαναφέρθηκε είναι στο Okeano και έχουν public IPs. Και στις τρεις εικόνες έχει εκτυπωθεί το πακέτο σε δύο διαφορετικές μορφές. Η πρώτη είναι κρυπτογραφημένο και αποκωδικοποιημένο από base64 γι' αυτό το λόγο υπάρχουν οι περίεργοι χαρακτήρες που φαίνονται στις εικόνες. Η δεύτερη μορφή είναι αποκρυπτογραφημένα το πακέτο. Στις πρώτες δύο εικόνες στο πακέτο υπάρχει ένα πεδίο packet στο οποίο υπάρχει

κρυπτογραφημένη η πληροφορία που πρέπει να σταλεί στον επόμενο κόμβο. Πιο συγκεκριμένα το πεδίο packet που φαίνεται στην εικόνα 4.8 είναι ολόκληρο το πακέτο που φαίνεται στην εικόνα 4.9, δηλαδή στο δεύτερο κόμβο. Πιο αναλυτικά οι πληροφορίες που βλέπουμε στις εικόνες είναι exit το οποίο false στις εικόνες 4.8 και 4.9 και true στην 4.10 καθώς είναι ο τρίτος κόμβος από τον οποίο το πακέτο βγαίνει από το δίκτυο ανωνυμίας και η αναφορά στέλνεται σε κάποια αρχή προστασίας. Στη συνέχεια βλέπουμε τα id τα οποία είναι τα μοναδικά αναγνωριστικά των προσωρινών κλειδιών κρυπτογράφησης τα οποία πρέπει κάθε κόμβος να μεταβιβάζει στον επόμενο μη κρυπτογραφημένα ώστε να μπορέσει ο επόμενος κόμβος να βρει το κλειδί που χρειάζεται για να αποκρυπτογραφήσει τη πληροφορία που τον αφορά. Γι' αυτό το id που φαίνεται στην εικόνα 4.8 είναι αυτό που πρέπει να στείλει στο δεύτερο κόμβο για να αποκρυπτογραφήσει το πακέτο του. Οι άλλες πληροφορίες είναι το cmd που δείχνει στο κόμβο το τύπο πακέτου ώστε να ξέρει πώς να το διαχειριστεί και το relay το οποίο δείχνει στο κόμβο τον επόμενο κόμβο στον οποίο πρέπει να στείλει τη κρυπτογραφημένη πληροφορία του πακέτου του (packet).

Στις τρεις εικόνες των κόμβων 4.8, 4.9 και 4.10 παρατηρούμε ότι σε κάθε επόμενο κόμβο το κρυπτογράφημα μικραίνει. Αυτό συμβαίνει γιατί σε κάθε κόμβο αφαιρείται και ένα στρώμα πληροφορίας. Επίσης στην εικόνα του τελευταίου κόμβου 4.10 το πακέτο έχει πιο απλή δομή καθώς δεν υπάρχει επόμενος κόμβος στο δίκτυο ώστε να μεταβιβάσει id και κρυπτογραφημένη πληροφορία. Φυσικά το κανάλι ανάμεσα στο τρίτο κόμβο και στην αντίστοιχη αρχή προστασίας θα είναι προστατευμένο με TLS. Τέλος στις εικόνες των πακέτων φαίνονται η χρόνοι στους οποίους οι κόμβοι έλαβαν πακέτο. Από αυτό μπορούμε να καταλάβουμε τη καθυστέρηση που σχολιάστηκε προηγουμένως καθώς και την αρχιτεκτονική η οποία έχει παρουσιαστεί στο τρίτο κεφάλαιο. Πιο αναλυτικά φαίνεται στο πρώτο κόμβο (εικόνα 4.8) ότι ο κόμβος δέχθηκε τέσσερα πακέτα, τα οποία είναι ένα create δύο extend και ένα data. Τα πακέτα create και data όπως φαίνεται τα δέχονται όλοι ο κόμβοι στο ίδιο πλήθος, δηλαδή από ένα. Τα extend όμως διαφέρουν καθώς ο πρώτος κόμβος έχει να μεταβιβάσει πακέτα δημιουργίας κλειδιών για τους δύο επόμενους κόμβους, ο δεύτερος κόμβος έχει να μεταβιβάσει μόνο το πακέτο δημιουργίας κλειδιού του τρίτου ενώ ο τρίτος κόμβος δεν έχει κανένα καθώς είναι ο τελευταίος.

5 Επίλογος

Στη παρούσα διπλωματική ασχολήθηκα με την κατασκευή ενός δικτύου το οποίο παρέχει ανωνυμία στους χρήστες του, για αποστολή αναφορών περιστατικών, σχετικών με την ασφάλεια του πολίτη και κάνοντας χρήση τεχνολογιών Onion Routing. Μελέτησα και ανέφερα τους βασικούς κινδύνους τέτοιων τεχνολογιών καθώς και το τρόπο με τον οποίο λειτουργούν τέτοια δίκτυα. Στη συνέχεια βασιζόμενος σε αυτές τις τεχνολογίες σχεδίασα και κατασκεύασα ένα δικό μου δίκτυο ανωνυμίας στα πρότυπα του TOR το οποίο είναι πιο ελαφρύ από το TOR και πολύ πιο ελεγχόμενο τόσο στη λειτουργία όσο και στη στήριξη του από εθελοντές.

Με την ολοκλήρωση της μελέτης τεχνολογιών ανώνυμης δικτύωσης ξεκίνησα να σχεδιάζω μια λύση η οποία θα είναι ικανή να καλύψει τις ανάγκες υπηρεσιών έκτακτης ανάγκης για αναφορές των χρηστών για διάφορα περιστατικά. Το δίκτυο το οποίο σχεδίασα το έκανα ώστε να μπορεί να εξασφαλίσει στο χρήστη την αίσθηση σιγουριάς και ασφάλειας προς την αναφορά του για κάποιο συμβάν. Στόχος μου στη σχεδίαση και κατασκευή ενός τέτοιο δικτύου δεν ήταν μόνο να δώσω την αίσθηση ασφάλειας και ανωνυμίας στο χρήστη αλλά και έμπρακτα να διασφαλίσω την εμπιστευτικότητα της αναφοράς του και την ακεραιότητα αυτής κάνοντας χρήση όλων των κατάλληλων μηχανισμών οι οποίοι περιγράφηκαν στα προηγούμενα κεφάλαια.

5.1 Προβλήματα και Εξέλιξη

Στη διάρκεια της διπλωματικής συνάντησα αρκετές δυσκολίες τόσο στη σχεδίαση όσο και στη κατασκευή του δικτύου. Κλήθηκα να βρω τρόπους για να δυσκολέψω την ανάλυση της δικτυακής κίνησης η οποία μπορεί να οδηγήσει σε συμπεράσματα για τη ταυτότητα του χρήστη και να αξιολογήσω την λύση μου όχι μόνο προς την ασφάλεια αλλά και προς την απόδοση την οποία θα έχει το δίκτυο μου. Ένα από τα βασικά προβλήματα της ασφάλειας είναι ότι αρκετές φορές όταν προσπαθείς για το μέγιστο (στην ασφάλεια) χάνεις στην απόδοση και στη λειτουργικότητα το οποίο με τη σειρά του οδηγεί στην αχρήστευση της κατασκευής. Γι’ αυτό το λόγο κατά τη διάρκεια της διπλωματικής μου προσπάθησα να δώσω την ασφάλεια που είχα θέσει ως στόχο, με σεβασμό στην απόδοση του δικτύου και αυτό το κατάφερα επιλέγοντας κατάλληλες λύσεις όπως η καθυστέρηση αποστολής με τις κατάλληλες παραμέτρους (χρόνος καθυστέρησης). Μια ακόμη δυσκολία κατά την έκβαση της διπλωματικής μου ήταν η έλλειψη βιβλιοθηκών για μηχανισμό τον οποίο επέλεξα να χρησιμοποιήσω με αποτέλεσμα να πρέπει να τον μελετήσω και να τον κατασκευάσω. Ένα τέτοιος μηχανισμός ήταν ο Diffie-Hellman ο οποίος απαιτούσε ιδιαίτερη προσοχή καθώς είναι ο αλγόριθμος ο οποίους γεννάει κλειδιά για τη κρυπτογράφηση των αναφορών, το οποίο καθιστά το συγκεκριμένο μηχανισμό βασικό μέρος για την διασφάλιση της εμπιστευτικότητας των αναφορών του χρήστη αλλά και της ανωνυμίας του.

Στο δίκτυο που σχεδίασα στη παρούσα διπλωματική υπάρχει εξέλιξη και μελλοντικές βελτιώσεις οι οποίες μπορούν να γίνουν ώστε να κάνουν το δίκτυο πιο λειτουργικό και ανθεκτικό. Μια μελλοντική προσθήκη για την εξέλιξη του παρόντος δικτύου μπορεί να είναι η

υιοθέτηση του μηχανισμού ανωνυμοποίησης τον οποίο σχεδίασαν και κατασκεύασαν οι ερευνητές Χρήστος Χατζηγεωργίου, Λάζαρος Τουμανίδης, Κόγιας Δημήτριος, Χαράλαμπος Πατρικάκης και Eric Jacksch, ο οποίος χρησιμοποιεί έναν proxy server για να αφαιρέσει πληροφορία η οποία σχετίζεται με την ταυτότητα του ατόμου και χρησιμοποιείται επίσης για αποστολή αναφορών σε υπηρεσίες προστασίας του πολίτη. Με αυτό τον τρόπο ο χρήστης θα είναι σίγουρος για την διασφάλιση της ανωνυμίας του καθώς και σε περίπτωση που εισάγει κάτι που μπορεί να τον ταυτοποιεί , αυτό θα αφαιρείται και στη συνέχεια το μήνυμα θα στέλνεται στην αντίστοιχη αρχή. Ένα ερωτηματικό στην υλοποίηση της παρούσας διπλωματικής είναι η ύπαρξη ενός μηχανισμού ο οποίος θα μπορεί να αξιολογεί τις αναφορές και να βγάζει συμπεράσματα για την ύπαρξη ενός συμβάντος. Λόγω του ότι δεν ζούμε σε έναν κόσμο ιδανικό είναι πολύ πιθανό το δίκτυο να χρησιμοποιηθεί για ψευδείς αναφορές οι οποίες μπορεί να είναι μια φάρσα ή το χειρότερο σενάριο, κακόβουλοι χρήστες οι οποίοι να προσπαθήσουν να απασχολήσουν αρχές προστασίας του πολίτη με στόχο μια εγκληματική. Επομένως κρίνεται αναγκαία η εξέλιξη του δικτύου έτσι ώστε να μπορεί να διαχειριστεί τέτοιες περιπτώσεις.

Η υιοθέτηση του δικτύου το οποίο παρουσίασα στη παρούσα διπλωματική από αρχές προστασίας του πολίτη θα μπορούσε να οδηγήσει σε αύξηση της συμμετοχής των πολιτών στην αναφορά συμβάντων ώστε και οι αρχές προστασίας να δρουν γρηγορότερα και πιο αποτελεσματικά. Η ανωνυμία η οποία προσφέρει το δίκτυο σε συνδυασμό με την μη επικοινωνία του χρήστη με την αρχή (η αποστολή μιας αναφοράς είναι μονοδρομη) μπορεί να δώσει τη σιγουριά και το θάρρος που χρειάζεται ένας πολίτης για να συνεργαστεί με μια αρχή προστασίας. Εξελίσσοντας το δίκτυο με όσα αναφέρθηκαν θα αποτελέσει μια ολοκληρωμένη λύση τόσο για τους πολίτες όσο και για τις αρχές προστασίας που θα το υιοθετήσουν.

Παράρτημα Α

Κώδικας Onion Proxy

```
import random
from socket import *
from ssl import wrap_socket
import hashlib
import json
from Crypto.Cipher import AES
import base64
import pprint

IV = "Dg7zIVomY8Zo44ZT"


def diffie_hellman(public_b=None, id=None):

################################################################################
##########
    #
#
    #                   ****Diffie-Hellman Symmetric key exchange****
#
    #
#
    #                       q = large prime number
#
    #                       random_a = random number < q (secret)
#
    #                       g = static number = 2
#
    #                       public_a = (g**random_a) mod q
#
    #                       random_b = random integer for b < q (secret)
#
    #                       key_a = (public_b**random_a) mod q - must be equal
to key_b     #
    #                       key_b = (public_a**random_b) mod q - must be equal
to key_a     #
    #
#

################################################################################
##########

    if public_b:
        # calculate the key
        key = pow(public_b, random_a, q)
        session_key = hashlib.sha256(str(key).encode()).hexdigest()[:32]

        keys.append({'id': id,
                     'key': session_key})
        return
    else:
        # calculate the public_a
        public_a = pow(g, random_a, q)

        return [str(public_a), str(g), str(q)]

    return
```

```
def create_packet(num=2, data=None, relay=None, cmd=None, exit="False"):
    json_data = {}
    if cmd == 'create':
        dh = diffie_hellman()

        json_data['cmd'] = cmd
        json_data['relay'] = [hops[(num + 1)]['host'], str(hops[(num +
1)]['port'])]
        json_data['id'] = None
        json_data['data'] = {
            'g': str(dh[1]),
            'public_a': str(dh[0]),
            'q': str(dh[2])
        }
        if num == 0:
            return json_data
        else:

            return create_packet(num - 1, data=json_data, cmd="extend")
    elif cmd == 'extend':
        json_data['cmd'] = cmd
        json_data['relay'] = [hops[(num + 1)]['host'], str(hops[(num +
1)]['port'])]
        json_data['id'] = None
        json_data['data'] = data
        if num == 0:
            return json_data
        else:
            return create_packet(num - 1, data=json_data, cmd="extend")
    elif cmd == 'data':
        keys_len = len(keys) - 1
        key = keys[num]['key']
        id = keys[num]['id']

        json_data['cmd'] = cmd
        json_data['id'] = str(id)
        json_data['packet'] = {
            'relay': [hops[(num + 1)]['host'], str(hops[(num + 1)]['port'])],
            'exit': exit,
            'data': data
        }
        # print (key)
        encryption_suite = AES.new(key, AES.MODE_CFB, IV)
        json_data['packet'] = json.dumps(json_data['packet'])
        cipher_text = encryption_suite.encrypt(json_data['packet'])
        json_data['packet'] = base64.b64encode(cipher_text)
        pprint.pprint(json_data)
        print("\n\n\n")
        if num == 0:
            return json.dumps(json_data)
        else:
            return create_packet(num - 1, data=json.dumps(json_data),
cmd="data")

    return

# ------------------------------------------ Create the circuit (keys) -------
-------------------------------------------------
```

```
# Diffie Hellman Parameters
q = 20560288036117
g = 2
random_a = random.randint(20, 100)

num_of_keys = 0
keys = []
hops = [
    {
        'host': '83.212.72.234',
        'port': 6668,
    },
    {
        'host': '83.212.74.86',
        'port': 6668
    },
    {
        'host': '83.212.74.87',
        'port': 6670,
    },
    {
        'host': 'localhost',
        'port': 6671
    }
]

while num_of_keys <= 2:
    client_socket = socket(AF_INET, SOCK_STREAM)
    wrappedSocket = wrap_socket(client_socket)

    # Connect to the node
    wrappedSocket.connect(('83.212.72.234', hops[0]['port']))

    json_data = create_packet(num=num_of_keys, cmd="create")

    data_out = json.dumps(json_data)

    # send dataout
    wrappedSocket.send(data_out.encode())

    # receive data
    data_in = (client_socket.recv(1024)).decode()

    json_data = json.loads(data_in)

    if json_data['cmd'] == 'created' or json_data['cmd'] == 'extended':
        diffie_hellman(int(json_data['data']['public_b']),
str(json_data['data']['id']))
        num_of_keys = num_of_keys + 1

    # End session
    client_socket.close()

# ------------------------------------ SEND THE REPORT -----------------------
-------------------------------------

message = "The house is down!!!"
data = {'message': message}
packet = create_packet(data=data, cmd="data", exit="True")
print keys

client_socket = socket(AF_INET, SOCK_STREAM)
```

```
wrappedSocket = wrap_socket(client_socket)

# Connect to the node
wrappedSocket.connect(('83.212.72.234', hops[0]['port']))
wrappedSocket.send(packet.encode())
```

Κώδικας Onion – Router

```
import datetime
import random
import uuid
from ssl import *
import json
import hashlib
import threading
import base64
from Crypto.Cipher import AES
from time import sleep


# Class for the Clients in threads
class ClientThread(threading.Thread):
    def __init__(self, server_socket):
        threading.Thread.__init__(self)
        self.server_socket = server_socket

    def diffie_hellman(self, q, g, public_a):

##########################################################################
##########
        #
#
        #                   ****Diffie-Hellman Symmetric key exchange****
#
        #
#
        #                       q = large prime number
#
        #                       random_a = random number < q (secret)
#
        #                       g = static number = 2
#
        #                       public_a = (g**random_a) mod q
#
        #                       random_b = random integer for b < q (secret)
#
        #                       key_a = (public_b**random_a) mod q - must be
equal to key_b     #
        #                       key_b = (public_a**random_b) mod q - must be
equal to key_a     #
        #
#

##########################################################################
##########

        # Huge prime number which is the secret value
        q = 20560288036117
        # g a random value which will be shared between the two parties
        g = 2
        random_b = random.randint(20, 100)
```

```
        # calculate the public_a
        public_b = pow(g, random_b, q)

        # calculate the key
        key = pow(public_a, random_b, q)
        session_key = hashlib.sha256(str(key).encode()).hexdigest()[:32]

        id = uuid.uuid4()

        keys.append({'id': id,
                     'key': session_key})

        return {'id': str(id),
                'public_b': str(public_b)}

    def run(self):

        while True:
            connection, address = server_socket.accept()
            sslll = wrap_socket(connection,
                                server_side=True,
                                keyfile='./cert/server.key',
                                certfile='./cert/server.crt')
            data_in = sslll.read()

            if data_in:
                message = data_in.decode()
                # Convert the incoming string to JSON
                data_in = json.loads(message)

                # Case 1: Create b
                if data_in['cmd'] == 'create':
                    print("Create: "+str(datetime.datetime.now().time()))
                    dh_parameters = data_in['data']

                    # Calculate the session key using diffie-hellman
                    response = self.diffie_hellman(q=int(dh_parameters['q']),
                                                   g=int(dh_parameters['g']),
public_a=int(dh_parameters['public_a']))

                    # The packet to be returned
                    data_out = data_in
                    data_out['cmd'] = "created"
                    data_out['data'] = response

                    data_out = json.dumps(data_out).encode()
                    # connection.send(data_out)

                    # Random Delay
                    sleep(random.randint(10, 40))
                    connection.send(data_out)
                    connection.close()
                elif data_in['cmd'] == 'extend':
                    print("Extend: "+str(datetime.datetime.now().time()))
                    forward = data_in['data']
                    relay_socket = socket(AF_INET, SOCK_STREAM)
                    wrappedSocket = wrap_socket(relay_socket)

                    wrappedSocket.connect((data_in['relay'][0],
int(data_in['relay'][1])))
```

```
                    # Random Delay
                    sleep(random.randint(10, 40))

                    wrappedSocket.send(json.dumps(forward).encode())


                    reply = relay_socket.recv(1024)

                    relay_socket.close()

                    # Random Delay
                    sleep(random.randint(10, 40))

                    # data_out = json.dumps(json_data).encode()
                    connection.send(reply)

                    # connection.send(data_out)
                    connection.close()
                elif data_in['cmd'] == 'data':
                    packet = base64.b64decode(data_in['packet'])
                    id = data_in['id']
                    key = None
                    for client in keys:
                        if str(client['id']) == id:
                            key = client['key']
                    encryption_suite = AES.new(key, AES.MODE_CFB, IV)
                    plain_text = encryption_suite.decrypt(packet)
                    plain_text = json.loads(plain_text)
                    print("Data: " + str(datetime.datetime.now().time()))
                    print ("\nEncrypted:\n"+str(packet))
                    print("\n\nPlain Text:\n"+json.dumps(plain_text))
                    if plain_text['exit'] == "True":
                        # HTTP Request
                        print("Relay 3 - Provide Report to Police")
                    else:

                        forward = plain_text['data']
                        relay_socket = socket(AF_INET, SOCK_STREAM)
                        wrappedSocket = wrap_socket(relay_socket)

                        wrappedSocket.connect((plain_text['relay'][0],
int(plain_text['relay'][1])))

                        # Random Delay
                        sleep(random.randint(10, 40))

                        wrappedSocket.send(forward)

                        reply = relay_socket.recv(1024)

                        wrappedSocket.close()

                        # Random Delay
                        sleep(random.randint(10, 40))

                        # data_out = json.dumps(json_data).encode()
                        sslll.send(reply)

                        # connection.send(data_out)
                        sslll.close()
```

```
# Main Section of the Programm
host = "83.212.72.234"
port = 6668
server_socket = socket(AF_INET, SOCK_STREAM)
server_socket.bind((host, port))
server_socket.listen(5)

keys = []
IV = "Dg7zIVomY8Zo44ZT"

print("Server started")
newthread = ClientThread(server_socket)
newthread.start()
```

Βιβλιογραφία

[1] torproject Available at: https://metrics.torproject.org/userstats-relay-country.html [Πρόσβαση Νοέμβριος 2018].

[2] Wikipedia, "Onion routing" Available at: https://en.wikipedia.org/wiki/Onion_routing [Πρόσβαση Νοέμβριος 2018].

[3] Wikipedia, "Transport Layer Security" Available at: https://en.wikipedia.org/wiki/Transport_Layer_Security [Πρόσβαση Δεκέμβριος 2018].

[4] Michael G.Reed, P. F. S. D. M. G., 1998, "Anonymous Connections and Onion Routing."

[5] Paul Syverson, M. G. R. D. G., 1996. "Onion Routing" Available at: https://www.onion-router.net/Publications/IH-1996.pdf [Πρόσβαση Δεκέμβριος 2018].

[6] Roger Dingledine, R. D. R. D., 2004, "Tor: The Second-Generation Onion Router"

[7] Tiwari, A., 2017, Fossbytes , " Everything About Tor: What is Tor? How Tor Works?" Available at: https://fossbytes.com/everything-tor-tor-tor-works [Πρόσβαση Δεκέμβριος 2018].

[8] CryptoWiki, "Anonymity networks", Available at: http://cryptowiki.net/index.php?title=Anonymity_networks [Πρόσβαση Δεκέμβριος 2018].

[9] "A Gentle Introduction to How I2P Works", Available at: https://geti2p.net/en/docs/how/intro [Πρόσβαση Δεκέμβριος 2018].

[10] Filip Jelic, DeepDotWeb, "Tor's Biggest Threat – Correlation Attack", Available at: https://www.deepdotweb.com/2016/10/25/tors-biggest-threat-correlation-attack [Πρόσβαση Δεκέμβριος 2018].

[11] tutorialspoint, "SDLC - Waterfall Model", Available at: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm [Πρόσβαση Δεκέμβριος 2018].

[12] Pierluigi Paganini, securityaffairs, "81 percent of Tor clients can be identified with traffic analysis attack", Available at: https://securityaffairs.co/wordpress/30202/hacking/tor-traffic-analysis-attack.html [Πρόσβαση Δεκέμβριος 2018].

[13] "Pylint User Manual", Available at: https://docs.pylint.org/en/1.6.0/index.html

[14] Jim Thomas, "Why Don't People Report Crimes to the Police?", Available at: https://legalbeagle.com/5733254-dont-people-report-crimes-police.html [Πρόσβαση Ιανουάριος 2019]

[15] Carol Fredrickson, “7 Reasons Employees Don't Report Workplace Violence”, Available at: https://www.businessknowhow.com/manage/reportviolence.html [Πρόσβαση Ιανουάριος 2019]